Privacy-Preserving Data Search with Fine-grained Dynamic Search Right Management in Fog-assisted Internet of Things

Rang Zhou, Xiaosong Zhang*, Xiaofen Wang

Center for Cyber Security and School of Computer Science and Engineering, University of Electronic Science and Technology of China,
Chengdu, Sichuan, China

Guowu Yang

Big Data Research Center and School of Computer Science and Engineering, University of Electronic Science and Technology of China,
Chengdu, Sichuan, China

Hao Wang

Information Technology and Electrical Engineering, Norwegian University of Science and Technology,
Hovedbygget, B318, Ålesund, Norway

Yulei Wu*

Department of Computer Science, College of Engineering, Mathematics, and Physical Sciences, University of Exeter,
Exeter, EX4 4QF, United Kingdom

Abstract

Fog computing, as an assisted method for cloud computing, collects Internet of Things (IoT) data to multiple fog nodes on the edge of IoT and outsources them to the cloud for data search, and it reduces the computation cost on IoT nodes and provides fine-grained search right management. However, to provide privacy-preserving IoT data search, the existing searchable encryptions are very inefficient as the computation cost is too high for the resource-constrained IoT ends. Moreover, to provide dynamic search right management, the users need to be online all the time in the existing schemes, which is impractical. In this

[☆]Fully documented templates are available in the elsarticle package on CTAN.
[1]Corresponding authors: Xiaosong Zhang and Yulei Wu.

paper, we first present a new fog-assisted privacy-preserving IoT data search framework, where the data from each IoT device is collected by a fog node, stored in a determined document and outsourced to the cloud, the users search the data through the fog nodes, and the fine-grained search right management is maintained at document level. Under this framework, two searchable encryption schemes are proposed, i.e. Credible Fog Nodes assisted Searchable Encryption (CFN-SE) and Semi-trusted Fog Nodes assisted Searchable Encryption (STFN-SE). In CFN-SE scheme, the indexes and trapdoors are generated by the fog nodes, which greatly reduce the computation costs at the IoT devices and user ends, and fog nodes are used to support offline users' key update. In STFN-SE scheme, the semi-trusted fog nodes are used to provide storage of encrypted key update information to assist offline users' search right update. In both schemes, no re-encryption of the keywords is needed in search right updates. The performance evaluations of our schemes demonstrate the feasibility and high efficiency of our system.

Keywords: Searchable encryption, Internet of Thing, Fine-grained search right management, Fog node.

1. Introduction

The Internet of Things (IoT) is a kind of network where massive physical devices are connected with each other through wireless or wired links. In the last few years, the industry has witnessed a rapid expansion of IoT devices. It is estimated that there will be around 50 billion IoT devices by 2020 [2].

Due to massive physic devices in IoT network, a great amount of IoT data is produced, and cloud is introduced to store these data [31]. However, the cloud server is not full-trusted. Thus, the sensitive IoT data should be encrypted before outsourced to the cloud server. If an authorized user wants to retrieve

[2]URL:https://spectrum.ieee.org/tech-talk/telecom/internet/popular-internet-of-things-forecast-of-50-billion-devices-by-2020-is-outdated

the specific data, a privacy-preserving data search service must be provided. Researchers have designed many searchable encryption schemes [1, 5, 9, 10, 12, 13, 16, 17, 18, 19, 21, 30, 34, 36] to meet this requirement.

In a practical IoT system, user members always are changed. For example, in a company or organization, as some new workers are employed and pervious workers may leave, data search rights are authorized to the new workers and reclaimed from the left workers. Thus, the users' search rights are changed dynamically, including search right authorization and revocation, which can be achieved through key update or data index update in searchable encryption schemes. In the previous studies [1, 9, 10, 30], the search right management is user-oriented, where the users can be either authorized or totally revoked.

However, in IoT data search system, more fine-grained search right management is needed. For instance, when a user transfers between different projects in the same company, his manager may reclaim the data search right of the former project and authorize that of the new project. For example, in a factory of industrial manufacturing, a worker changes his position from workshop A to B. At this time, the data search right for the IoT data in workshop A should be reclaimed and the data search right for the IoT data in workshop B should be authorized to the worker. Such search right management is not considered in the previous schemes. Therefore, search right revocation at document level should be introduced.

Another problem in IoT data search system is that IoT devices are computation and storage limited. To handle this problem, fog computing, which is a practical platform with strong computation power and expandable storage[32], is introduced by [20, 33]. However, in [20, 33], the data owner generates access tree and uploads all keyword indexes, which consumes large computation and communication overhead at the data owner end and leads to network congestion.

Therefore, an efficient fog assisted privacy-preserving IoT data search system with dynamic search right management at document level should be designed. With this aim, as shown in Fig.1, we propose a fog assisted privacy-preserving IoT data search framework, where the data from each IoT device is collected by a fog node, stored in a determined document and outsourced to the cloud, the users search the data through the fog node, and the fine-grained search right management is maintained at document level.

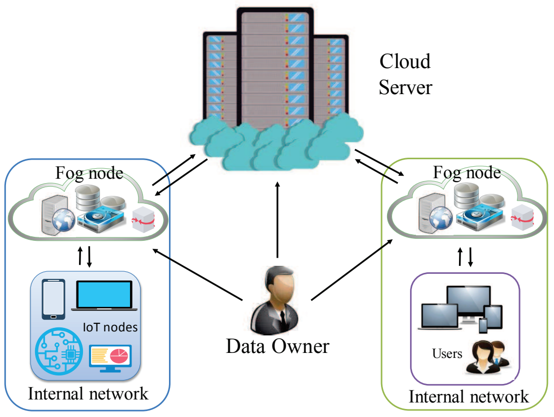


Figure 1: The fog-assisted privacy-preserving IoT data search framework

1.1. Our Contribution

In this paper, we aim to solve the problem of fine-grained search right update in fog-assisted IoT data search system. The main contributions include:

- We present a new fog-assisted privacy-preserving IoT data search framework, which meets the performance requirement for resource-constrained IoT nodes.
- We design a new keyword searchable encryption system with dynamic search right management at document level for IoT data. Moreover, the credible fog nodes are introduced to maintain the function of user key update for offline users.
- A concrete construction of credible fog node assisted searchable encryption, i.e. CFN-SE, is designed to achieve the dynamic search right update for IoT data. Comparing with previous schemes, re-encryption on keyword indexes is not needed in our construction, which greatly reduces the computation cost.

- We present a semi-trusted fog node assisted searchable encryption system with dynamic search right management at document level. A concrete scheme, i.e. STFN-SE, is constructed, which provides further fine-grained search right management, where different users have different authorized keys for the same document subset.

- We evaluate the performance of our schemes. The results show that the proposed constructions are practical for IoT applications.

The paper is organized as follows: related work is described in Section 2. In Section 3, the preliminaries are introduced. The system model of credible fog-assisted IoT data search system are described in Section 4. In Section 5, our CFN-SE scheme is proposed. Security and performance analysis are discussed in Section 6. Section 7 describes a refined system model of semi-trusted fog assisted IoT data search system and the STFN-SE scheme. The conclusion is drawn in Section 8.

2. Related Work

To resolve the problem of complex trapdoor construction in symmetric searchable encryption [27], the notion of public key encryption with keyword search (PEKS) is firstly proposed and a valid construction is designed in [3]. In recent studies, researchers concern on the constructions of multi-key searchable encryption and searchable encryption with search right update in practical applications.

Multi-key searchable encryption. The multi-key searchable encryption framework is designed by Popa et al. [23], and the first web application is built on Mylar [24], where only a single token is generated and sent to server for keyword search of different documents encrypted with different keys. Moreover, to solve the security problems under multi-owner setting in this framework, Tang et al. [29] proposed a provable security scheme. Liu et al. [15] designed a searchable encryption scheme with data sharing. However, the major problem in these schemes[15, 23, 29] is low performance, as the trapdoor size is linear with the number of documents. To achieve restricted provable security, a proxy is introduced to make a valid construction [22]. Unfortunately, heavy computation and communication overhead at the proxy are inevitable.

Key aggregation method was introduced in [7], with which a key-aggregate searchable encryption scheme [5] is proposed. In [5], for each user, authorized search key is computed from keys of all authorized files, and only one trapdoor is generated for each query. However, Kiayias et al. [18] mounted two key guessing attacks to the schemes [5]. Li et al. [13] and Liu et al. [16] proposed their improved schemes to maintain data verification and multi-owner functions. Unfortunately, in [13] and [16], similar drawbacks as in [5] can be found. To overcome the vulnerabilities in [5], Kiayias et al. proposed an improved scheme [18]. However, it leads to more communication and computation cost, as a fully trusted aid server is introduced in the scheme. To reduce the overhead, Zhou et al. proposed an improved scheme [36]. However, in their scheme [36], dynamic search right management in practical applications is not considered.

Dynamic searchable encryption. Recently, to maintain forward secrecy in dynamic searchable encryption system, the researchers introduced random number to generate different authorized search keys in different time periods. Based on BLS short signature [4], the keyword indexes are generated in two phases [1]. In the first phase, an index is computed by a corresponding complementary key stored on cloud server, and in the second phase, the other one is computed by data sender. Proxy method was introduced in [9, 10], where a search key consists of a secret key at the user side and a proxy re-encryption key at the cloud server. In the constructions [9, 10], re-encryption consumes large computation cost. Wang et al. [30] proposed a new forward secrecy searchable encryption and reduced the computation overhead of keyword index re-encryption by employing a proxy. However, it is only for peer-to-peer application. Moreover, fine-grained user right management at document level is not implemented in above schemes.

Further, the researchers introduced attribute-based keyword searchable encryption to provide fine-grained search right management[25]. In a recent study [14], the search token is computed in two steps, where one is done by a secret key holder and the other by fully trusted party. Unfortunately, the schemes [14, 25] are not applicable in IoT network due to the heavy computation overhead at the IoT devices.

Searchable encryption in IoT networks. The researchers introduced many constructions of attribute-based keyword searchable encryptions [35, 36] for IoT data search. Yang et al. [35] proposed a lightweight sharable and traceable secure mobile health system, where attribute matrix is stored at data collection ends, which is not practical for many applications, where the data collection ends, e.g. sensors, have very limited storage space. Zhou et al. [36] proposed a new construction, where the sensors collect and encrypt data without attribute matrix computation. Unfortunately, their scheme [36] is not practical if the cloud server is far from the users, and large amount of communication is needed between the IoT devices and the cloud server, as it would lead to network congestion and delay.

To solve this problem, fog-assisted Internet of Things is introduced [33]. To improve the efficiency of [33], Miao et al. [20] designed an efficient ciphertext policy attribute-based keyword searchable encryption scheme in fog computing. Their scheme supports attribute update to dynamically change user's search right. However, in [20], the data owner generates access trees and computes all keyword indexes, which are of heavy computation cost.

3. Preliminaries

3.1. Bilinear Pairing

Bilinear Map. Let two multiplicative cyclic groups $\mathbb{G}_1$ and $\mathbb{G}_2$ be of the same prime order p, and g, h be the generators of $\mathbb{G}_1$. A bilinear pairing e is a map $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ with the following properties:

1.Bilinearity: $e(g^{r_1}, h^{r_2}) = e(g, h)^{r_1 r_2}$ for all $g, h \in \mathbb{G}_1$ and $r_1, r_2 \in \mathbb{Z}_p^*$.

2.Non-degeneracy: $e(g, g) \neq 1$.

3.Computability: for any $g, h \in \mathbb{G}_1$, $e(g, h)$ can be computed efficiently.

3.2. Complexity Assumptions

We present two Multi-Sequence of Exponents Diffie-Hellman (MSE-DH) problems, which are two special cases of the general Diffe-Hellman exponent problems in [2]. The intractability of the two problems is similar to the analysis in [8].

Definition 1. ***(2n,n,1)-MSE-DDH Problem.***

Let n be an integer and $(p, \mathbb{G}_1, \mathbb{G}_2, e)$ be a bilinear map group system. Let g be a generator of $\mathbb{G}$ and $h_1 = g^{\gamma_1}, h_2 = g^{\gamma_2}$, where $\gamma_1, \gamma_2 \in \mathbb{Z}_p^*$. $\alpha, \beta_1, \beta_2, H_1, H_2, d_b \in \mathbb{Z}_p^*$. Let σ be a one-time random number, f be a random polynomial of order $\deg(f) = 1$ and co-prime with α, Z be an element in $\mathbb{G}_1$. Given the following sequences of group elements,

$$\begin{array}{lll}
g, & g^{\beta_1}, & g^{\beta_2}, \\
g^{\beta_2 d_b \alpha^1}, & \cdots, & g^{\beta_2 d_b \alpha^{n-*}}, \\
g^{\beta_2 d_b \alpha^{n+2-*}}, & \cdots, & g^{\beta_2 d_b \alpha^{n}}, \\
g^{\gamma_1 d_b \alpha^1}, & \cdots, & g^{\gamma_1 d_b \alpha^{n}}, \\
g^{\gamma_2 d_b \alpha^1}, & \cdots, & g^{\gamma_2 d_b \alpha^{2n}}, \\
g^{\beta_2 (q(\alpha) d_b H_1 + \sigma)}, & g^{\gamma_1 \beta_1 f}, & \\
g^{\beta_2 (q(\alpha) d_b H_2 + \sigma)}, & g^{\gamma_1 \beta_2 f}, & \\
g^{\beta_2 ((q(\alpha) + \alpha^*) d_b H_1 + \sigma)}, & g^{\beta_1 \sigma}, &
\end{array}$$

where $q(\alpha) = \sum_{j, j \in F_{at}} \alpha^{n+1-j}$ for any subset F_{at} and $F_{at} \subset U - F^*$, and $*$ means a random number chosen from $\{1, \cdots, n\}$, the problem is to distinguish whether Z equals to $g^{(\beta_2 H_2 + \alpha^* \gamma_2) f}$ or some random element in $\mathbb{G}_1$.

Intractability of *(2n,n,1)-MSE-DDH Problem.*

Comparing with [36], one more random number is introduced in our hard problem. Therefore, the intractability of *(2n,n,1)-MSE-DDH Problem* is similar to [36] and we omit its proof here.

Definition 2. ***(2n,n,2)-MSE-DDH Problem.***

Let n be an integer and $(p, \mathbb{G}_1, \mathbb{G}_2, e)$ be a bilinear map group system. Let g be a *generator of $\mathbb{G}$ and $h_1 = g^{\gamma_1}, h_2 = g^{\gamma_2}$, where $\gamma_1, \gamma_2 \in \mathbb{Z}_p^*$. $\alpha, \beta_1, \beta_2, H_1, H_2, d_b \in \mathbb{Z}_p^*$. Let f be random co-prime with α, whose order is $\deg(f) = 1$, $Z \in \mathbb{G}_1$. Given the following sequences of group elements,*

$$\begin{array}{lll}
g, & g^{\beta_2}, & g^{x} \\
g^{\beta_2 d_b \alpha^1}, & \cdots, & g^{\beta_2 d_b \alpha^{n-*}}, \\
g^{\beta_2 d_b \alpha^{n+2-*}}, & \cdots, & g^{\beta_2 d_b \alpha^{n}}, \\
g^{\gamma_1 d_b \alpha^1}, & \cdots, & g^{\gamma_1 d_b \alpha^{n}}, \\
g^{\gamma_2 d_b \alpha^1}, & \cdots, & g^{\gamma_2 d_b \alpha^{2n}}, \\
g^{\gamma_1 f}, & g^{\gamma_1 \beta_2 f}, & \\
g^{(\beta_2 H_2 + \gamma_2 \alpha^1) f}, & \cdots, & g^{(\beta_2 H_2 + \gamma_2 \alpha^{*-1}) f}, \\
g^{(\beta_2 H_2 + \gamma_2 \alpha^{*+1}) f}, & \cdots, & g^{(\beta_2 H_2 + \gamma_2 \alpha^{n}) f}, \\
g^{(\beta_2 H_1 + \gamma_2 \alpha^1) f}, & \cdots, & g^{(\beta_2 H_1 + \gamma_2 \alpha^{n}) f},
\end{array}$$

where $*$ means a random number chosen from $\{1, \cdots, n\}$, the problem is to distinguish whether Z equals to $g^{\beta_2(\alpha^{n+1-*} d_b H_2 + x)}$ or some random element in $\mathbb{G}_1$.

Intractability of *(2n,n,2)-MSE-DDH Problem.*
Comparing with [36], one more random number is introduced in our hard problem. Therefore, the intractability of *(2n,n,2)-MSE-DDH Problem* is similar to [36] and we omit its proof here.

4. System Model

4.1. Credible Fog-assisted Data Search System

Fig.2 shows the credible fog-assisted data search system, which consists of data owner, cloud server, user, IoT nodes and credible fog nodes, which are described as follows.

Data Owner. The data owner manages the search right of his IoT data in the system. He computes the search keys for the authorized users and stores them on the fog node for users ends, and index generation key for IoT nodes and stores them on the fog node for IoT nodes.

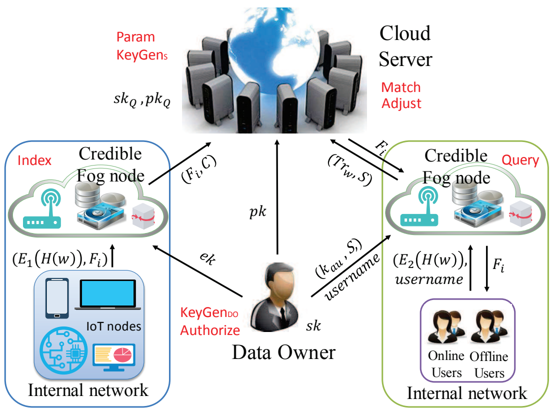


Figure 2: Credible Fog-assisted Data Search System

IoT nodes. The IoT nodes collect data, encrypt the data and the data's keyword information with symmetric ciphers, and upload them to the fog nodes for IoT nodes. In general, most of IoT nodes are resource-constrained devices.

Fog nodes. Two kinds of credible fog nodes, in internal network, are introduced to reduce the computation overhead at the IoT nodes and the user ends. The fog node for IoT nodes obtains data and keywords information from IoT nodes, computes the keywords' indexes, and outsources them to the cloud server. The fog node for user ends receives the new search keys from the data owner and stores them, and it obtains the encrypted keywords from the users to generate the trapdoors for query. Then it uploads the query trapdoors to the cloud server, and also returns the cloud server's query results to corresponding users honestly.

Cloud Server. The cloud server provides the IoT data storage and search service for the users. The cloud server is supposed to be honest but curious, who completes search queries honestly and does not modify the stored information maliciously, but it is curious about the privacy of users' search. Moreover, it does not collude with other parties to guess the keyword information from the indexes and search queries.

Users. The users obtain search right from the data owner. When they want to search data, they encrypt the keywords, send them to the fog node for user ends, and they receive the retrieved data from the fog node.

4.2. System Definition

Definition 3. *As shown in Fig.2, the keyword searchable encryption system with IoT data search right update consists of the following eight algorithms:*

Param(ξ)*: The algorithm takes as input security parameter* ξ*, and generates the global parameter* $\mathcal{GP}$ *.*

$\textbf{KeyGen}_{\textbf{S}}(\mathcal{GP})$*: The cloud server takes system parameter* $\mathcal{GP}$ *as input, and outputs server's public and secret key pair* (pk_Q, sk_Q)*.*

$\textbf{KeyGen}_{\textbf{DO}}(\mathcal{GP}, \tau)$*: The data owner takes system parameter* $\mathcal{GP}$*, time period* τ_b *as input, and outputs his/her public and secret key pair* (pk, sk) *and index generation key* ek*.*

Authorize$(\mathcal{GP}, \tau, sk, S)$*: The data owner takes the system parameter* $\mathcal{GP}$*, time period* τ_b*, data owner's private key* sk *and authorized document set* S *as input, and outputs authorized key* k_{au}*. The data owner sends* (k_{au}, S) *to each corresponding fog node for user ends through a secure channel.*

Index$(\mathcal{GP}, F_i, pk_Q, ek, w)$*: For a document* F_i $(i \in \{1, \cdots, n\})$*, fog node for IoT nodes takes system parameter* $\mathcal{GP}$*, index generation key* ek*, server's public key* pk_Q*, the document number* F_i*, keywords* w *as input, and generates the index* $Index$*.*

Query$(\mathcal{GP}, \tau, k_{au}, pk, sk_Q, w)$*: A fog node for user ends takes system parameter* $\mathcal{GP}$*, time period* τ*, authorization key* k_{au}*, data owner's public key* pk*, server's public key* pk_Q*, a keyword* w *as input, and generates query trapdoor* Tr_w*.*

Adjust$(\mathcal{GP}, \tau, pk, S, Tr_w)$*: The cloud server takes system parameter* $\mathcal{GP}$*, data owner's public key* PK*, authorized document set* S*, query trapdoor* Tr_w *as input, and outputs an adjust trapdoor* Tr_i *for each* F_i *in* S*.*

Match$(\mathcal{GP}, \tau, pk, sk_Q, S, Tr_i, Index)$*: A deterministic algorithm runs by the cloud server, which takes system parameter* $\mathcal{GP}$*, time period* τ*, data owner's public key* pk*, server's private key* sk_Q*, authorized document set* S*, an adjust* *trapdoor Tr_i, an index $Index$ as input, and outputs a symbol "True" if $Index$ contains the keyword w; Otherwise, outputs "False".*

4.3. Security Requirement

In a keyword searchable encryption system, two security properties, i.e. keyword confidentiality and trapdoor privacy, are defined, whose security model are similar as those in [36].

Definition 4. *A keyword searchable encryption scheme maintains keyword confidentiality, if no polynomial time adversary $\mathcal{A}$, who does not keep search key of the challenged document, has non-negligible advantage in winning the chosen keyword attack game.*

Definition 5. *A keyword searchable encryption scheme maintains trapdoor privacy, if no polynomial time adversary $\mathcal{A}$, who does not keep search key of the challenged document, has non-negligible advantage in winning keyword guessing attack game.*

Moreover, in the keyword searchable encryption system, user right update consists of search right granting and right revocation. When a user's search right over some specific documents are revoked, the cloud server will never return these documents as the query results. Thus, the forward secrecy must be maintained in our system.

Definition 6. Forward Secrecy. *A keyword searchable encryption scheme maintains forward secrecy, if no polynomial time adversary $\mathcal{A}$, whose search key of the challenged document is revoked, has non-negligible advantage in guessing the information of his revoked documents.*

5. The CFN-SE Scheme

Param(ξ). The algorithm takes the security parameter ξ as input and generates the bilinear parameters $(p, \mathbb{G}_1, \mathbb{G}_2, e)$. It sets the maximum number of documents as n and the keyword space as m. It chooses a generator $g \in \mathbb{G}_1$ and a collision resistant hash function $H : \{0,1\}^* \to \mathbb{Z}_p^*$. The system parameters are published as $\{p, \mathbb{G}_1, \mathbb{G}_2, e, g, n, m, H\}$.

KeyGen$_\mathbf{S}$. The cloud server chooses a random secret key $\beta_1 \in \mathbb{Z}_p^*$, computes $u = g^{\beta_1}$, and sets the server's private key and public key as $(sk_Q, pk_Q) = (\beta_1, u)$.

KeyGen$_\mathbf{DO}$ $(\mathcal{GP})$. At time period τ_b $(b = 1, \cdots, \rho)$, the algorithm randomly chooses $d_b \in \mathbb{Z}_p^*$. For authorized users, the algorithm performs the following steps:

1. Randomly choose an element $\alpha \in \mathbb{Z}_p^*$ and compute the secret keys $g_i = g^{(\alpha)^i} \in \mathbb{G}_1$ for $i = (1, 2, \ldots, 2n)$.
2. Randomly choose $\beta_2, \gamma_1, \gamma_2 \in \mathbb{Z}_p^*$ and compute the public parameters $v = g^{\beta_2} \in \mathbb{G}_1$, $h_{1,i,b} = g_i^{\gamma_1 \cdot d_b} \in \mathbb{G}_1$ for $i = (1, 2, \ldots, n)$ and $h_{2,i,b} = g_i^{\gamma_2 \cdot d_b} \in \mathbb{G}_1$ for $i = (1, 2, \ldots, n, n+1, \ldots, 2n)$.
3. Compute the index generation key $ek = (ek_1, ek_2) = (u^{\gamma_1}, v^{\gamma_1})$.
4. Destroy α.

The data owner's private key and public key are set as $sk = (\beta_2, \gamma_1, \gamma_2, \{g_i\}_{i=1,2,\ldots,2n})$ and $pk = (v, \{h_{1,i,b}\}_{i=1,2,\ldots,n}, \{h_{2,i,b}\}_{i=1,2,\ldots,n,n+1,\ldots,2n})$, respectively. Moreover, the data owner distributes the secret key ek to fog node for IoT nodes.

Authorize(sk, S). The algorithm takes as input the data owner's private key sk and the document subset $S \subseteq \{1, \ldots, n\}$, and computes the authorized key: $k_{au,b} = \prod_{j \in S} g_{n+1-j}^{\beta_2 \cdot d_b}$. The data owner securely sends $(k_{au,b}, S)$ to the fog node for user ends.

Index(pk_Q, pk, ek, F_i, l). For the keyword w_l $(l \in \{1, \ldots, m\})$ in document F_i $(i \in \{1, \ldots, n\})$, the fog node for IoT nodes randomly chooses $t_{i,l} \in \mathbb{Z}_p^*$, and computes the indexes $C = (c_{1,i,l}, c_{2,i,l}, c_{3,i,w_l})$ as:

$$c_{1,i,l} = ek_1^{t_{i,l}} = g^{\gamma_1 \beta_1 t_{i,l}}, c_{2,i,l} = ek_2^{t_{i,l}} = g^{\beta_2 \gamma_1 t_{i,l}},$$
$$c_{3,i,w_l} = (v^{H(w_l)} h_{2,i})^{t_{i,l}} = (g^{\beta_2 H(w_l)} g_i^{\gamma_2})^{t_{i,l}}.$$

Then the fog node for IoT nodes sends (C, F_i) to the cloud server.

Query$(k_{au,b}, u, v, w_l)$. The fog node for user ends chooses a random $x \in \mathbb{Z}_p^*$, generates the trapdoor $Tr_b = (Tr_{1,b}, Tr_2) = ({k_{au,b}}^{H(w_l)} v^x, u^x)$ and sends (Tr_b, S)

to the cloud server for query.

Adjust(pk, i, S, Tr). The cloud server carries out the adjust algorithm and computes the discrete trapdoors $Tr_{1,i}$ for each document F_i as:

$$Tr_{1,i,b} = Tr_{1,b} \cdot \prod_{j \in S, j \neq i} h_{2,(n+1-j+i),b} = Tr_{1,b} \cdot \prod_{j \in S, j \neq i} g_{n+1-j+i}^{\gamma_2 \cdot d_b}.$$

Match$(Tr_{1,i,b}, Tr_2, pk, sk_Q, Index)$. The cloud server does the keyword match computation for each document. For the i-th document F_i, the cloud server carries out the test as follows:

1. Compute $pub_b = \prod_{j \in S} h_{1,(n+1-j),b} = \prod_{j \in S} g_{n+1-j}^{\gamma_1 \cdot d_b}$ based on the subset S;
2. Verify the equation $\frac{e(pub_b, c_{3,i,w_l})^{\beta_1} \cdot e(c_{2,i,l}, Tr_2)}{e(Tr_{1,i,b}, c_{1,i,l})} \overset{?}{=} e(h_{2,n+1,b}, c_{1,i,l})$. If the equation holds, outputs “True”. Otherwise, “False”.

The correctness of the match algorithm in our construction is shown in equation (1).

$$\begin{aligned}
&\frac{e(pub_b, c_{3,i,w_l})^{\beta_1} \cdot e(c_{2,i,l}, Tr_2)}{e(Tr_{1,i,b}, c_{1,i,l})} \\
&= \frac{e(\prod_{j \in S} g_{n+1-j}^{\gamma_1 \cdot d_b}, (g^{\beta_2 H(w_l)} \cdot g_i^{\gamma_2})^{t_{i,l}})^{\beta_1} \cdot e(g^{\beta_2 \gamma_1 t_{i,l}}, g^{\beta_1 x})}{e(Tr_{1,b} \cdot \prod_{j \in S, j \neq i} g_{n+1-j+i}^{\gamma_2 \cdot d_b}, g^{\gamma_1 \beta_1 t_{i,l}})} \\
&= \frac{e(\prod_{j \in S} g_{n+1-j}^{\gamma_1 \cdot d_b}, g^{\beta_2 H(w_l) \beta_1 t_{i,l}}) \cdot e(\prod_{j \in S} g_{n+1-j}^{\gamma_1 \cdot d_b}, g_i^{\gamma_2 \beta_1 t_{i,l}}) \cdot e(g^{\beta_2 \gamma_1 t_{i,l}}, g^{\beta_1 x})}{e((\prod_{j \in S} g_{n+1-j}^{\beta_2 \cdot d_b})^{H(w_l)} \cdot v^x \cdot \prod_{j \in S, j \neq i} g_{n+1-j+i}^{\gamma_2 \cdot d_b}, g^{\gamma_1 \beta_1 t_{i,l}})} \\
&= \frac{e(\prod_{j \in S} g_{n+1-j}^{\gamma_1 \cdot d_b}, g^{\beta_2 H(w_l) \beta_1 t_{i,l}}) \cdot e(\prod_{j \in S} g_{n+1-j}^{\gamma_1 \cdot d_b}, g_i^{\gamma_2 \beta_1 t_{i,l}}) \cdot e(g^{\beta_2 \gamma_1 t_{i,l}}, g^{\beta_1 x})}{e((\prod_{j \in S} g_{n+1-j}^{\beta_2 \cdot d_b})^{H(w_l)}, g^{\gamma_1 \beta_1 t_{i,l}}) \cdot e(g^{\beta_2 x}, g^{\gamma_1 \beta_1 t_{i,l}}) \cdot e(\prod_{j \in S, j \neq i} g_{n+1-j+i}^{\gamma_2 \cdot d_b}, g^{\gamma_1 \beta_1 t_{i,l}})} \\
&= \frac{e(\prod_{j \in S} g_{n+1-j+i}, g)^{\gamma_1 \cdot d_b \gamma_2 \beta_1 t_{i,l}}}{e(\prod_{j \in S, j \neq i} g_{n+1-j+i}, g)^{\gamma_2 \cdot d_b \gamma_1 \beta_1 t_{i,l}}} \\
&= e(g_{n+1}, g)^{\gamma_1 \gamma_2 \cdot d_b \beta_1 t_{i,l}} \\
&= e(g_{n+1}^{\gamma_2 \cdot d_b}, g^{\gamma_1 \beta_1 t_{i,l}}) \\
&= e(h_{2,n+1,b}, c_{1,i,l}).
\end{aligned} \tag{1}$$

6. Security and Performance Analysis

6.1. Security Analysis

Assuming that the public cloud server is “honest but curious” and does not collude with the the revoked users. We analyze the security properties of our scheme including keyword confidentiality, trapdoor privacy and forward secrecy.

Theorem 1. Keyword Confidentiality. *In CFN-SE scheme, if there exists a polynomial time adversary $\mathcal{A}$, who can win the chosen keyword attack game with non-negligible probability, then we can construct an algorithm $\mathcal{B}$ which can use $\mathcal{A}$ to solve the* (2n,n,1)-MSE-DDH Problem. *Therefore, keyword confidentiality is satisfied in CFN-SE scheme.*

Proof 1. *Suppose there exists a polynomial-time adversary $\mathcal{A}$, who can win the chosen keyword attack with advantage ϵ. We build an algorithm $\mathcal{B}$, who simulates the challenger and has advantage $\frac{\epsilon}{e^2 q_f}$ in solving $(2n, n, 1)$-MSE-DDH Problem. $\mathcal{B}$'s running time is approximately the same as $\mathcal{A}$'s. The universal keyword space and the file set are assumed to be W of size m and U of size n, respectively.*

Init. *$\mathcal{A}$ declares the challenge file $F^{\#}$ with size 1.*

Setup. *$\mathcal{B}$ is given the global parameters $(p, \mathbb{G}_1, \mathbb{G}_2, e(\cdot,\cdot))$ as input and the instance of $(2n, n, 1)$-MSE-DDH. We also have $q(\alpha) = \sum_{j,j\in F_{at}} \alpha^{n+1-j}$ for any subset $F_{at} \subset U - F^{\#}$. f is random polynomial co-prime with α, whose order is $\deg(f) = 1$, and $H_1 = H(w_l)(w_l \neq w_\theta)$ and $H_2 = H(w_\theta)$. σ denotes a one-time random number which different in each query. Then, H_1 and H_2 can be computed. $\mathcal{B}$ is further given $Z \in \mathbb{G}_1$, and uses $\mathcal{A}$ as a subroutine to distinguish $Z = g^{(\beta_2 H_2 + \alpha^{\#}\gamma_2) f}$ or a random element in $\mathbb{G}_1$. If $Z = g^{(\beta_2 H_2 + \alpha^{\#}\gamma_2) f}$, $\mathcal{B}$ outputs 1; Otherwise, if Z is random, $\mathcal{B}$ outputs 0.*

To generate the system parameters, the simulator $\mathcal{B}$ sets $h_1 = g^{\gamma_1}$, $h_2 = g^{\gamma_2}$, and we have

$$v = g^{\beta_2}, \quad h_{1,1,b} = g^{\gamma_1 d_b \alpha^1}, \quad \cdots, \quad h_{1,n} = g^{\gamma_1 d_b \alpha^n},$$
$$u = g^{\beta_1}, \quad h_{2,1,b} = g^{\gamma_2 d_b \alpha^1}, \quad \cdots, \quad h_{2,n} = g^{\gamma_2 d_b \alpha^{2n}}.$$

$\mathcal{B}$ sends $\mathcal{A}$ the public key $pk = \{v, u, \{h_{1,i}\}_{i=1,2,\ldots,n}, \{h_{2,i}\}_{i=1,2,\ldots,n,n+1,\ldots,2n}\}$.

Hash Query. *$\mathcal{B}$ maintains a hash lists $L(w_l, y_l)$ and the hash list is initially set empty. w_l is queried keyword and $w_\theta(w_\theta \in \{w_0, w_1\})$ is the challenged keyword. a_l is the value of the hash query for corresponding queried keyword and a_θ is the hash query result of the challenged keyword. Upon receiving a hash query for w_l, if w_l is in the list L, $\mathcal{B}$ returns the corresponding tuple y_l to $\mathcal{A}$. Otherwise, $\mathcal{B}$ sets the hash values as*

$$H(w_l) = y_l = \begin{cases} a_\theta, & \textit{if } w_l = w_\theta, \\ a_l, & \textit{otherwise.} \end{cases}$$

Then $\mathcal{B}$ adds (w_l, y_l) to the list L and returns y_l to $\mathcal{A}$. It implicitly sets $H_1 = H(w_l)(w_l \neq w_\theta)$ and $H_2 = H(w_\theta)$.

Phase 1. *The adversary asks for the private key query and trapdoor query as follows:*

Authorization Key Query. *$\mathcal{A}$ asks for the authorization key query for F_{at}, where $F_{at} \subseteq U - F^{\#}$. $\mathcal{B}$ responds $\mathcal{A}$ with the authorization key. The result of the authorization key query k_{au,b_q} can be combined from $g^{\beta_2 d_b \alpha^1}, \cdots, g^{\beta_2 d_b \alpha^{n-\#}}$, $g^{\beta_2 d_b \alpha^{n+2-\#}}, \cdots, g^{\beta_2 d_b \alpha^{n}}$ in $(2n, n, 1)$-MSE-DDH instances.*

Trapdoor Query. *$\mathcal{A}$ asks for the trapdoor query for (w_l, y_l) in the supposed group $F_t(F_t \subseteq U)$. $\mathcal{B}$ runs the Trapdoor algorithm and responds as follows:*

Let (w_l, y_l) be the corresponding tuple in L and $H(w_l) = H_1$. If $F^{\#} \subseteq F_t$, $\mathcal{B}$ randomly chooses a one-time random number $\sigma \in \mathbb{Z}_p^*$ and computes the trapdoor $Tr = (Tr_{1,w_l}, Tr_{2,w_l})$.

1). If $F^{\#} \subseteq F_t$,

$$\begin{aligned} Tr_{1,w_l} &= {k_{au,b}}^{H(w_l)} v^\sigma = g^{\beta_2 d_b (q(\alpha)+\alpha^{\#}) H(w_l) + \beta_2 \sigma} \\ &= g^{\beta_2 d_b (q(\alpha)+\alpha^{\#}) H_1 + \beta_2 \sigma}, \\ Tr_{2,w_l} &= g^{\beta_1 \sigma}. \end{aligned}$$

2).If $F^{\#} \nsubseteq F_t$,

$$\begin{aligned} Tr_{1,w_l} &= {k_{au,b}}^{H(w_l)} v^\sigma = g^{\beta_2 d_b (q(\alpha)) H(w_l) + \beta_2 \sigma} \\ &= g^{\beta_2 d_b (q(\alpha)) H_1 + \beta_2 \sigma}, \\ Tr_{2,w_l} &= g^{\beta_1 \sigma}. \end{aligned}$$

Tr_{1,w_l}, Tr_{2,w_l} can be computed from $(2n,n,1)$-MSE-DDH instance.

Challenge. *The challenge file set is $F^{\#}$, and two same length challenge keywords are w_0 and w_1 ($w_\theta \in \{w_0, w_1\}$). $\mathcal{A}$ did not previously ask for the private key query for $F^{\#}$, and trapdoor query for (F_t, w_0) or (F_t, w_1), where $F^{\#} \subseteq F_t$. $\mathcal{B}$ randomly picks w_θ from $\{w_0, w_1\}$. Let (w_θ, y_θ) be the corresponding tuple in L. Then $\mathcal{B}$ randomly chooses $s' \in \mathbb{Z}_p^*$ and responds $\mathcal{A}$ with the challenge index $C_\theta = (c_{1,\#,\theta}, c_{2,\#,\theta}, c_{3,\#,w_\theta})$ as*

$$c_{1,\#,\theta} = g^{\gamma_1\beta_1 s'}, \quad c_{2,\#,\theta} = g^{\beta_2\gamma_1 s'}, \quad c_{3,\#,w_\theta} = Z,$$

where $H(w_\theta) = H_2$. $c_{1,\#,\theta}, c_{2,\#,\theta}$ can be computed from the element $g^{\gamma_1\beta_1 f}, g^{\beta_2\gamma_1 f}$ in $(2n,n,1)$-MSE-DDH instance.

If $Z = g^{(\beta_2 H_2 + \alpha^{\#}\gamma_2)f}$, one can verify it by implicitly setting $s' = fs''$ as

$$\begin{aligned} c_{1,\#,\theta} &= g^{\gamma_1\beta_1 s'} = (g^{\gamma_1\beta_1 f})^{s''}, \\ c_{2,\#,\theta} &= g^{\beta_2\gamma_1 s'} = (g^{\beta_2\gamma_1 f})^{s''}, \\ c_{3,\#,w_\theta} &= (g^{\beta_2 H(w_\theta) + \gamma_2\alpha^{\#}})^{s'} = (g^{(\beta_2 H_2 + \gamma_2\alpha^{\#})f})^{s''}. \end{aligned}$$

If Z is a random element in $\mathbb{G}$, the challenge index C_θ will be random from $\mathcal{A}$'s view.

Phase 2. *$\mathcal{A}$ continues to ask for the private key query for F_{at} as Phase 1. $\mathcal{A}$ continues to ask for trapdoor query for $(F_t, w_l) \notin \{(F_t, w_0), (F_t, w_1)\}$, where the queried file set F_t and $F^{\#} \subseteq F_t$, or (F_t, w_l), where $F^{\#} \nsubseteq F_t$. $\mathcal{B}$ randomly chooses a one-time random number $\sigma \in \mathbb{Z}_p^*$ and computes the trapdoor $Tr = (Tr_{1,w_l}, Tr_{2,w_l})$.*

1). If $F^{\#} \subseteq F_t$, and $w_l \in \{w_0, w_1\}$, outputs failure and aborts the game.

2). If $F^{\#} \subseteq F_t$, and $w_l \notin \{w_0, w_1\}$, $H(w_l) = H_1$.

$$\begin{aligned} Tr_{1,w_l} &= {k_{au,b}}^{H(w_l)} v^\sigma = g^{\beta_2 d_b (q(\alpha)+\alpha^{\#})H(w_l) + \beta_2\sigma} \\ &= g^{\beta_2 d_b (q(\alpha)+\alpha^{\#})H_1 + \beta_2\sigma}, \\ Tr_{2,w_l} &= g^{\beta_1\sigma}. \end{aligned}$$

3). If $F^{\#} \nsubseteq F_t$, and $w_l \notin \{w_0, w_1\}$, $H(w_l) = H_1$.

$$\begin{aligned} Tr_{1,w_l} &= {k_{au,b}}^{H(w_l)} v^{\sigma} = g^{\beta_2 d_b q(\alpha) H(w_l) + \beta_2 \sigma} \\ &= g^{\beta_2 d_b q(\alpha) H_1 + \beta_2 \sigma}, \\ Tr_{2,w_l} &= g^{\beta_1 \sigma}. \end{aligned}$$

4). If $F^{\#} \nsubseteq F_t$, and $w_l \in \{w_0, w_1\}$, $H(w_\theta) = H_2$.

$$\begin{aligned} Tr_{1,w_\theta} &= {k_{au,b}}^{H(w_\theta)} v^{\sigma} = g^{\beta_2 d_b q(\alpha) H(w_\theta) + \beta_2 \sigma} \\ &= g^{\beta_2 d_b q(\alpha) H_2 + \beta_2 \sigma}, \\ Tr_{2,w_\theta} &= g^{\beta_1 \sigma}, \end{aligned}$$

$Tr_{1,w_\theta}, Tr_{2,w_\theta}$ can be computed from $(2n, n, 1)$-MSE-DDH instance.

Guess. *$\mathcal{A}$ outputs its guess θ'. $\mathcal{B}$ outputs 1 if $\theta' = \theta$; otherwise, outputs 0.*

We can describe the simulation as that if $Z = g^{(\beta_2 H(w_\theta) + \gamma_2 \alpha^{\#}) f}$, the challenge index is a right index which can be checked.

It is indistinguishable between the simulation and the actual attack. Therefore, when $Z = Z^{\#} = g^{(\beta_2 H(w_\theta) + \gamma_2 \alpha^{\#}) f}$, the probability that θ' is a correct guess of θ is $\Pr[\theta' = \theta | Z = Z^{\#}] = \epsilon$. In phase 1 and phase 2, assume that $\mathcal{A}$ asks for trapdoor queries and authorization key queries at most q_f and q_a times, respectively. Therefore, the probability of the event that $\mathcal{B}$ does not abort in the trapdoor queries is $1 - \left(\frac{1}{q_f}\right)^{2((2^n-1)-(2^{n-1}-1))} \geqslant 1/e$ and the probability of the event that $\mathcal{B}$ does not abort in the authorization key queries is $1 - \left(\frac{1}{q_a}\right)^{((2^n-1)-(2^{n-1}-1))} \geqslant 1/e$. $\mathcal{B}$ will abort if $\mathcal{A}$ does not choose w_0 or w_1 in the challenge phase. Therefore, the probability of the event that $\mathcal{B}$ does not abort in the challenge queries is at least $1/q_f$.

The adversary here is similar as that in [36]. Therefore, the probability of the adversary $\mathcal{A}$ to win the attack game is at least $\epsilon/e^2 q_f$ when $\mathcal{B}$ does not abort, where e is a constant size named natural logarithm in mathematics. The running time on $\mathcal{B}$ is almost the same as that on $\mathcal{A}$. In conclusion, the keyword confidentiality is maintained in CFN-SE.

Theorem 2. Query Privacy. *In CFN-SE scheme, if there exists a polynomial time adversary $\mathcal{A}$, who can win the keyword guessing attack game with non-negligible probability, then we can construct an algorithm $\mathcal{B}$ which can use $\mathcal{A}$ to* *solve the* (2n,n,2)-MSE-DDH Problem. *Therefore, the query privacy is satisfied in CFN-SE scheme.*

Proof 2. *The query privacy in CFN-SE scheme can be proved similarly as that of Theorem 1. Therefore, we omit its proof.*

Theorem 3. Forward Secrecy. *The forward secrecy is maintained in our CFN-SE scheme.*

Proof 3. *In the time period τ_b, the revoked users could not get the new short time authorized key $k_{au,b}$ from the data owner and only has the old short time authorized $k_{au,past}$ in time period τ_{past}. There are two methods for the malicious users to guess infer the information of the revoked documents.*

In the first case, the malicious user infers the keyword information of the revoked document directly. This is similar as the chosen keyword attack in Theorem 1, which has been proved. Thus, the malicious user can not obtain the keyword information of the revoked documents.

In the second case, the malicious user generates the trapdoor $Tr_{past} = (Tr_{1,past}, Tr_2)$ from the old key $k_{au,past}$, and submits Tr_{past} to search the revoked documents. Upon receiving Tr_{past}, the server does the adjust and match algorithms as follows:

1. *Compute $Tr'_{1,i,b} = Tr_{1,past} \cdot \prod_{j \in S, j \neq i} h_{2,(n+1-j+i),b} = Tr_{1,past} \cdot \prod_{j \in S, j \neq i} g_{n+1-j+i}^{\gamma_2 \cdot d_b}$*
2. *Compute the $pub_b = \prod_{j \in S} h_{1,(n+1-j),b} = \prod_{j \in S} g_{n+1-j}^{\gamma_1 \cdot d_b}$ based on the subset S.*
3. *Test the equation $\dfrac{e(pub_b, c_{3,i,w_l})^{\beta_1} \cdot e(c_{2,i,l}, Tr_2)}{e(Tr'_{1,i,b}, c_{1,i,l})} = e(h_{2,n+1,b}, c_{1,i,l})$*

Due to the fact that $Tr'_{1,i,b} \neq Tr_{1,i,b}$, the test equation does not hold. Then the revoked documents will not be returned to the user.

In conclusion, the forward secrecy is achieved in CFN-SE scheme.

6.2. Performance Analysis

In our CFN-SE scheme, **Param**, **KeyGen$_\mathbf{S}$**, **Adjust** and **Match** are run on the cloud server. **KeyGen$_\mathbf{DO}$** and **Authorize** are run at the data owner's end and **Encrypt** is run at the fog node for IoT nodes. **Query** is run at the fog node for user ends. In general, the cloud server is a computer cluster, which has rich computation power and can complete many computation operations quickly. Thus, we only analyze the computation overheads at the data owner's end and the fog nodes.

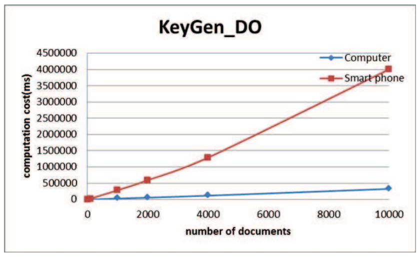
Two different experimental settings are used to implement our system. The JPBC and PBC library[3] are used to complete the experiments. The experiments are carried out on both smart phone and computer. The smart phone has a 64-bit 8 core CPU processor (4 core processor runs at 1.5 GHz and 4 core processors runs at 1.2GHz), 3GB RAM, and Android 5.1.1 operation system is run in it. The computer has Intel Core i3-2120 CPU @3.30 GHz, 4.00GB RAM, and windows 7 64-bits operation system is run in it.

In our implementations, the type A elliptic curve $E : y^2 = x^3 + x$ of 160-bit group order is used. To maintain the security, our experiment is conducted as $|Z_p| = 160$ bits, $|\mathbb{G}_1| = 1024$ bits and $|\mathbb{G}_2| = 1024$ bits.

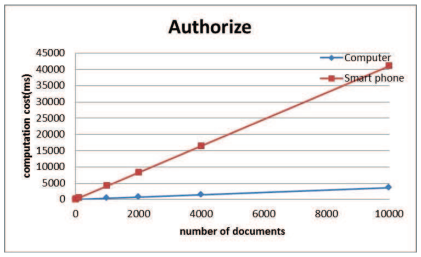
The experimental results are shown in Fig.3. The computation costs of $\mathbf{KeyGen_{DO}}$ and **Authorize** running at the data owner's end are shown in Fig.3 (a) and (b). The computation cost of **Index** running at the fog node for IoT nodes is shown in Fig.3 (c), and the computation cost of **Index** running at the fog node for user ends is shown in Fig.3 (d). The evaluation analysis is shown as follows:

Data owner: The algorithms $\mathbf{KeyGen_{DO}}$ and **Authorize** are run at the data owner's end. From the results shown in Fig.3 (a) and (b), the computations costs of $\mathbf{KeyGen_{DO}}$ and **Authorzie** are linear with the maximum number of documents and authorized search documents for one user, respectively. Especially, when $n = 1000$, it takes 28067 ms and 283620 ms to run $\mathbf{KeyGen_{DO}}$ at a computer and a smart phone, respectively. In fact, before each period of search right update, the data owner can use system idle in the data owner's device to complete the operations in $\mathbf{KeyGen_{DO}}$. Thus, the computation costs

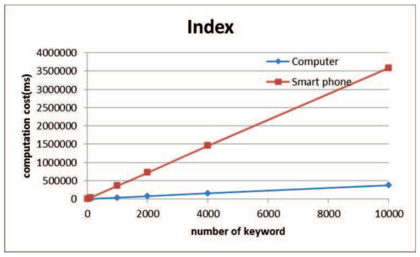
[3]URL:https://crypto.stanford.edu/pbc/



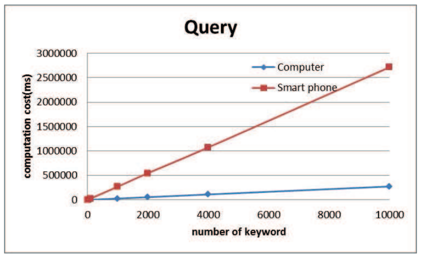
(a) Time cost of **KeyGen$_{\mathbf{DO}}$**



(b) Time cost of **Authorize**



(c) Time cost of **Index**



(d) Time cost of **Query**

Figure 3: The execution of the CFN-SE system

are acceptable.

Fog node for IoT nodes: The algorithm **Index** is run at the fog node for IoT nodes. As shown in Fig.3 (c), the computation costs of **Index** is linear with the number of keywords for each document. Furthermore, we find that the computation cost for generating every keyword's index is a constant. Thus, for both smart phone and computer, the performance of **Index** is efficient enough for practical use.

Fog node for user ends: The algorithm **Query** is run at the fog node for user ends. As shown in Fig.3 (d), the computation cost of **Query** is linear with the number of keywords in each query. Furthermore,we find that the computation cost for generating a trapdoor of each keyword is a constant. Thus, this performance is also practical for both smart phone and computer.

7. Refined Scheme on semi-trusted fog nodes

7.1. Semi-Trusted Fog-assisted Data Search System

System Framework for Semi Trusted Fog assisted Data Search. In the basic construction, the fog nodes are credible. However, in practical applications, the fog nodes are usually semi-trusted rather than full-trusted. Therefore, the fog nodes only provide computation and storage service, and complete the operations honestly, but they maybe curious about private data. To maintain data privacy protection, the IoT nodes should compute keyword indexes and the authorized users should generate query trapdoors by themselves. In addition, to meet further fine-grained search right management, different from CFN-SE scheme, the data owner should distribute different authorized secret key to each user in the system.

The semi-trusted fog-assisted data search system is shown in Fig.4. First, the semi-trusted fog nodes provide the storage and forwarding services for all ends. For instance, the semi-trusted fog nodes for IoT nodes aggregate and forward the data and indexes, which are collected by IoT devices, to the cloud server. The semi-trusted fog nodes for user ends store encrypted search keys which are given by the data owner for all authorized users. When an authorized user wants to launch a search query, it obtains the encrypted search key from the fog nodes and generates search queries by himself.

Security Requirement. The security requirements of the semi-trusted fog-assisted data search system is similar to the credible fog-assisted data search system described in Section 4, including keyword confidentiality, trapdoor privacy and forward secrecy.

7.2. System Definition

Definition 7. *The semi-trusted fog-assisted data search system consists of eight algorithms which are similar as those in Section 4. The only differences are* **Authorize** *and* **Query***, which are shown as follows:*

Authorize$(\mathcal{GP}, \tau, sk, S)$*: The data owner takes the system parameter* $\mathcal{GP}$*, time period* τ_b*, data owner's private key sk and authorized document set* S *as input,*

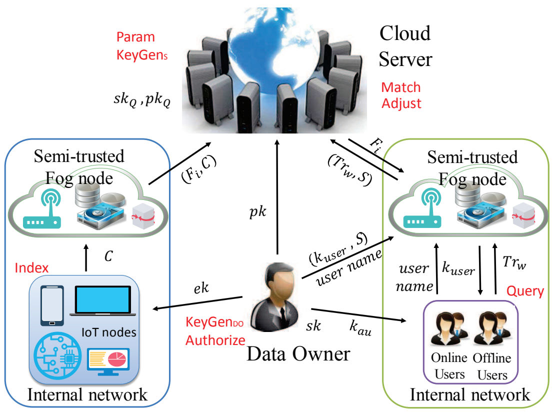


Figure 4: Semi-Trusted Fog-assisted Data Search System

and outputs the authorized secret key k_{au} and sends k_{au} to each corresponding user through a secure channel. For document set S, the data owner computes k_{user} and stores (k_{user}, S) on the fog nodes for user ends.

Query$(\mathcal{GP}, \tau, k_{user}, k_{au}, pk, pk_Q, w)$: *An authorized user takes the system parameter $\mathcal{GP}$, the time period τ, k_{user}, the data owner's public key pk, the server's public key pk_Q, a keyword w, user secret key k_{au},as input, and generates a query trapdoor Tr_w.*

7.3. The STFN-SE scheme

Param(ξ). The algorithm takes as input the security parameter ξ and generates the bilinear parameters $(p, \mathbb{G}_1, \mathbb{G}_2, e)$; It sets the maximum number of documents as n for a data owner and the keyword space as m. It Chooses a generator $g \in \mathbb{G}_1$ and three collision resistant hash functions $H : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$, $H_1 : \{0,1\}^* \rightarrow \mathbb{G}_1$, $H_2 : \{0,1\}^* \rightarrow \mathbb{G}_1$. The system parameters are published as $\{p, \mathbb{G}_1, \mathbb{G}_2, e, g, n, m, H, H_1, H_2\}$.

KeyGen$_\mathbf{S}$. The cloud server chooses a random secret key $\beta_1 \in \mathbb{Z}_p^*$, and computes $u = g^{\beta_1}$, and sets the server's private key and public key as $(sk_Q, pk_Q) = (\beta_1, u)$.

KeyGen$_\mathbf{DO}$ $(\mathcal{GP})$. At time period τ_b $(b = 1, \cdots, \rho)$, the algorithm randomly chooses $d_b \in \mathbb{Z}_p^*$ and performs the following steps:

1. Randomly choose an element $\alpha \in \mathbb{Z}_p^*$ and compute the secret keys $g_i = g^{(\alpha)^i} \in \mathbb{G}_1$ for $i = (1, 2, \ldots, 2n)$.
2. Randomly choose the secret keys $\beta_2, \beta_3, \gamma_1, \gamma_2 \in \mathbb{Z}_p^*$ and compute the public parameters $v_1 = g^{\beta_2} \in \mathbb{G}_1$, $v_2 = g^{\beta_3} \in \mathbb{G}_1$, $h_{1,i,b} = g_i^{\gamma_1 \cdot d_b} \in \mathbb{G}_1$ for $i = (1, 2, \ldots, n)$ and $h_{2,i,b} = g_i^{\gamma_2 \cdot d_b} \in \mathbb{G}_1$ for $i = (1, 2, \ldots, n, n+1, \ldots, 2n)$.
3. Compute the IoT node's secret encryption key $ek = (ek_1, ek_2) = (u^{\gamma_1}, v_1^{\gamma_1 \beta_3^{-1}})$.
4. Destroy α.

The data owner's private key and public key are set as $sk = (\beta_2, \beta_3, \gamma_1, \gamma_2, \{g_i\}_{i=1,2,\ldots,2n})$ and $pk = (v_1, v_2, \{h_{1,i,b}\}_{i=1,2,\ldots,n}, \{h_{2,i,b}\}_{i=1,2,\ldots,n,n+1,\ldots,2n})$, respectively. Moreover, the data owner distributes the secret encryption key ek to each IoT node.

Authorize(sk, S). For each user with an identity ID, the data owner computes the secret key $k_{au} = H_1(ID)^{\beta_3}$ and securely sends it to the corresponding user. For the document subset $S \subseteq \{1, \ldots, n\}$, the algorithm computes the search key $k_{au,b} = H_1(ID)^{\beta_2} \cdot \prod_{j \in S} g_{n+1-j}^{\beta_2 \cdot d_b}$ and $k_{user} = k_{au,b} \cdot H_2(H_1(ID)^{\beta_3}, h_{1,i,1})$. The data owner sends (k_{user}, S) to the fog node for user ends.

Index(pk_Q, pk, ek, F_i, l). For a keyword w_l $(l \in \{1, \ldots, m\})$ in a document F_i $(i \in \{1, \ldots, n\})$, the IoT node randomly chooses $t_{i,l} \in \mathbb{Z}_p^*$ and computes the indexes $C = (c_{1,i,l}, c_{2,i,l}, c_{3,i,w_l})$:

$$\begin{aligned} &c_{1,i,l} = ek_1^{t_{i,l}} = g^{\gamma_1 \beta_1 t_{i,l}}, c_{2,i,l} = ek_2^{t_{i,l}} = g^{\beta_2 \beta_3^{-1} \gamma_1 t_{i,l}}, \\ &c_{3,i,w_l} = (v_1^{H(w_l)} h_{2,i})^{t_{i,l}} = (g^{\beta_2 H(w_l)} g_i^{\gamma_2})^{t_{i,l}}. \end{aligned}$$

Then the IoT nodes sends the index $(c_{1,i,l}, c_{2,i,l}, c_{3,i,w_l})$ to the fog node for IoT nodes. The fog node for IoT nodes sends (C, F_i) to the cloud server.

Query$(k_{au,b}, u, v, w_l)$. A user computes the authorized key $k_{au,b} = k_{user}/H_2(k_{au}, h_{1,i,1})$. The user chooses a random $x \in Z_p^*$, computes the trapdoor $Tr_b = (Tr_{1,b}, Tr_{2,b}) = ({k_{au,b}}^{H(w_l)} v_1^x, k_{au}^{H(w_l)} v_2^x)$, and sends Tr_b to the fog node for user ends. The fog node for user ends sends (Tr_b, S) to the cloud server for keyword search.

Adjust(pk, i, S, Tr_b). The cloud server carries out the adjust algorithm and computes the discrete trapdoors $Tr_{1,i}$ for each document F_i as:

$$Tr_{1,i,b} \quad = Tr_{1,b} \cdot \prod_{j \in S, j \neq i} h_{2,(n+1-j+i),b} = Tr_{1,b} \cdot \prod_{j \in S, j \neq i} g_{n+1-j+i}^{\gamma_2 \cdot d_b}.$$

Match$(Tr_{1,i,b}, Tr_2, pk, sk_Q, Index)$. For the i-th document, the cloud server carries out the keyword match test as follows:

1. Compute $pub_b = \prod_{j \in S} h_{1,(n+1-j),b} = \prod_{j \in S} g_{n+1-j}^{\gamma_1 \cdot d_b}$ based on the subset S;
2. Verify the equation $\frac{(e(pub_b, c_{3,i,w_l}) \cdot e(c_{2,i,l}, Tr_{2,b}))^{\beta_1}}{e(Tr_{1,i,b}, c_{1,i,l})} \overset{?}{=} e(h_{2,n+1,b}, c_{1,i,l})$. If the equation holds, outputs “True”. Otherwise, “False”.

The correctness of the match test in our construction of STFN-SE is shown in equation (2).

$$
\begin{aligned}
&\frac{(e(pub_b, c_{3,i,w_l}) \cdot e(c_{2,i,l}, Tr_{2,b}))^{\beta_1}}{e(Tr_{1,i,b}, c_{1,i,l})} \\
&= \frac{(e(\prod_{j \in S} g_{n+1-j}^{\gamma_1 \cdot d_b}, (g^{\beta_2 H(w_l)} \cdot g_i^{\gamma_2})^{t_{i,l}}) \cdot e(g^{\beta_2 \beta_3^{-1} \gamma_1 t_{i,l}}, k_{au}^{H(w_l)} v_2^x))^{\beta_1}}{e(Tr_{1,b} \cdot \prod_{j \in S, j \neq i} g_{n+1-j+i}^{\gamma_2 \cdot d_b}, g^{\gamma_1 \beta_1 t_{i,l}})} \\
&= \frac{e(\prod_{j \in S} g_{n+1-j}^{\gamma_1 \cdot d_b}, (g^{\beta_2 H(w_l)} \cdot g_i^{\gamma_2})^{\beta_1 t_{i,l}}) \cdot e(g^{\beta_2 \beta_3^{-1} \gamma_1 t_{i,l}}, H_1(ID)^{\beta_3 H(w_l)} g^{\beta_3 x})^{\beta_1}}{e(k_{au,b}^{H(w_l)} \cdot v_1^x \cdot \prod_{j \in S, j \neq i} g_{n+1-j+i}^{\gamma_2 \cdot d_b}, g^{\gamma_1 \beta_1 t_{i,l}})} \\
&= \frac{e(\prod_{j \in S} g_{n+1-j}^{\gamma_1 \cdot d_b}, g^{\beta_2 H(w_l) \beta_1 t_{i,l}}) \cdot e(\prod_{j \in S} g_{n+1-j}^{\gamma_1 \cdot d_b}, g_i^{\gamma_2 \beta_1 t_{i,l}}) \cdot e(g^{\beta_2 \gamma_1 t_{i,l}}, H_1(ID)^{H(w_l)} g^x)^{\beta_1}}{e((H_1(ID)^{\beta_2} \prod_{j \in S} g_{n+1-j}^{\beta_2 \cdot d_b})^{H(w_l)} \cdot g^{\beta_2 x} \cdot \prod_{j \in S, j \neq i} g_{n+1-j+i}^{\gamma_2 \cdot d_b}, g^{\gamma_1 \beta_1 t_{i,l}})} \\
&= \frac{e(\prod_{j \in S} g_{n+1-j}^{\gamma_1 \cdot d_b}, g^{\beta_2 H(w_l) \beta_1 t_{i,l}}) \cdot e(\prod_{j \in S} g_{n+1-j}^{\gamma_1 \cdot d_b}, g_i^{\gamma_2 \beta_1 t_{i,l}}) \cdot e(g^{\beta_2 \gamma_1 t_{i,l}}, H_1(ID)^{H(w_l)} g^x)^{\beta_1}}{e(\prod_{j \in S} g_{n+1-j}^{\beta_2 \cdot d_b H(w_l)}, g^{\gamma_1 \beta_1 t_{i,l}}) \cdot e(H_1(ID)^{\beta_2 H(w_l)} g^{\beta_2 x}, g^{\gamma_1 \beta_1 t_{i,l}}) \cdot e(\prod_{j \in S, j \neq i} g_{n+1-j+i}^{\gamma_2 \cdot d_b}, g^{\gamma_1 \beta_1 t_{i,l}})} \\
&= \frac{e(\prod_{j \in S} g_{n+1-j+i}, g)^{\gamma_1 \cdot d_b \gamma_2 \beta_1 t_{i,l}}}{e(\prod_{j \in S, j \neq i} g_{n+1-j+i}, g)^{\gamma_2 \cdot d_b \gamma_1 \beta_1 t_{i,l}}} \\
&= e(g_{n+1}, g)^{\gamma_1 \gamma_2 \cdot d_b \beta_1 t_{i,l}} \\
&= e(g_{n+1}^{\gamma_2 \cdot d_b}, g^{\gamma_1 \beta_1 t_{i,l}}) \\
&= e(h_{2,n+1,b}, c_{1,i,l}).
\end{aligned}
\tag{2}
$$

7.4. Security and Performance Analysis

7.4.1. Security analysis

In STFN-SE scheme, the security properties including keyword confidentiality, trapdoor privacy and forward secrecy are similar as those of CFN-SE scheme. We analyze the security properties as follows.

Keyword Confidentiality and Query Privacy. In STFN-SE scheme, the semi-trusted fog nodes are one kind of unauthorized users and obtain the same information as the other unauthorized users. The only differences from CFN-SE scheme are the indexes $c_{2,i,l}$ in STFN-SE scheme are modified, and the public key v_2 and encrypted information k_{user} are added. From these information, the unauthorized users cannot obtain valid information to infer the search key and keywords. Therefore, the keyword confidentiality and query privacy are still maintained in STFN-SE.

Forward Secrecy. To maintain the fine-grained forward secrecy, the data owner manages the search right for each document. Every user can not retrieve his revoked encrypted documents, which are deleted from his search list. The analysis of forward secrecy is omitted, as it is similar as that for CFN-SE.

7.4.2. Performance Analysis

In STFN-SE scheme, **Param**, **KeyGen$_{\mathbf{S}}$**, **Adjust**, **Match** algorithms are run at the cloud server, which has rich computation resource and large storage. **KeyGen$_{\mathbf{DO}}$** and **Authorize** algorithms are run at the data owner's end, and **Query** is run at the user ends, both of which also have rich computation resource. In STFN-SE scheme, **Index** is run at the IoT devices, which are of very limited computation resource. Therefore, we only analyze the computation cost of **Index** run at the IoT nodes.

In practical applications, the IoT devices collect data automatically, and the computers or smart phones at the user ends do the data query. The data collection frequency at the IoT devices is much higher than that of the operations at the user end. Thus, we only analyze the operations in computing **Index**, and compare its computation and communication overhead in our scheme with those in [6, 14, 28, 35], which support the similar fine-grained search right management function. The notions P, M_1, M_2 denote pair computation, exponentiations in $\mathbb{G}_1$, exponentiations in $\mathbb{G}_2$, respectively. The value of l is the size of the authorized attribute set.

As shown in Table 1, only in our scheme, computation and communication

Table 1: Comparison of computation cost, communication overhead

scheme	computation cost	communication cost
[6]	$(2l+6)M_1 + M_2 + P$	$\|\mathbb{Z}_p^*\| + (2l+2)\|\mathbb{G}_1\|$
[14]	$(l+2)M_1 + M_2 + P$	$(l+1)\|\mathbb{G}_1\| + \|\mathbb{G}_2\|$
[28]	$(2l+2)M_1$	$(l+2)\|\mathbb{G}_1\|$
[35]	$2M_1 + 3M_2$	$2\|\mathbb{G}_1\|+2\|\mathbb{G}_2\|+2l\|\mathbb{Z}_p^*\|$
Our	$4M_1$	$3\|\mathbb{G}_1\|$

cost on IoT nodes are of constant size. Moreover, for index computation, each IoT device needs to do 4 exponentiations in $\mathbb{G}_1$ for each keyword. Therefore, this computation cost is of constant size and can be completed on resource-constrained devices.

Then, we analyze the performance on resource-constrained devices, e.g. sensors. The experimental results in [26] are used in our analysis. In [26], based on MICA2, the simulation runs on an ATmega128 8-bit processor clocked at 7.3728 MHz, 4-KB RAM, and 128-KB ROM. To maintain the 80-bit security level, we utilize the super singular curve $y^2 + y = x^3 + x$ with an embedding degree 4 and implementing η_T pairing: $E(\mathbb{F}_{2^{271}}) \times E(\mathbb{F}_{2^{271}}) \to \mathbb{F}_{2^{4*271}}$. From [11], a scalar multiplication operation in $\mathbb{G}_1$ takes 0.81 s and a exponentiation operation in $\mathbb{G}_2$ takes 0.9 s.

From Table 1, the test results show that the computation cost for running **Index** in STFN-SE scheme at IoT devices is about $4 * 0.81 = 3.24\ s$ in our scheme, and it takes about $2 * 0.81 + 3 * 0.9 = 4.32\ s$ to run **Index** in [35] at the same IoT devices. Therefore, our scheme is more efficient and practical for IoT applications.

8. Conclusion

In this paper, we propose two fog-assisted keyword searchable encryption systems on credible fog nodes and semi-trusted fog nodes, respectively. We construct two concrete schemes, which support offline users key update and fine-grained search right management at document level. The performance evaluations of our schemes demonstrate the feasibility and high efficiency of our system.

Acknowledgment

This work is supported by the National Key Research and Development Program under Grant 2017YFB0802300, National Natural Science Foundation of China under grant No. U1833122 and 61572115; the Sichuan Provincial Major Frontier Issues (2016JY0007); the fundation from the State Key Laboratory of Integrated Services Networks, Xidian University (No.ISN1809).

[1] F. Bao, R. H. Deng, X. Ding, Y. Yang, Private query on encrypted data in multi-user settings, Information Security Practice and Experience, International Conference, Ispec 2008, Springer, 2008, pp.71-85.

[2] D. Boneh, X. Boyen, E. J. Goh, Hierarchical identity based encryption with constant size ciphertext, Advances in Cryptology - EUROCRYPT 2005. Springer, 2005, pp.440-456.

[3] D. Boneh, G. D. Crescenzo, R. Ostrovsky, G. Persiano, Public key encryption with keyword search, In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 506-522. Springer, Heidelberg (2004)

[4] D. Boneh, B, Lynn, H. Shacham, Short signatures from the weil pairing, Advances in Cryptology - ASIACRYPT 2001, 7th International Conference on the Theory and Application of Cryptology and Information Security, Springer, 2001, pp.514-532.

[5] B. Cui, Z. Liu, L, Wang, Key-aggregate searchable encryption for group data sharing via cloud storage. IEEE Trans. Comput. 65(8):2374-2385, 2016.

[6] J. Cui, H. Zhou, H. Zhong, Y. Xu, AKSER: attribute-based keyword search with efficient revocation in cloud computing. Information Sciences, vol. 423, pp.343-352, 2018.

[7] C.K. Chu, S. S.M. Chow, W.G. Tzeng, J. Zhou, R. H. Deng, Key-aggregate cryptosystem for scalable data sharing in cloud storage. IEEE Trans. Parallel Distrib. Syst. 25(2), 468-477 (2014)

[8] C. Delerable, D. Pointcheval, Dynamic threshold public-key encryption, Advances in Cryptology - CRYPTO 2008, Springer, 2008, pp.317-334.

[9] C. Dong, G. Russello, N. Dulay, Shared and Searchable Encrypted Data for Untrusted Servers. Proceeedings of the, Ifip Wg 11.3 Working Conference on Data and Applications Security 2008, Springer, 2004, vol.19, pp.127-143.

[10] C. Dong, G. Russello, N. Dulay, Shared and searchable encrypted data for untrusted servers, Journal of Computer Security, 19(3):367-397 (2011).

[11] N. Gura, A. Patel, A. Wander, H. Eberle, S. C. Shantz, Comparing elliptic curve cryptography and RSA on 8-bit CPUs, in Cryptographic Hardware and Embedded Systems-CHES 2004.Springer, 2004, vol.3156, pp.119-132.

[12] Q. Huang, H. Li, An efficient public-key searchable encryption scheme secure against inside keyword guessing attacks. Information Sciences 403-404, 343-352 (2017).

[13] T. Li, Z. Liu, P. Li, C. Jia, Z. L. Jiang, J. Li, Verifiable Searchable Encryption with Aggregate Keys for Data Sharing in Outsourcing Storage, In J.K. Liu and R. Steinfeld (Eds.): ACISP 2016, Part II, LNCS 9723, pp. 153-169, 2016.

[14] K. Liang, and S. Willy: Searchable Attribute-Based Mechanism With Efficient Data Sharing for Secure Cloud Storage. IEEE Transactions on Information Forensics and Security 10(9), 1981-1992 (2015).

[15] Z. Liu, J. Li, X. Chen, J. Yang, C. Jia, TMDS: thin-model data sharing scheme supporting keyword search in cloud storage. In: S.usilo, W., Mu, Y. (eds.) ACISP 2014. LNCS, vol. 8544, pp. 115-130. Springer, Heidelberg (2014)

[16] Z. Liu, T. Li, P. Li, C. Jia, J. Li, Verifiable searchable encryption with aggregate keys for data sharing system. Future Generation Computer Systems 78, 778-788 (2018).

[17] Y. Lu, G. Wang, J. Li, Keyword guessing attacks on a public key encryption with keyword search scheme without random oracle and its improvement. Information Sciences 479, 270-276 (2019).

[18] A. Kiayias, O. Oksuz, A. Russell, Q. Tang, B. Wang, Efficient Encrypted Keyword Search for Multi-user Data Sharing. In Ioannis G. Askoxylakis, Sotiris Ioannidis, Sokratis K. Katsikas, and Catherine A. Meadows, editors, ESORICS 2016, Part I, vol. 9878 of LNCS, pages 173-195. Springer, 2016.

[19] S. Ma, Y. Mu, W. Susilo, B, Yang, Witness-based searchable encryption. Information Sciences 453, 364-378 (2018).

[20] Y. Miao, J. Ma, X. Liu, J. Weng, H. Li, H. Li, Lightweight fine-grained search over encrypted data in fog computing. IEEE Transactions on Services Computing, (99), 1-1(2018).

[21] Y. Miao, J. Weng, X. Liu, K. R. Choo, Z. Liu, H. Li, Enabling verifiable multiple keywords search over encrypted cloud data. Information Sciences 465, 21-37 (2018).

[22] C. V. Rompay, R. Molva, M. Önen, Multi-user searchable encryption in the cloud. In: López, J., Mitchell, C.J. (eds.) ISC 2015. LNCS, vol. 9290, pp. 299-316. Springer, Heidelberg (2015)

[23] R. A. PopaN. Zeldovich, Multi-Key Searchable Encryption, Cryptology ePrint Archive, Report 2013/508, 2013.

[24] R. A. Popa, E. Stark, S. Valdez, J. Helfer, N. Zeldovich, H. Balakrishnan, Building web applications on top of encrypted data using Mylar. In Proceedings of the 11th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2014, 157-172 (2014).

[25] J. Shi, J. Lai, Y. Li, R. H. Deng, J. Weng, Authorized keyword search on encrypted data. In: Kutylowski, M., Vaidya, J. (eds.) ESORICS 2014, Part I. LNCS, vol. 8712, pp. 419-435. Springer, Heidelberg (2014).

[26] K. A. Shim, Y. R. Lee, and C. M. Park, EIBAS: An efficient identity-based broadcast authentication scheme in wireless sensor networks, Ad Hoc Netw.4. Jan.2013, vol.11, no.1, pp.182-189.

[27] D. X. Song, D. Wagner, A. Perrig, Practical techniques for searches on encrypted data. In:2000 IEEE Symposium on Security and Privacy, pp. 44-55. IEEE Computer Society Press (May 2000).

[28] W. Sun, S. Yu, W. Lou, Y. Hou, H. Li, Protecting Your Right: Verifiable Attribute-based Keyword Search with Finegrained Owner-enforced Search Authorization in the Cloud, IEEE Transactions on Parallel and Distributed Systems, 2016, vol. 27, no. 4, pp. 1187-1198.

[29] Q. Tang, Nothing is for free: security in searching shared and encrypted data. IEEE Transactions on Information Forensics and Security, 9(11):1943-1952, 2014.

[30] X. Wang, Y. Mu, R. Chen, X. Zhang, Secure Channel Free ID-Based Searchable Encryption for Peer-to-Peer Group. Journal of Computer Science and Technology. 31(5), 1012-1027 (2016).

[31] X. Wang, L. T. Yang, H. Liu, M. J. Deen, A Big Data-as-a-Service Framework: State-of-the-art and Perspectives. IEEE Transactions on Big Data, 2018, 4(3), 325-340.

[32] X. Wang, L. T. Yang, X. Xie, J. Jin, M. J. Deen, A Cloud-Edge Computing Framework for Cyber-Physical-Social Services. IEEE Communications Magazine, 2017, 55(11), 80-85.

[33] M. Xiao, J. Zhou, X. Liu, M. Jiang, A hybrid scheme for fine-grained search and access authorization in fog computing environment, Sensors, vol. 17, no. 6, pp. 1423, 2017.

[34] X. Xie, X. Yang, X. Wang, H. Jin, D. Wang, X. Ke, BFSI-B: An improved K-hop graph reachability queries for cyber-physical systems. Information Fusion, 2017, 38, 35-42.

[35] Y. Yang, X. Liu, R. H. Deng, Y. Li, Lightweight Sharable and Traceable Secure Mobile Health System, IEEE Transactions on Dependable and Secure Computing. 2017, vol.99, pp.1-1.

[36] R. Zhou, X. Zhang, X. Du, X. Wang, G. Yang, M. Guizani, File-centric Multi-Key Aggregate Keyword Searchable Encryption for Industrial Internet of Things, IEEE Transactions on Industrial Informatics, 14(8), 3648-3658 (2018).