Provably Secure Searchable Attribute-Based Authenticated Encryption Scheme

Zhenhua Liu and Yaqing Fan
(Corresponding author: Zhenhua Liu)

School of Mathematics and Statistics, Xidian University
No. 2 South Taibai Road, Xi'an, Shaanxi 710071, P.R. China
(Email: zhualiu@hotmail.com)
(Received Sept. 26, 2017; revised and accepted Mar. 27, 2018)

Abstract

In cloud storage system, attribute-based encryption can support fine-gained access control over encrypted data. Furthermore, searchable attribute-based encryption can allow data users to retrieve encrypted data from a cloud storage system. However, these encryption algorithms cannot provide authenticity. In this paper, we propose a new concept——searchable attribute-based authenticated encryption and establish its security model framework. Then, we embed ingeniously search mechanism into key-policy attribute-based signcryption, and present a concrete searchable attribute-based authenticated encryption scheme. Finally, according to the proposed framework, our scheme is proven to achieve (1) Ciphertext indistinguishability under the Decisional Bilinear Diffie-Hellman Exponent hardness assumption; (2) Existential unforgeability based on the hardness assumption of Computational Diffie-Hellman Exponent problem; (3) Selective security against chosen-keyword attack under the Decisional Linear hardness assumption; (4) Keyword secrecy based on the one-way hardness of hash function.

Keywords: Attribute-Based Encryption; Authenticated Encryption; Searchable Encryption; Signcryption

1 Introduction

With the development of cloud computing, many data owners store their data in the cloud server for simplifying local IT management and reducing the cost. Although cloud services have various advantages, they will bring security and privacy concerns to upload sensitive information to the cloud server [14]. Therefore, it is essential to encrypt sensitive data before uploading them to the remote server.

Traditional public key encryption technology can protect the confidentiality of data, but cannot provide the data sharing service. However, as a kind of "one-to-many" public key encryption, attribute-based encryption can solve this problem. Thus, attribute-based encryption is considered as one of the most appropriate encryption technology to achieve the data confidentiality and expressive fine-grained access control in cloud system.

Sahai and Waters [17] first introduced the concept of attribute-based encryption, and proposed a concrete attribute-based encryption scheme which only supports threshold access policy. In order to support more flexible access policy, Goyal *et al.* [7] first presented a key-policy attribute-based encryption (KP-ABE) scheme in 2006. In key-policy attribute-based encryption, a private or decryption key is associated with access control policy and a ciphertext is computed with respect to a set of attributes. On the contrary, if a ciphertext specifies an access control policy and a private or decryption key is associated with a set of attributes, such an attribute-based encryption is called as ciphertext-policy attribute-based encryption (CP-ABE). Bethencourt *et al.* [1] proposed the first ciphertext-policy attribute-based encryption scheme in 2007. Due to the suitable application for cloud computing, a number of attribute-based encryption schemes [3, 4, 11, 12, 23, 24] have been proposed to obtain better expressive, efficiency and security.

Attribute-based encryption provides the data confidentiality and expressive fine-grained access control. Nevertheless, encryption may prevent the ciphertext from being searched quickly. To solve this problem, Song *et al.* [19] first proposed the concept of searchable encryption which provides a fundamental approach to search over encrypted cloud data. Further, Boneh *et al.* [2] presented the first public key encryption scheme with keyword search scheme, in which one can search the encrypted data by a keyword. Since then, a number of searchable public key encryption schemes [8–10, 21] have been proposed to enrich the search feature of scenarios and improve security and efficiency.

However, users are considered to be legitimate in the above search schemes, which is not suitable for more practical scenarios. The reason is that without access control, all users in these systems have not been restricted access to the entire database. Thus, the confidentiality of

sensitive data may be compromised. In order to eliminate this threat, in 2013, Wang *et al.* [22] constructed a ciphertext-policy attribute-based encryption scheme supporting keyword search function. In their scheme, data owners encrypt data with an access policy, generate the index for the corresponding keyword collection, and then store them to the cloud server. It is only when an authorized user's certificate meets the access policy that she or he can search and decrypt the encrypted data. In the recent years, attribute-based searchable encryption schemes [5, 13, 20, 27] have been made great development.

Furthermore, in 2014, to assure the cloud server to execute faithfully the search operations on behalf of the data users, Zheng *et al.* [28] proposed a novel cryptographic concept of verifiable attribute-based keyword search, and constructed a verifiable key-policy attribute-based searchable encryption scheme and a verifiable ciphertext-policy attribute-based searchable encryption scheme based on the access tree. Such search encryption schemes can support fine-grained access control and search, and verify the cloud server's faith. But these schemes can neither authenticate data owners nor guarantee the availability of the shared data. Thus, it is necessary to provide the authentication for searchable attribute-based encryption schemes.

It is well known that encryption can guarantee the confidentiality, signature can provide the authenticity, and signcryption can support these two functionalities in public key cryptosystems. In 1997, Zheng *et al.* [29] first introduced the concept of signcryption, which is a logical incorporation of signature and encryption. Subsequently, in order to ensure fine-grained access control for the data in the cloud and authenticate data owners, Gagné *et al.* [6] presented the first attribute-based signcryption (ABSC) scheme. Since then, a number of attribute-based signcryption [15, 25, 26] have been proposed. Though attribute-based signcryption schemes can ensure that data owners share the encrypted and authenticated data with data users, these schemes cannot take the retrieval of the signcryption data into account. As a result, it is important to support the searchability for attribute-based signcryption schemes.

1.1 Our Contribution

The main contribution of this paper can be summarized as follows:

- We introduce a novel concept of searchable attribute-based authenticated encryption (SAAE), which can achieve expressive fine-grained access control, efficient data retrieval and authentication, simultaneously.

- We replace access tree structure with linear secret sharing scheme to modify Zheng *et al.*'s attribute-based keyword search method [28], embed such a search mechanism into Rao *et al.*'s key-policy attribute-based signcryption scheme [16], and propose a searchable attribute-based authenticated encryption scheme. More to the point, we use an identical secret key to generate a trapdoor and decrypt a ciphertext, which promotes the logical combination of the above two schemes [16, 28]. Hence, the cost is significantly reduced than the cumulative cost of signcryption and search.

- Our scheme is proven to achieve selective encryption attribute set secure against chosen-keyword attack and keyword secrecy in the standard model rather than random model [28].

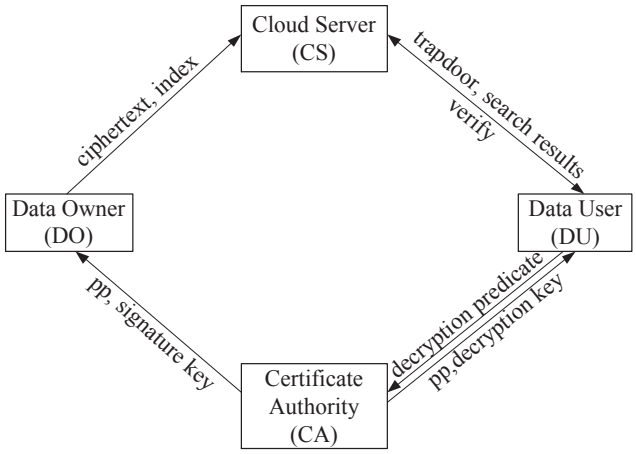

Figure 1: The system framework

2 Generic Framework and Its Security Models

2.1 System Framework

We consider a cryptographic cloud storage system supporting fine-grained access control, data retrieval and data authentication over encrypted data.

As shown in Figure 1, the system framework consists of four entities: Certificate Authority (CA), Data Owners (DO), Data Users(DU), and Cloud Server (CS).

CA: Certificate authority is a global trusted authority. It is responsible for generating the public parameters and the master secret key. Meanwhile, CA is also in charge of generating private/secret keys for participants in the system, such as signing keys for data owners and decryption keys for data users.

DO: Data owners take charge of providing the shared data. They first obtain signing keys from CA according to their roles. Then they select an attribute set to signcrypt their personal data and generate corresponding keyword index. Finally, data owners upload the signcrypted data and the encrypted index to the cloud server.

DU: Data users are some entities who want to access the encrypted data in the cloud server. In order to search data of interest, data users first need to generate a trapdoor and send it to the cloud server. Then, with the help of the cloud server, data users complete the data retrieval. Finally, data users check the validity of the returned results and decrypt the valid ciphertext.

CS: Cloud server is honest-but-curious. It is responsible for storing the ciphertext and index for data owners, and providing the data retrieval service for data users.

2.2 Generic Framework

Let $\mathcal{M}$ be a message space, $\mathcal{G}$ be an access structure space, $\mathcal{U}_s$ be a space of signing attributes, and $\mathcal{U}_e$ be a space of encryption attributes. A generic searchable attribute-based authenticated encryption scheme consists of the following eight algorithms:

- **Setup**$(k)\rightarrow(pp, msk)$: Given a security parameter k, CA creates the public parameters pp and the master secret key msk, where msk is owned by CA.
- **sExtract**$(msk, \mathbb{A}_s)\rightarrow sk_{\mathbb{A}_s}$: Taking as input the master secret key msk and a signing access structure $\mathbb{A}_s$ over a set $W_s \subset \mathcal{U}_s$, CA outputs a signing key $sk_{\mathbb{A}_s}$ to a legitimate data owner over a secret channel.
- **dExtract**$(msk, \mathbb{A}_d)\rightarrow sk_{\mathbb{A}_d}$: On input the master secret key msk and a decryption access structure $\mathbb{A}_d$, CA outputs a decryption key $sk_{\mathbb{A}_d}$ to the user over a secret channel.
- **Signcrypt**$(pp, m, sk_{\mathbb{A}_s}, W_e, W_s) \rightarrow CT$: Data owner performs this algorithm to signcrypt a message $m \in \mathcal{M}$ with the input the public parameters pp, a signing key $sk_{\mathbb{A}_s}$ with the signing attribute set W_s, and an encryption attribute set W_e.
- **GenIndex**$(W_e, pp, kw) \rightarrow I$: Data owner chooses a keyword kw for the message m, then encrypts the keyword kw with the input pp and W_e, then outputs a keyword ciphertext or index I.
- **GenToken**$(sk_{\mathbb{A}_d}, pp, kw') \rightarrow T$: Given an interested keyword kw', data user executes this algorithm with the input $sk_{\mathbb{A}_d}$ and pp to generate a keyword ciphertext or corresponding token T.
- **Search**$(I, T) \rightarrow CT'$: After gaining the token T, CS first matches it with the index I, then returns the relevant search results CT' to data user.
- **Unsigncrypt**$(pp, CT', sk_{\mathbb{A}_d}, W_s) \rightarrow m$: The algorithm first checks the validity of ciphertext CT' with the input signing attribute set W_s and public parameters pp. If CT' can pass the verification, the algorithm continues with the input $sk_{\mathbb{A}_d}$ and returns to data user the message m corresponding to CT'.

2.3 Security Model

The security of the proposed scheme is considered from two aspects. On the one hand, the security of the authenticated encryption or signcryption consists of the indistinguishability of data ciphertext against selective encryption attribute set and chosen-ciphertext attacks and the unforgeability of signature against selective signature attribute set and chosen-message attacks [16].

On the other hand, the security of searchable encryption is composed of the indistinguishability of keyword ciphertext or index against selective encryption attribute set and chosen-keyword attacks, and the secrecy of keyword against chosen-token attacks [28].

1) **Indistinguishability of Data Ciphertext**
We formalize the indistinguishability of data ciphertext against selective encryption attribute set and adaptive chosen-ciphertext attacks by the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$:

Init: The challenger $\mathcal{C}$ gives the space of signature attributes $\mathcal{U}_s$ and the space of encryption attributes $\mathcal{U}_e$. Next, the adversary $\mathcal{A}$ chooses a challenge encryption attribute set W_e^* and send it to $\mathcal{C}$.

Setup: Given a security parameter k, $\mathcal{C}$ runs the **Setup**(k) algorithm to output the public parameters pp, while keeps the master secret key msk to himself.

Phase 1: $\mathcal{A}$ queries the following oracles for polynomial times:

- $\mathcal{O}_{sE}(\mathbb{A}_s)$: On input a signature access structure $\mathbb{A}_s$, $\mathcal{C}$ runs **sExtract**$(msk, \mathbb{A}_s)$ to generate a signing key $sk_{\mathbb{A}_s}$ and sends it to $\mathcal{A}$.
- $\mathcal{O}_{dE}(\mathbb{A}_d)$: On input a decryption access structure $\mathbb{A}_d$ such that $W_e^* \notin \mathbb{A}_d$, $\mathcal{C}$ runs **dExtract**$(msk, \mathbb{A}_d)$ to generate a decryption key $sk_{\mathbb{A}_d}$ for $\mathcal{A}$.
- $\mathcal{O}_{SC}(m, W_e, W_s)$: Given a message m, an encryption attributes set W_e, and a signature attribute set W_s, $\mathcal{C}$ selects a signature access structure $\mathbb{A}_s$ with the restriction $W_s \in \mathbb{A}_s$ and computes $sk_{\mathbb{A}_s}$ through algorithm **sExtract**$(msk, \mathbb{A}_s)$. Finally, $\mathcal{C}$ sends the ciphertext CT generated by running algorithm **Signcryption**$(pp, m, sk_{\mathbb{A}_s}, W_e, W_s)$ to $\mathcal{A}$.
- $\mathcal{O}_{US}(pp, CT, sk_{\mathbb{A}_d}, W_s)$: Taking as input ciphertext CT and decryption access structure $\mathbb{A}_d$, $\mathcal{C}$ runs **dExtract**$(msk, \mathbb{A}_d)$ to obtain the decryption key $sk_{\mathbb{A}_d}$ and runs **Unsigncrypt**$(pp, CT, sk_{\mathbb{A}_d}, W_s)$ to output the message m.

Challenge: $\mathcal{A}$ sends two equal length messages m_0, m_1 and a signature attribute set W_s to $\mathcal{C}$.

Then, $\mathcal{C}$ chooses a signature access structure $\mathbb{A}_s$ such that $W_s \in \mathbb{A}_s$ and runs the algorithm **sExtract**$(msk, \mathbb{A}_s)$ to generate the signing key $sk_{\mathbb{A}_s}$. Finally, $\mathcal{C}$ selects $b \in \{0,1\}$ and runs **Signcrypt**$(pp, m_b, sk_{\mathbb{A}_s}, W_e, W_s)$ to output the ciphertext CT_b.

Phase 2: $\mathcal{A}$ continues to query the oracles as in Phase 1. The restriction is that $W_e^* \notin \mathbb{A}_d$ for any $\mathbb{A}_d$ in $\mathcal{O}_{US}(pp, CT, sk_{\mathbb{A}_d}, W_s)$.

Guess: $\mathcal{A}$ returns a guess $b' \in \{0,1\}$. It wins the game if $b' = b$.

The advantage of $\mathcal{A}$ in the above game is defined as

$$Adv\left(1^k\right) = \left|\Pr\left[b = b'\right] - \frac{1}{2}\right|$$

Definition 1. *A searchable attribute-based authenticated encryption scheme can achieve the ciphertext indistinguishability under selective encryption attribute set and adaptive chosen-ciphertext attacks if an adversary $\mathcal{A}$ wins the above game with a negligible advantage.*

2) **Unforgeability of Signature**

We formalize the unforgeability of signature against selective signature attribute set and adaptive chosen-message attacks by the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

Init: The challenger $\mathcal{C}$ gives the space of signature attributes $\mathcal{U}_s$ and the space of encryption attributes $\mathcal{U}_e$. Then, the adversary $\mathcal{A}$ chooses a signature attribute set W_s^* and send it to $\mathcal{C}$.

Setup: Given a security parameter k, $\mathcal{C}$ runs the algorithm **Setup**(k) to output the public parameter pp and keeps the master secret key msk to himself.

Phase: $\mathcal{A}$ can query the following oracles for polynomial times:

- $\mathcal{O}_{sE}(\mathbb{A}_s)$: Given a signature access structure $\mathbb{A}_s$ such that $W_s^* \notin \mathbb{A}_s$, $\mathcal{C}$ computes $sk_{\mathbb{A}_s}$ by running **sExtract**$(msk, \mathbb{A}_s)$ and sends it to $\mathcal{A}$.
- $\mathcal{O}_{dE}(\mathbb{A}_d)$: Taking as input a decryption access structure $\mathbb{A}_d$, $\mathcal{C}$ runs the algorithm **dExtract**$(msk, \mathbb{A}_d)$ to output a decryption key $sk_{\mathbb{A}_d}$.
- $\mathcal{O}_{SC}(m, W_e, W_s)$: Given a message m, an encryption attribute set W_e and a signature attribute set W_s, $\mathcal{C}$ selects a signature access structure $\mathbb{A}_s$ with the restriction $W_s \in \mathbb{A}_s$, obtains $sk_{\mathbb{A}_s}$ by the oracle $\mathcal{O}_{sE}(\mathbb{A}_s)$, and runs **Signcrypt**$(m, pp, sk_{\mathbb{A}_s}, W_e)$ to generate CT for $\mathcal{A}$.
- $\mathcal{O}_{US}(CT, \mathbb{A}_d)$: On input ciphertext CT and decryption access structure $\mathbb{A}_d$, $\mathcal{C}$ obtains a decryption key $sk_{\mathbb{A}_d}$ by the oracle $\mathcal{O}_{dE}(\mathbb{A}_d)$ and runs the algorithm **Unsigncrypt**$(pp, CT, sk_{\mathbb{A}_d}, W_s)$ to output the message m.

Forgery: $\mathcal{A}$ sends a forgery $CT^*_{(W_s^*, W_e)}$ of message m^* and a decryption access structure $\mathbb{A}_d$. If the ciphertext $CT^*_{(W_s^*, W_e)}$ is valid and cannot be gained from $\mathcal{O}_{SC}(m, W_e, W_s)$, $\mathcal{A}$ wins the game.

The advantage of $\mathcal{A}$ in the above game is defined as

$$Adv_{\mathcal{A}} = \Pr[\mathcal{A} \ \ wins].$$

Definition 2. *A searchable attribute-based authenticated encryption scheme can achieve the unforgeability of signature against selective signature attribute set and adaptive chosen-message attacks if adversary $\mathcal{A}$ wins the above game with a negligible advantage.*

3) **Indistinguishability of Keyword Ciphertext**

We formalize the indistinguishability of keyword ciphertext or index against selective encryption attribute set and adaptive chosen-keyword attacks by the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

Init: $\mathcal{C}$ gives the space of encryption attributes $\mathcal{U}_e$ and $\mathcal{A}$ selects a challenge encryption attribute set W_e^* and sends it to $\mathcal{C}$.

Setup: Given a security parameter k, $\mathcal{C}$ runs algorithm **Setup**(k) to output the public parameters pp, while the master secret key msk is owned by himself.

Phase 1: $\mathcal{A}$ queries the following oracles for polynomial times. Meanwhile, $\mathcal{C}$ maintains a keyword list L_{kw}, which is initially empty.

- $\mathcal{O}_{dE}(\mathbb{A}_d)$: Taking as input a decryption access structure $\mathbb{A}_d$ such that $W_e^* \notin \mathbb{A}_d$, $\mathcal{C}$ runs **dExtract**$(msk, \mathbb{A}_d)$ to output a decryption key $sk_{\mathbb{A}_d}$. Otherwise, $\mathcal{C}$ outputs $\perp$.
- $\mathcal{O}_{GT}(\mathbb{A}_d, kw)$: $\mathcal{C}$ first runs $\mathcal{O}_{dE}(\mathbb{A}_d)$ to output a decryption key $sk_{\mathbb{A}_d}$ and runs **GenToken**$(sk_{\mathbb{A}_d}, kw)$ to return to $\mathcal{A}$ a token T. If the attribute set W_e^* satisfies the access structure $\mathbb{A}_d$, $\mathcal{C}$ adds kw to L_{kw}.

Challenge: $\mathcal{A}$ sends two equal length keywords kw_0 and kw_1 such that $kw_0, kw_1 \notin L_{kw}$ to $\mathcal{C}$. Then, $\mathcal{C}$ randomly picks a bit $b \in \{0,1\}$ and runs the algorithm **GenIndex**(W_e^*, kw_b) to generate I for $\mathcal{A}$.

Phase 2: $\mathcal{A}$ issues a series of queries as in Phase 1. The restriction is that if $W_e^* \in \mathbb{A}_d$, $\mathcal{A}$ cannot query $\mathcal{O}_{GT}$ with $(\mathbb{A}_d, kw_0)$ or $(\mathbb{A}_d, kw_1)$.

Guess: $\mathcal{A}$ outputs a guess $b' \in \{0,1\}$. If $b' = b$, $\mathcal{A}$ wins the game.

The advantage of $\mathcal{A}$ in breaking the above scheme is defined as

$$Adv\left(1^k\right) = \left| Pr\left[b = b'\right] - \frac{1}{2} \right|$$

Definition 3. *A searchable attribute-based authenticated encryption scheme can achieve the indistinguishability of keyword ciphertext or index against selective encryption attribute set and adaptive chosen-keyword attacks if adversary $\mathcal{A}$ wins the above game with a negligible advantage.*

4) **Keyword Secrecy**

Finally, we formalize the secrecy of keyword against adaptive chosen-token attacks by the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

Setup: Given a security parameter k, $\mathcal{C}$ runs the algorithm **Setup**(k) to generate the public parameters pp and master secret key msk.

Phase: $\mathcal{A}$ issues the following queries for $|q|$ times.

- $\mathcal{O}_{dE}(\mathbb{A}_d, msk)$: On input a decryption access structure $\mathbb{A}_d$, $\mathcal{C}$ returns to $\mathcal{A}$ a corresponding secret key $sk_{\mathbb{A}_d}$ and adds $\mathbb{A}_d$ to the list $L_{\mathbb{A}_d}$, which is initially empty.
- $\mathcal{O}_{GT}(\mathbb{A}_d, kw)$: Given a decryption access structure $\mathbb{A}_d$, $\mathcal{C}$ generates a secret key $sk_{\mathbb{A}_d}$ by running $\mathcal{O}_{dE}(\mathbb{A}_d, msk)$ and returns to $\mathcal{A}$ a token T by the algorithm **GenToken**$(sk_{\mathbb{A}_d}, kw)$.

Challenge: $\mathcal{A}$ submits a challenge W_e^* to $\mathcal{C}$. Then, $\mathcal{C}$ selects kw^* and $\mathbb{A}_d^*$ such that $W_e^* \in \mathbb{A}_d^*$. $\mathcal{C}$ runs **GenIndex**(pp, W_e^*, kw^*) and **GenToken**$(sk_{\mathbb{A}_d}^*, kw^*)$ to return $\mathcal{A}$ index I^* and token T^*. Note that $\forall \mathbb{A}_d \in L_{\mathbb{A}_d}, W_e^* \notin \mathbb{A}_d$.

Guess: After issuing q distinct keywords, $\mathcal{A}$ outputs a keyword kw' and wins the keyword secrecy game if $kw' = kw^*$.

Definition 4. *A searchable attribute-based authenticated encryption scheme can achieve the secrecy of keyword against adaptive chosen-token attacks if the advantage of $\mathcal{A}$ in breaking the above keyword secrecy game is at most $\frac{1}{|\mathcal{M}|-|q|} + \varepsilon$, where $|q|$ denotes the number of queried keywords, ε is a negligible probability in security parameter l, and $\mathcal{M}$ is the message space.*

3 Our Concrete Construction

By using Rao *et al.*'s key-policy attribute-based signcryption scheme [16] and modifying Zheng *et al.*'s attribute-based keyword search method [28], we construct a concrete scheme for searchable attribute-based authenticated encryption as follows:

3.1 The Proposed Scheme

Setup: Given the secure parameter k, CA outputs two cyclic groups $\mathbb{G}_1, \mathbb{G}_2$ of an prime order p, a generator g of $\mathbb{G}_1$, and a map $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$. Let $\mathcal{U}_e = \{att_j\}$ and $\mathcal{U}_s = \{att'_j\}$ be the set of encryption and signature attributes, respectively. CA chooses four one-way, collision-resistant hash functions $H_1 : \mathbb{G}_2 \times \mathbb{G}_1 \times \{0,1\}^{l_\tau} \to \{0,1\}^*$, $H_2 : \{0,1\}^* \to \{0,1\}^l$, $H_3 : \mathbb{G}_1 \to \mathbb{Z}_p$ and $H_4 : \{0,1\}^* \to \mathbb{Z}_p$, where l is large enough so that the hash functions are collision resistant and $l_\tau \approx 40$. CA randomly chooses $a, b, c \leftarrow \mathbb{Z}_p$, $T_0, K_0, \delta_1, \delta_2, y_0, y_1, \cdots, y_l \in \mathbb{G}_1$, $T_j \in \mathbb{G}_1$ for each signature attribute $att'_j \in \mathcal{U}_s$, $K_j \in \mathbb{G}_1$ for each encryption attribute $att_j \in \mathcal{U}_e$, and sets public parameters pp and master secret key msk as follows:

$$\begin{aligned} pp &= \left\{ \begin{array}{l} e(g,g)^{ac}, g^a, g^b, g^c, \delta_1, \delta_2, \{H_i\}_{i=1}^{i=4}, \{y_i\}_{i\in[l]}, \\ T_0, K_0, y_0, \{T_j \,|att'_j \in \mathcal{U}_s\}, \{K_j \,|att_j \in \mathcal{U}_e\} \end{array} \right\}, \\ msk &= \{a, b, c\} \end{aligned}$$

sEtract: On input a signature predicate(S, ρ), where S is an $l_s \times n_s$ matrix and i-th row (i.e $\vec{S}_i$) is associated with an attribute $att'_{\rho(i)}$. Then, CA chooses a random vector $\vec{v}_s = (ac, v_2, \cdots, v_{n_s}) \in \mathbb{Z}_p^{n_s}$ and sets $\left\{\lambda_{\rho(i)} = \vec{S}_i \cdot \vec{v}_s | i \in [l_s]\right\}$. For each row $i \in [l_s]$, CA selects $r_i \in \mathbb{Z}_p$ at random and calculates

$$\begin{aligned} D_{s,i} &= g^{\lambda_{\rho(i)}} \left(T_0 T_{\rho(i)}\right)^{r_i} \\ D'_{s,i} &= g^{r_i} \\ D''_{s,i} &= \left\{D''_{s,i,j} = T_j^{r_i}, \forall att'_j \in \mathcal{U}_s \backslash att'_{\rho(i)}\right\} \end{aligned}$$

Finally, the signature key is set as:

$$sk_{(S,\rho)} = \left\{(S,\rho), \left\{D_{s,i}, D'_{s,i}, D''_{s,i}\right\}_{i\in[l_s]}\right\}.$$

dExtract: Take as input a decryption predicate (D, φ), where D is an $l_e \times n_e$ matrix and i-th row (i.e $\vec{D}_i$) is associated with an attribute $att_{\varphi(i)}$. Then, CA chooses a random vector $\vec{v}_e = \left(ac, v'_2, \cdots, v'_{n_e}\right) \in \mathbb{Z}_p^{n_e}$ and sets $\left\{\lambda_{\varphi(i)} = \vec{D}_i \cdot \vec{v}_e \,| i \in [l_e]\right\}$. For each row $i \in [l_e]$, CA selects $\tau_i \in \mathbb{Z}_p$ and computes

$$\begin{aligned} D_{e,i} &= g^{\lambda_{\varphi(i)}} \left(K_0 K_{\varphi(i)}\right)^{\tau_i} \\ D'_{e,i} &= g^{\tau_i} \\ D''_{e,i} &= \left\{D''_{e,i,j} = K_j^{r_i}, \forall att_j \in \mathcal{U}_e \backslash att_{\varphi(i)}\right\} \end{aligned}$$

Finally, CA sets the decryption key as follows:

$$sk_{(D,\varphi)} = \left\{(D,\varphi), \left\{D_{e,i}, D'_{e,i}, D''_{e,i}\right\}_{i\in[l_e]}\right\}.$$

Signcrypt: For each legitimate data owner, he holds an authorized signature attribute set W_s, which satisfies the signature predicate (S, ρ). Therefore, data owner can find a coefficient set

$\{w_i : i \in I_s\}$ such that $\sum_{i \in I_s} w_i \lambda_{\rho(i)} = ac$, where $I_s = \left\{ i \in [l_s] \middle| att'_{\rho(i)} \in W_s \right\}$. Note that the secret shares $\{\lambda_{\varphi(i)}\}_{i \in [l_s]}$ and the secret value ac are not explicitly known to the data owner [16]. To signcrypt a message $m \in \{0,1\}^{l_m}$, data owner chooses an encryption attribute set W_e which describes the target users. Then, he randomly selects $\theta, \vartheta \in \mathbb{Z}_p$ and calculates

$$\begin{aligned} C_1 &= g^\theta, C_2 = \left(\prod_{att_j \in W_e} K_0 K_j \right)^\theta, \sigma_1 = \left(g^\theta\right)^\vartheta, \\ C_3 &= H_1\left(e(g,g)^{ac\theta}, \sigma_1, \tau\right) \oplus m, \mu = H_3\left(C_1\right) \end{aligned}$$

where $\tau \in \{0,1\}^{l_\tau}$. Next, he picks $\xi \in \mathbb{Z}_p$ at random, computes

$$\begin{aligned} \sigma_2 &= g^\xi \prod_{i \in I_s} \left(D'_{s,i}\right)^{w_i}, C_4 = \left(\delta_1^\mu \delta_2\right)^\theta \\ (k_1, \cdots, k_l) &= H_2\left(\sigma_2 || \tau || W_s || W_e\right) \\ \beta &= H_4\left(\sigma_1 || C_2 || C_3 || C_4 || W_s || W_e\right) \\ \sigma_3 &= \prod_{i \in I_s} \left(D_{s,i} \cdot \prod_{att'_j \in W_s, j \neq \rho(i)} D''_{s,i,j} \right)^{w_i} \\ &\quad \cdot \left(T_0 \prod_{att'_j \in W_s} T_j \right)^\xi \cdot \left(y_0 \prod_{i \in [l]} y_i^{k_i} \right)^\theta \cdot C_4^{\beta\vartheta} \end{aligned}$$

and sets $\sigma_4 = \tau$. Finally, the signcryption of m is set as:

$$CT = \{W_s, W_e, C_1, C_2, C_3, C_4, \sigma_1, \sigma_2, \sigma_3, \sigma_4\}$$

GenIndex: In order to quickly search the encrypted data when needed, data owner extracts and encrypts a keyword kw before outsourcing CT to the cloud sever. he first picks $r_1, r_2 \in \mathbb{Z}_p$ and computes

$$W_0 = g^{cr_1}, W_1 = g^{a(r_1+r_2)} g^{br_1 H_4(kw)}, W_2 = g^{r_2}$$

For each $att_j \in W_e$, he sets $W_j = (K_0 K_j)^{r_2}$. Finally, data owner uploads a index I of keyword kw and the signcryption CT of message m to CS, where

$$I = \left\{ W_0, W_1, W_2, \{W_j\}_{att_j \in W_e} \right\}$$

GenToken: Data user inputs an interested keyword kw' and runs algorithm **GenToken**$(sk_{(D,\varphi)}, kw')$ to generate a token T. He first selects a random element $\beta \in \mathbb{Z}_p$ and computes

$$tk_1 = \left(g^a g^{bH_4(kw')} \right)^\beta, tk_2 = g^{c\beta}$$

For each $i \in [l_e]$, calculate

$$D'_i = \left(D_{e,i}\right)^\beta, K'_i = \left(D'_{e,i}\right)^\beta$$

A token of keyword kw' is set as:

$$T = \left\{ \left(D, \varphi\right), tk_1, tk_2, \left\{ \left(D'_i, K'_i\right) | i \in [l_e] \right\} \right\}$$

Finally, he sends the token T of keyword kw' to CS.

Search: After verifying that the attribute set W_e satisfies the decryption predicate (D, φ), CS executes algorithm **Search**(I, T) to return the relevant ciphertext CT'. If the attribute set W_e satisfies the decryption predicate $\mathbb{D}$, CS can find a coefficient set $\{w'_i | i \in I_e\}$ such that $\sum_{i \in I_e} w'_i \vec{D}_i = (1, 0, \cdots, 0)$, where $I_e = \left\{ i \in [l_e] \middle| att_{\varphi(i)} \in W_e \right\}$. And thus, $\sum_{i \in I_e} w'_i \lambda_{\varphi(i)} = ac$. Then, CS computes

$$E = \prod_{i \in I_e} \left(\frac{e\left(D'_i, W_2\right)}{e\left(K'_i, W_j\right)} \right)^{w'_i} = e(g,g)^{ac\beta r_2}$$

and checks whether the following Equation (1) holds or not.

$$e\left(W_0, tk_1\right) \cdot E = e\left(W_1, tk_2\right) \tag{1}$$

If the equation holds, CS returns the relevant ciphertext CT' to data user. Otherwise, output $\perp$.

Unsigncrypt: After obtaining the search results CT', data user first computes

$$\begin{aligned} \mu &= H_3\left(C_1\right), \\ (k_1, \cdots, k_l) &= H_2\left(\sigma_2 || \sigma_4 || W_s || W_e\right), \\ \beta &= H_4\left(\sigma_1 || C_2 || C_3 || C_4 || W_s || W_e\right) \end{aligned}$$

and checks the validity of CT' according to the following Equation (2).

$$\begin{aligned} e\left(\sigma_3, g\right) &= e(g,g)^{ac} \cdot e\left(T_0 \prod_{att'_j \in W_s} T_j, \sigma_2 \right) \\ &\quad \cdot e\left(y_0 \prod_{i \in [l]} y_i^{k_i}, C_1 \right) \cdot e\left(\left(\delta_1^\mu \delta_2\right)^\beta, \sigma_1 \right) \end{aligned} \tag{2}$$

If Equation (2) does not hold, output $\perp$. Otherwise, data user decrypt CT' as follows.

1) Reconstruct a set of coefficient $\{v_i : i \in I_e\}$ such that $\sum_{i \in I_e} \lambda_{\varphi(i)} v_i = ac$.
2) Set

$$\begin{aligned} E_1 &= \prod_{i \in I_e} \left(D_{e,i} \prod_{att_j \in W_e} D''_{e,i,j} \right)^{v_i} \\ E_2 &= \prod_{i \in I_e} \left(D'_{e,i}\right)^{v_i} \end{aligned}$$

3) Compute

$$e(g,g)^{ac\theta} = \frac{e\left(C_1, E_1\right)}{e\left(C_2, E_2\right)} \tag{3}$$

4) Recover $m = C_3 \oplus H_1\left(e(g,g)^{ac\theta}, \sigma_1, \sigma_4\right)$.

3.2 Correctness

In this section, we illustrate the correctness of the above equations.

3.2.1 Correctness of Equation(1)

W_e satisfies (D, φ), so there is $\sum_{i \in I_e} w_i' \lambda_{\varphi(i)} = ac$. Then,

$$E = \prod_{i \in I_e} \left(\frac{e\left(D_i', W_2\right)}{e\left(K_i', W_j\right)} \right)^{w_i'} = e(g,g)^{ac\beta r_2}$$

$$e\left(W_0, tk_1\right) = e(g,g)^{ac\beta r_1} \cdot e(g,g)^{bc\beta r_1 H_4\left(kw'\right)}$$

$$e\left(W_1, tk_2\right) = e(g,g)^{ac\beta (r_1 + r_2)} \cdot e(g,g)^{bc\beta r_1 H_4(kw)}$$

Therefore, we have $e\left(W_0, tk_1\right) \cdot E = e\left(W_1, tk_2\right)$.

3.2.2 Correctness of Equation(2)

When W_s satisfies (S, ρ), there exists $\sum_{i \in I_s} \lambda_{\rho(i)} w_i = ac$. Then,

$$\begin{aligned} & \prod_{i \in I_s} \left(D_{s,i} \prod_{att_j' \in W_s, j \neq \rho(i)} D''_{s,i,j} \right)^{w_i} \\ = & \prod_{i \in I_s} \left(g^{\lambda_{\rho(i)}} \left(T_0 T_{\rho(i)}\right)^{r_i} \prod_{att_j' \in W_s, j \neq \rho(i)} T_j^{r_j} \right)^{w_i} \\ = & g^{ac} \left(T_0 \prod_{att_j' \in W_s} T_j \right)^{\sum_{i \in I_s} r_i w_i} \end{aligned}$$

So,

$$\begin{aligned} e\left(\sigma_3, g\right) = & e(g,g)^{ac} \cdot e\left(T_0 \prod_{att_j' \in W_s} T_j, \sigma_2 \right) \\ & \cdot e\left(y_0 \prod_{j \in [l]} y_i^{k_i}, C_1 \right) \cdot e\left(\left(\delta_1^{\mu} \delta_2\right)^{\beta}, \sigma_1 \right) \end{aligned}$$

3.2.3 Correctness of Equation(3)

Since W_e satisfies (D, φ), we have $\sum_{i \in I_e} \lambda_{\varphi(i)} v_i = ac$. Next,

$$\begin{aligned} E_1 = & \prod_{i \in I_e} \left(g^{\lambda_{\varphi(i)}} \left(K_0 K_{\varphi(i)}\right)^{\tau_i} \cdot \prod_{att_j \in W_e, j \neq \varphi(i)} K_j^{\tau_j} \right)^{v_i} \\ = & g^{ac} \left(K_0 \prod_{att_j \in W_e} K_j^{\tau_j} \right)^{\sum_{i \in I_e} \tau_i v_i} \\ E_2 = & \prod_{i \in I_e} \left(D'_{e,i}\right)^{v_i} = \prod_{i \in I_e} \left(g^{\tau_i}\right)^{v_i} = g^{\sum_{i \in I_e} \tau_i v_i} \end{aligned}$$

Hence,

$$\begin{aligned} \frac{e\left(C_1, E_1\right)}{e\left(C_2, E_2\right)} = & \frac{e\left(g^{\theta}, g^{ac} \left(K_0 \prod_{att_j \in W_e} K_j^{\tau_j} \right)^{\sum_{i \in I_e} \tau_i v_i} \right)}{e\left(\left(K_0 \prod_{att_j \in W_e} K_j \right)^{\theta}, g^{\sum_{i \in I_e} \tau_i v_i} \right)} \\ = & e(g,g)^{ac\theta} \end{aligned}$$

Therefore, we prove that our scheme is correct.

4 Security Proof

Based on Rao *et al.*'s scheme [16] and Zheng *et al.*'s scheme [28], the security of the proposed scheme can be guaranteed through the following four theorems. Due to space constraints, these proofs will be shown in **Appendix A-D**.

Theorem 1. *Our scheme can achieve the indistinguishability of data ciphertext under selective encryption attribute set and adaptive chosen-ciphertext attacks based on the hardness assumption of n-DBDHE problem without using any random oracle.*

Theorem 2. *Our scheme is unforgeable under selective signature attribute set and adaptive chosen-message attacks based on the hardness assumption of the n-CDHE problem without using the random oracle.*

Theorem 3. *Our scheme can achieve the indistinguishability of keyword ciphertext or index under selective encryption attribute set and chosen-keyword attacks based on the hardness assumption of DL problem in the standard model.*

Theorem 4. *Given the given one-way hash function H_4, our scheme can guarantee the secrecy of keyword against adaptive chosen-token attacks.*

4.1 Efficiency Analysis

Table 1 shows the computation cost and size in different phases. For convenience, we use the following notations.

- e: Time cost of an exponentiation;
- p: Time cost of a bilinear pairing;
- $|G_1|(|G_2|)$: Size of a group element in group $\mathbb{G}_1(\mathbb{G}_2)$;
- $u_s(u_e)$: Number of signature(decryption) attributes in $\mathcal{U}_s(\mathcal{U}_e)$;
- $l_s(l_e)$: Number of signature(decryption) attributes in $(S, \rho)((D, \varphi))$;
- ϕ_s: Number of signature attributes required in the signcryption;

- ϕ_e: Number of decryption attributes required in the unsigncryption;
- l: Minimum value that the hash functions are collision resistant.

Table 1: Efficiency analysis of the proposed scheme

Algorithm	Computation Cost	Size
Setup	-	$(u_s + u_e + l)\,\|G_1\| + \|G_2\|$
Signing key	$u_s l_s \cdot e$	$u_s l_s$
Decryption key	$u_e l_e \cdot e$	$u_e l_e$
Signcryption	$(\phi_s + 10) \cdot e$	6
Index	$(\phi_e + 4) \cdot e$	$\phi_e + 3$
Token	$(2l_e + 2) \cdot e$	$2l_e + 2$
Search	$(2\phi_e + 2) \cdot p + \phi_e \cdot e$	-
Unsigncryption	$6p + 2\phi_e \cdot e$	-

In the system **Setup** phase, CA takes charge of generating public parameters, whose size is $u_s + u_e + l$ group elements in group $\mathbb{G}_1$ and one group element in group $\mathbb{G}_2$. When a data owner wants to upload join the system, he needs to require the CA to generate the signing key, which needs $u_s l_s$ exponentiation operations to output $u_s l_s$ group elements in group $\mathbb{G}_1$. Similar to the singing key, decryption key contains $u_e l_e$ group elements in group $\mathbb{G}_1$ and takes $u_e l_e$ exponentiation operations. Before upload some data, the data owner needs to encrypt the data and its key. At the **Signcrypt** phase, data ciphertext only contains 6 group elements in group $\mathbb{G}_1$, which needs $(\phi_s + 10)$ exponentiation operations. In addition, in the **GenIndex** phase, the size of the keyword ciphertext or index is $\phi_e + 3$ group elements in group $\mathbb{G}_1$, whose generation needs $(\phi_e + 4)$ exponentiation operations. With the decryption key, a data user can run the **GenToken** algorithm to generate the token, which needs $(2l_e + 2)$ exponentiation operations to generate $2l_e + 2$ group elements in group $\mathbb{G}_1$. To search over an index, the main computation cost is $2\phi_e + 2$ bilinear pairing operations and ϕ_e exponentiation operations. In the **Unsigncryption** phase, there exits 6 bilinear pairing operations and $2\phi_e$ exponentiation operations.

5 Conclusions

In this paper, we have proposed a new concept of searchable attribute-based authenticated encryption and constructed a concrete scheme. The proposed scheme is more appealing on account of its supporting for expressive fine-grained access control, data retrieval and authentication. In our scheme, data owners are allowed to share their data with flexible access policy and authorize legitimate data users to issue search queries according to their token. To avoid receiving illegal data, the users can check the validity of the ciphertext. Furthermore, security proof shows that the proposed scheme is selectively secure in the standard model and has keyword secrecy.

Acknowledgment

This work is supported by the National Key R&D Program of China under Grant No. 2017YFB0802000, the National Natural Science Foundation of China under Grants No.61472470 and 61807026, and the Scientific Research Plan Project of Education Department of Shaanxi Province under Grant No.17JK0362.

References

[1] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *IEEE Symposium on Security and Privacy (SP'07)*, pp. 321–334, 2007.

[2] D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in *Eurocrypt*, vol. 3027, pp. 506–522, 2004.

[3] Z. Cao, L. Liu, Z. Guo, "Ruminations on attribute-based encryption," *International Journal of Electronics and Information Engineering*, vol. 8, no. 1, pp. 9–19, 2018.

[4] P. S. Chung, C. W. Liu, and M. S. Hwang, "A study of attribute-based proxy re-encryption scheme in cloud environments", *International Journal of Network Security*, vol. 16, no. 1, pp. 1-13, 2014.

[5] Q. Dong, Z. Guan, and Z. Chen, "Attribute-based keyword search efficiency enhancement via an online/offline approach," in *IEEE 21st International Conference on Parallel and Distributed Systems (ICPADS'15)*, pp. 298–305, 2015.

[6] M. Gagné, S. Narayan, and R. Safavi-Naini, "Threshold attribute-based signcryption," in *SCN*, vol. 6280, pp. 154–171, 2010.

[7] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM Conference on Computer and Communications Security*, pp. 89–98, 2006.

[8] C. Gu and Y. Zhu, "New efficient searchable encryption schemes from bilinear pairings," *International Journal Network Security*, vol. 10, no. 1, pp. 25–31, 2010.

[9] S. T. Hsu, C. C. Yang, and M. S. Hwang, "A study of public key encryption with keyword search," *International Journal Network Security*, vol. 15, no. 2, pp. 71–79, 2013.

[10] F. G. Jeng, S. Y. Lin, B. J. Wang, C. H. Wang, and T. H. Chen, "On the security of privacy-preserving keyword searching for cloud storage services," *International Journal Network Security*, vol. 18, no. 3, pp. 597–600, 2016.

[11] J. Lai, R. H. Deng, C. Guan, and J. Weng, "Attribute-based encryption with verifiable outsourced decryption," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 8, pp. 1343–1354, 2013.

[12] C. C. Lee, P. S. Chung, and M. S. Hwang, "A survey on attribute-based encryption schemes of access control in cloud environments," *International Journal Network Security*, vol. 15, no. 4, pp. 231–240, 2013.
[13] J. Li and L. Zhang, "Attribute-based keyword search and data access control in cloud," in *Tenth International Conference on Computational Intelligence and Security (CIS'14)*, pp. 382–386, 2014.
[14] L. Liu, Z. Cao, C. Mao, "A note on one outsourcing scheme for big data access control in cloud," *International Journal of Electronics and Information Engineering*, vol. 9, no. 1, pp. 29–35, 2018.
[15] X. Meng and X. Meng, "A novel attribute-based signcryption scheme in cloud computing environments," in *IEEE International Conference on Information and Automation (ICIA'16)*, pp. 1976–1979, 2016.
[16] Y. S. Rao and R. Dutta, "Efficient attribute-based signature and signcryption realizing expressive access structures," *International Journal of Information Security*, vol. 15, no. 1, pp. 81–109, 2016.
[17] A. Sahai, B. Waters, *et al.*, "Fuzzy identity-based encryption," in *Eurocrypt*, vol. 3494, pp. 457–473, 2005.
[18] S. Selvi, S. Vivek, D. Vinayagamurthy, *et al.*, "ID-based signcryption scheme in standard model," in *International Conference on Provable Security (ProvSec'12)*, vol. 7496, pp. 35–52, 2012.
[19] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *Proceedings of IEEE Symposium on Security and Privacy*, pp. 44–55, 2000.
[20] W. Sun, S. Yu, W. Lou, Y. T. Hou, and H. Li, "Protecting your right: Verifiable attribute-based keyword search with fine-grained owner-enforced search authorization in the cloud," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 4, pp. 1187–1198, 2016.
[21] B. Wang, S. Yu, W. Lou, and Y. T. Hou, "Privacy-preserving multi-keyword fuzzy search over encrypted data in the cloud," in *Proceedings IEEE*, pp. 2112–2120, 2014.
[22] C. Wang, W. Li, Y. Li, and X. L. Xu, "A ciphertext-policy attribute-based encryption scheme supporting keyword search function," in *CSS*, pp. 377–386, 2013.
[23] Y. Wang, "Lattice ciphertext policy attribute-based encryption in the standard model," *International Journal Network Security*, vol. 16, no. 6, pp. 444–451, 2014.
[24] B. Waters, "Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization," in *Public Key Cryptography*, vol. 6571, pp. 53–70, 2011.
[25] J. Wei, X. Hu, and W. Liu, "Traceable attribute-based signcryption," *Security and Communication Networks*, vol. 7, no. 12, pp. 2302–2317, 2014.
[26] H. Xiong, J. Geng, Z. Qin, and G. Zhu, "Cryptanalysis of attribute-based ring signcryption scheme," *International Journal Network Security*, vol. 17, no. 2, pp. 224–228, 2015.
[27] J. Ye, J. Wang, J. Zhao, J. Shen, and K. C. Li, "Fine-grained searchable encryption in multi-user setting," *Soft Computing*, pp. 1–12, 2016.
[28] Q. Zheng, S. Xu, and G. Ateniese, "Vabks: Verifiable attribute-based keyword search over outsourced encrypted data," in *Proceedings IEEE (INFOCOM'14)*, pp. 522–530, 2014.
[29] Y. Zheng, "Digital signcryption or how to achieve cost (signature & encryption) ¡¡ cost (signature)+ cost (encryption)," *Advances in Cryptology (CRYPTO'97)*, pp. 165–179, 1997.

Appendix A

Proof of Theorem 1

Proof. In this phase, the encryption attribute space $\mathcal{U}_e$ is considered to have n attributes and the hash functions $\{H_i\}_{i=1}^4$ are collision resistant. Assume that the simulator $\mathcal{C}$ has a n-DBDHE instance $(\vec{y}_{a,\theta}, Z)$, where

$$\vec{y}_{a,\theta} = \left(g, g^{\theta}, \{g_i\}_{i=1,\cdots n,n+2,\cdots,2n}\right), g_i = g^{a^i}, a, \theta \in \mathbb{Z}_p.$$

Then, $\mathcal{C}$ attempts to distinguish Z is $e\left(g_{n+1}, g^{\theta}\right)$ or a random element of $\mathbb{G}_2$ through the following game. In addition, $\mathcal{C}$ plays the role of challenger in the game of the security model and interact with adversary $\mathcal{A}$.

Init: $\mathcal{C}$ gives the space of signature attributes $\mathcal{U}_s = \{att'_j\}$, the space of encryption attributes $\mathcal{U}_e = \{att_1, \cdots, att_n\}$ and the message space $\mathcal{M} = \{0,1\}^{l_m}$. Then, the adversary $\mathcal{A}$ chooses a challenged encryption attribute set $W_e^* \subset \mathcal{U}_e$ and send it to $\mathcal{C}$.

Setup: $\mathcal{C}$ generates public parameters for $\mathcal{A}$ as follows:

1) Select $\alpha' \in \mathbb{Z}_p$ and set $e(g,g)^{ac} = e(g,g)^{\alpha'} \cdot e\left(g_1, g_n\right)$ by implicitly setting $ac = \alpha' + a^{n+1}$.
2) For each $att_j \in \mathcal{U}_e$, select $\gamma_j \in \mathbb{Z}_p$ and set $K_j = g^{\gamma_j} g_{n+1-j}$. In addition, $K_0 = g^{\gamma_0} \prod_{att_j \in W_e^*} K_j^{-1}$, where $\gamma_0 \in \mathbb{Z}_p$.
3) Choose $t_0 \in \mathbb{Z}_p$, sets $T_0 = g_1 g^{t_0}$, pick $t_j \in \mathbb{Z}_p$, and compute $T_j = g^{t_j}$ for each $att'_j \in \mathcal{U}_s$.
4) Let $C_1^* = g^{\theta}, \mu^* = H_3\left(C_1^*\right)$, and compute $\delta_1 = g_n^{1/\mu^*}, \delta_2 = g_n^{-1} g^d$, where $d \in \mathbb{Z}_p$.
5) Pick $x_0, \cdots, x_l \in \mathbb{Z}_p$ and compute $u_0 = g^{x_0}, \cdots, u_l = g^{x_l}$.

Phase 1: $\mathcal{A}$ queries the following oracles for polynomially times.

- $\mathcal{O}_{sE}(S, \rho)$: Given a signature LSSS access structure (S, ρ), $\mathcal{C}$ generates a signature key $sk_{(S,\rho)}$ for $\mathcal{A}$ as follows:
  1) Choose α' at random and set $ac = \alpha' + a^{n+1}$.

2) Let $S = (S_{i,j})_{l_s \times k_s}$, where $\vec{S}_i = (S_{i,1}, \cdots, S_{i,k_s})$ is the i-th row of S.
3) Pick $v_2, \cdots, v_{k_s} \in \mathbb{Z}_p$ and generate a vector $\vec{v}_s = (\alpha' + a^{n+1}, v_2, \cdots, v_{k_s})$, which implies the secret value is $\alpha' + a^{n+1}$.
4) Let $\vec{v}_s = \vec{w}_s + (a^{n+1}, 0, \cdots, 0)$, where $\vec{w}_s = (\alpha', v_2, \cdots, v_{k_s})$. So $\lambda_{\rho(i)} = \vec{S}_i \vec{v}_s = \vec{S}_i \vec{w}_s + a^{n+1} S_{i,1}$.
5) Select $r'_i \in \mathbb{Z}_p$ and compute

$$\begin{aligned} D_{s,i} &= g^{\vec{S}_i \vec{w}_s} \left(T_0 T_{\rho(i)}\right)^{r'_i} g_n^{-\left(t_0 + t_{\rho(i)}\right) S_{i,1}} \\ D'_{s,i} &= g^{r'_i} g_n^{-S_{i,1}} \\ D''_{s,i} &= \left\{ D''_{s,i,j} = T_j^{r'_i} g_n^{-t_j S_{i,1}}, \forall att'_j \in \mathcal{U}_s \backslash att'_{\rho(i)} \right\} \end{aligned}$$

where r_i is implicitly set $r_i = r'_i - a^n S_{i,1}$.

Hence, the signature key is set as

$$sk_{(S,\rho)} = \left\{ (S, \rho), \left\{ D_{s,i}, D'_{s,i}, D''_{s,i} \right\}_{i \in [l_s]} \right\}.$$

- $\mathcal{O}_{dE}(D, \varphi)$: Given a decryption LSSS access structure (D, φ) such that $W_e^* \notin (D, \varphi)$, where $D = (D_{i,j})_{l_e \times k_e}$. $\mathcal{C}$ computes a decryption key $sk_{(D,\varphi)}$ as follows:
Due to $W_e^* \notin (D, \rho)$, there is a vector $\vec{w} = (-1, w_2, \cdots, w_{k_e}) \in \mathbb{Z}_p^{k_e}$ so that $\vec{D}_i \vec{w} = 0$ for $\forall i \in [l_e]$ where $\varphi(i) \in W_e^*$.
1) Select $v'_2, \cdots, v'_{k_e} \in \mathbb{Z}_p$ at random and set $\vec{v}_e = -\left(\alpha' + a^{n+1}\right) \vec{w} + \vec{v}'$, in which $\vec{v}' = \left(0, v'_2, \cdots v'_{k_e}\right)$.
2) If $att_{\varphi(i)} \in W_e^*$, we have $\vec{D}_i \vec{w} = 0$. So $\lambda_{\varphi(i)} = \vec{D}_i \vec{v}_e = \vec{D}_i \vec{v}'$.
Select $\tau_i \in \mathbb{Z}_p$ and compute

$$\begin{aligned} D_{e,i} &= g^{\vec{D}_i \vec{v}'} \left(K_0 K_{\varphi(i)}\right)^{\tau_i} \\ D'_{e,i} &= g^{\tau_i} \\ D''_{e,i} &= \left\{ D''_{e,i,j} = K_j^{\tau_i}, \forall att_j \in \mathcal{U}_e \backslash att_{\varphi(i)} \right\} \end{aligned}$$

3) Otherwise, we have

$$\lambda_{\varphi(i)} = \vec{D}_i \vec{v}_e = \vec{D}_i \left(\vec{v}' - \alpha' \vec{w}\right) - \left(\vec{D}_i \vec{w}\right) a^{n+1}.$$

In this case, $g^{\lambda_{\varphi(i)}}$ contains g_{n+1} which is unknown to $\mathcal{C}$. $\mathcal{C}$ chooses $\tau'_i \in \mathbb{Z}_p$ and τ_i is implicity set as $\tau_i = \tau'_i + \left(\vec{D}_i \vec{w}\right) a^{\varphi(i)}$.
Next, set

$$\begin{aligned} D_{e,i} &= g^{\vec{D}_i \left(\vec{v}' - \alpha' \vec{w}\right)} \cdot \left(K_0 K_{\varphi(i)}\right)^{\tau'_i} \cdot g_{\varphi(i)}^{\left(\gamma_0 + \gamma_{\varphi(i)}\right) \vec{D}_i \vec{w}} \\ &\quad \cdot \prod_{att_j \in W_e^*} \left(g_{\varphi(i)}^{\gamma_j} g_{n+1-j+\varphi(i)} \right)^{-\vec{D}_i \vec{w}} \\ D'_{e,i} &= g^{\tau'_i} g_{\varphi(i)}^{\vec{D}_i \vec{w}} \\ D''_{e,i} &= \left\{ D''_{e,i,j} = K_j^{\tau'_i} \left(g_{\varphi(i)}^{\gamma_j} g_{n+1-j+\varphi(i)} \right)^{\vec{D}_i \vec{w}}, \right. \\ &\quad \left. \forall att_j \in \mathcal{U}_e \backslash att_{\varphi(i)} \right\} \end{aligned}$$

The decryption key is set as

$$sk_{(D,\varphi)} = \left\{ (D, \varphi), \left\{ D_{e,i}, D'_{e,i}, D''_{e,i} \right\}_{i \in [l_e]} \right\}.$$

- $\mathcal{O}_{SC}(m, W_e, W_s)$: On input a message $m \in \mathcal{M}$, an encryption attribute set $W_e \in \mathcal{U}_e$ and a signature attribute set $W_s \in \mathcal{U}_s$, $\mathcal{C}$ chooses a signature access structure (S, ρ) such that $W_s \in (S, \rho)$, runs $\mathcal{O}_{sE}(S, \rho)$ to generate a signature key $sk_{(S,\rho)}$ and runs **Signcrypt**$(pk, m, sk_{(S,\rho)}, W_e, W_s)$ to return $\mathcal{A}$ the ciphertext CT.
- $\mathcal{O}_{US}(CT, (D, \varphi))$: Suppose that $CT = \{W_e, W_s, C_1, C_2, C_3, C_4, \sigma_1, \sigma_2, \sigma_3, \sigma_4\}$. $\mathcal{C}$ first checks whether $C_1 = C_1^*$ or not. If yes, $\mathcal{C}$ outputs $\perp$. Otherwise, $\mathcal{C}$ continues the following process.
1) If $W_e^* \notin (D, \varphi)$, obtain the decryption key $sk_{(D,\varphi)}$ according to the oracle $\mathcal{O}_{dE}(D, \varphi)$, then run the algorithm **Unsgincrypt**$(CT, sk_{(D,\varphi)})$ to return the message m;
2) Otherwise, compute $\mu = H_3(C_1)$ and set

$$e(g,g)^{ac\theta} = e\left(C_4 / C_1^d, g_1\right)^{\left(\frac{\mu}{\mu^*} - 1\right)^{-1}} e\left(C_1, g^{\alpha'}\right)$$

Finally, output the message

$$m = C_3 \oplus H_1\left(e(g,g)^{ac\theta}, \sigma_1, \sigma_4\right)$$

Note that $C_1 = g^\theta$ is random for $\mathcal{A}$, so the probability of $C_1 = C_1^*$ is at most $1/p$.

Challenge: $\mathcal{A}$ sends two messages $m_0^*, m_1^* \in \mathcal{M}$ and a signature attribute set W_s^* to $\mathcal{C}$. $\mathcal{C}$ picks a random bit $b^* \in \{0,1\}$ and signcrypt the message m_b^* with the input W_e^* and W_s^* as follows:

1) Set $C_1^* = g^\theta$ and $\mu^* = H_3(C_1^*)$.
2) Select $v \in \mathbb{Z}_p$, and compute $C_2^* = \left(g^\theta\right)^{\gamma_0}$ and $\sigma_1^* = \left(g^\theta\right)^v$.
3) Choose $\tau^* \in \{0,1\}^{l_\tau}$, and compute $C_3^* = H_1\left(Z \cdot e\left(g^\theta, g^{\alpha'}\right), \sigma_1^*, \tau^*\right) \oplus m_b^*$.
4) Select $\xi \in \mathbb{Z}_p$, and set $\sigma_2^* = g^\xi g_n^{-1}$, which implies $\xi' = \xi - a^n$.
5) Set $C_4^* = \left(g^\theta\right)^d$.
6) Let

$$\begin{aligned} \left(k_1^*, \cdots, k_l^*\right) &= H_2\left(\sigma_2^* || \tau^* || W_s^* || W_e^*\right), \\ \beta^* &= H_4\left(\sigma_1^* || C_2^* || C_3^* || C_4^* || W_s^* || W_e^*\right) \end{aligned}$$

and

$$\begin{aligned} {\sigma_3}^* &= g^{\alpha'} \left(T_0 \prod_{att'_j \in W_s^*} T_j \right)^\xi \cdot \left(g_n^{-t_0} \prod_{att'_j \in W_s^*} g_n^{-t_j} \right) \\ &\quad \cdot \left(g^\theta\right)^{d\beta^* v + x_0 + \sum_{j \in [l]} k_j^* x_j} \end{aligned}$$

where $k_i^* \in \{0,1\}$, for all $i \in [l]$.

7) Set $\sigma_4^* = \tau^*$.

Phase 2: $\mathcal{A}$ continues to query the oracles as in Phase 1. The restriction is that $\mathcal{A}$ cannot query the $\mathcal{O}_{US}(CT, \mathbb{A}_d)$ for any $\mathbb{A}_d$ with $W_e^* \in \mathbb{A}_d$.

Guess: $\mathcal{A}$ returns a guess $b' \in \{0,1\}$. If $b' = b$, then $\mathcal{C}$ can guess that $Z = e\left(g_{n+1}, g^{\theta}\right)$ in the n-DBDHE instance.

We notice that $\mathcal{C}$ can distinguish $Z = e\left(g_{n+1}, g^{\theta}\right)$ or a random element in $\mathbb{G}_2$ if and only if $\mathcal{C}$ doesnot abort the game and $\mathcal{A}$ can output b' such that $b' = b$. Here, $\mathcal{C}$ aborts the game when $C_1 = C_1^*$. After querying q the $\mathcal{O}_{US}(\cdot)$, the possibility of $\mathcal{C}$ aborts is at most q/p. Hence, the possibility of $\mathcal{C}$ in solving the n-DBDHE problem is

$$\Pr\left[e\left(g_{n+1}, g^{\theta}\right) \leftarrow C\left(\vec{y}_{a,\theta}, Z\right)\right] > 1/2 + \varepsilon - q/p.$$

Appendix B

Proof of Theorem 2

Proof. Given an n-CDHE problem instance $(g, g_1, \cdots, g_n, g_{n+2}, \cdots, g_{2n}) \in \mathbb{G}_1^{2n}$, where $a \in \mathbb{Z}_p$, g is the generator of $\mathbb{G}_1$ and $g_i = g^{a^i}$. The simulator $\mathcal{C}$ attempts to compute g_{n+1} though the following game. In which, $\mathcal{C}$ plays the role of challenger and interacts with adversary $\mathcal{A}$. Here, $\{H_i\}_{i=1}^4$ are four one-way, collision resistant hash functions.

Init: $\mathcal{C}$ specifies the encryption attribute space $\mathcal{U}_e = \{att_j\}$ and the signature attribute space $\mathcal{U}_s = \{att'_1, \cdots, att'_n\}$. Then, $\mathcal{A}$ chooses a challenge signature set $W_s^* \subseteq \mathcal{U}_s$ and sends it to $\mathcal{C}$.

Setup: Given the security parameter k, $\mathcal{C}$ generates public parameters as follows:

1) Sample $\alpha' \in \mathbb{Z}_p$, and set $e(g,g)^{ac} = e(g,g)^{\alpha'} \cdot e\left(g_1, g_n\right)$ by implicitly setting $ac = \alpha' + a^{n+1}$.
2) Select $t_0 \in \mathbb{Z}_p$, and set $T_0 = g^{t_0} \prod_{att'_j \in W_s^*} T_j^{-1}$. For $\forall att'_j \in \mathcal{U}_s$, pick $t_j \in \mathbb{Z}_p$ and set $T_j = g^{t_j} g_{n+1-j}$.
3) Choose $\gamma_0, \{\gamma_j\}_{att_j \in \mathcal{U}_e}$ and compute
$$K_0 = g_1 g^{\gamma_0}, \{K_j = g^{\gamma_j}\}_{att_j \in \mathcal{U}_e}.$$
4) Pick $d, d' \in \mathbb{Z}_p$ and set $\delta_1 = g^d, \delta_2 = g^{d'}$.
5) Let $\zeta = k$, where $\zeta(l+1) < p$ and l is the output size of the hash function H_2. Select an integer ϖ with the restriction $0 \leqslant \varpi \leqslant l$. Pick $(d_0, \cdots, d_l) \in \mathbb{Z}_\zeta^{l+1}$, $(x_0, \cdots, x_l) \in \mathbb{Z}_p^{l+1}$ and compute $y_0 = g_n^{p-\zeta\varpi+d_0} g^{x_0}$, $\left\{y_j = g_n^{d_j} g^{x_j}\right\}_{j\in[l]}$.
For $\vec{k} = (k_1, \cdots, k_l) \in \{0,1\}^l$,let
$$F\left(\vec{k}\right) = p - \zeta\varpi + d_0 + \sum_{j\in[l]} k_j d_j$$
$$J\left(\vec{k}\right) = x_0 + \sum_{j\in[l]} k_j x_j$$
So, $y_0 \prod_{j\in[l]} y_j^{k_j} = g_n^{F(\vec{k})} g^{J(\vec{k})}$. In addition, set
$$\mathcal{K}\left(\vec{k}\right) = \begin{cases} 0, & if\ d_0 + \sum_{j\in[l]} k_j d_j = 0 \mod \zeta \\ 1, & otherwise. \end{cases}$$
Due to $\zeta(l+1) < p$, when $\mathcal{K}\left(\vec{k}\right) = 1$, $F\left(\vec{k}\right) \neq 0$.

□ **Phase:** $\mathcal{A}$ queries the following oracles for polynomial times:

- $\mathcal{O}'_{sE}(S, \rho)$: Given a signature LSSS access structure (S, ρ) such that $W_s^* \notin (S, \rho)$, where $S = (S_{i,j})_{l_s \times n_s}$. $\mathcal{C}$ computes a signature key $sk_{(S,\rho)}$ as follow.
Since $W_s^* \notin (S, \rho)$, there is a vector $\vec{w} = (-1, w_2, \cdots, w_{k_s})$ so that $\vec{S}_i \vec{w} = 0$ for $\forall i \in [l_s]$, where $\rho(i) \in W_s^*$.
  1) Pick $v'_2, \cdots, v'_{k_s} \in \mathbb{Z}_p$ and set $\vec{v}_s = -\left(\alpha' + a^{n+1}\right) \cdot \vec{w} + \vec{v}'$, where $\vec{v}' = \left(0, v'_2, \cdots, v'_{k_s}\right)$.
  2) If $att_{\rho(i)} \in W_s^*$, there exists $\vec{S}_i \vec{w} = 0$. So $\lambda_{\rho(i)} = \vec{S}_i \vec{v}_s = \vec{S}_i \vec{v}'$.
  3) Select $\tau_i \in \mathbb{Z}_p$, and compute
$$D_{s,i} = g^{\vec{S}_i \vec{v}'} \left(T_0 T_{\rho(i)}\right)^{\tau_i}$$
$$D'_{s,i} = g^{\tau_i}$$
$$D''_{s,i} = \left\{D''_{s,i,j} = T_j^{\tau_i}, \forall j \in [n] \setminus \{\rho(i)\}\right\}$$
  4) Otherwise, $\lambda_{\rho(i)} = \vec{S}_i \vec{v}_s = \vec{S}_i\left(\vec{v}' - \alpha' \vec{w}\right) - \left(\vec{S}_i \vec{w}\right) a^{n+1}$. Select $\tau'_i \in \mathbb{Z}_p$ and set
$$D_{s,i} = g^{\vec{S}_i\left(\vec{v} - \alpha' \vec{w}\right)} \left(T_0 T_{\rho(i)}\right)^{\tau_i} g_{\rho(i)}^{\left(t_0 + t_{\rho(i)}\right)\vec{S}_i \vec{w}} \prod_{att'_j \in W_e^*} \left(g_{\rho(i)}^{t_j} g_{n+1-j+\rho(i)}\right)^{-\vec{S}_i \vec{w}}$$
$$D'_{s,i} = g^{\tau' i} g_{\rho(i)}^{\vec{S}_i \vec{w}}$$
$$D''_{s,i} = \left\{D''_{s,i,j} = T_j^{\tau' i} \left(g_{\rho(i)}^{t_j} g_{n+1-j+\rho(i)}\right)^{\vec{S}_i \vec{w}}, \forall j \in [n] \setminus \rho(i)\right\}$$
where τ_i is implicity set
$$\tau_i = \tau'_i + \left(\vec{S}_i \vec{w}\right) a^{\rho(i)}.$$

Hence, the signature key is set as

$$sk_{(S,\rho)} = \left\{(S,\rho), \left\{D_{s,i}, D'_{s,i}, D''_{s,i}\right\}_{i\in[l_s]}\right\}.$$

- $\mathcal{O}'_{dE}(D,\varphi)$: Given a decryption LSSS access structure (D,φ), $\mathcal{C}$ generates a decryption key $sk_{(D,\varphi)}$ as follow:
  1) Pick $v_2,\cdots,v_{k_e}\in\mathbb{Z}_p$ and set
  $$\vec{v}_e = \vec{w}_e + \left(a^{n+1},0,\cdots,0\right),$$
  where $\vec{w}_e = (\alpha', v_2,\cdots,v_{k_e})$. So
  $$\lambda_{\varphi(i)} = \vec{D}_i\vec{v}_e = \vec{D}_i\vec{w}_e + a^{n+1}D_{i,1}.$$
  2) Select $r'_i\in\mathbb{Z}_p$ and implicitly set $r_i = r'_i - a^n D_{i,1}$. Compute
  $$\begin{aligned} D_{e,i} &= g^{\vec{D}_i\vec{w}_e}\left(K_0K_{\varphi(i)}\right)^{r'_i} g_n^{-\left(\gamma_0+\gamma_{\varphi(i)}\right)D_{i,1}} \\ D'_{e,i} &= g^{r'_i} g_n^{-D_{i,1}} \\ D''_{e,i} &= \left\{D''_{e,i,j} = K_j^{r'_i} g_n^{-\gamma_j D_{i,1}}, \right. \\ &\quad \left. \forall att_j \in U_e \backslash att_{\varphi(i)}\right\} \end{aligned}$$

  $\mathcal{C}$ outputs the decryption key
  $$sk_{(D,\varphi)} = \left\{(D,\varphi), \left\{D_{e,i}, D'_{e,i}, D''_{e,i}\right\}_{i\in[l_e]}\right\}.$$
- $\mathcal{O}'_{SC}(m,W_e,W_s)$: $\mathcal{C}$ constructs a signature access structure (S,ρ) such that $W_s\in(S,\rho)$. If $W_s^*\notin(S,\rho)$, $\mathcal{C}$ runs $\mathcal{O}'_{sE}(S,\rho)$ to obtain a signing key $sk_{(S,\rho)}$, then outputs a ciphertext CT by running the algorithm $\mathbf{Signcrypt}(pk,m,sk_{(S,\rho)},W_e,W_s)$. Otherwise, $\mathcal{C}$ outputs a ciphertext as follow.
  1) Pick $\theta',\xi'\in\mathbb{Z}_p$ and set $\sigma_2 = g^{\xi'}$.
  2) Select $\tau\in\{0,1\}^{l_\tau}$ such that $\mathcal{K}\left(\vec{k}\right)\neq 0$, sets $\sigma_4=\tau$ and compute $C_1 = g^{\theta'} g_1^{-1/F\left(\vec{k}\right)}$, which implies $\theta = \theta' - a\Big/F\left(\vec{k}\right)$.
  3) Compute
  $$\begin{aligned} C_2 &= \left(K_0\prod_{att_j\in W_e}K_j\right)^{\theta'} \\ &\quad\cdot\left(g_2 g_1^{\gamma_0}\prod_{att_j\in W_e} g_1^{\gamma_j}\right)^{\frac{-1}{F\left(\vec{k}\right)}} \end{aligned}$$
  4) Select $\vartheta\in\mathbb{Z}_p$ and compute $\sigma_1 = g_1^{\vartheta}$.
  5) Compute
  $$\begin{aligned} C_3 &= H_1\left(e(g,g)^{ac\theta'}\cdot e(g,g_1)^{-\alpha'/F\left(\vec{k}\right)}\right. \\ &\quad\left.\cdot e(g_2,g_n)^{-1/F\left(\vec{k}\right)},\sigma_1,\tau\right)\oplus m \end{aligned}$$
  6) Set $\mu = H_3(C_1)$, and compute
  $$C_4 = \left(\delta_1^{\mu}\delta_2\right)^{\theta'}\left(g_1^{\mu d+d'}\right)^{-1/F\left(\vec{k}\right)}.$$
  7) Set $\beta = H_4(\sigma_1||C_2||C_3||C_4||W_s||W_e)$.
  8) Compute
  $$\sigma_3 = g^{\alpha'}\left(T_0\prod_{att'_j\in W_s}T_j\right)^{\xi'}\left(g_n^{F\left(\vec{k}\right)}g^{J\left(\vec{k}\right)}\right)^{\theta'}$$
- $\mathcal{O}'_{US}(CT,(D,\varphi))$: Given the ciphertext CT and the decryption access structure (D,φ), $\mathcal{C}$ first runs the oracle $\mathcal{O}'_{dE}(D,\varphi)$ to get a decryption key $sk_{(D,\varphi)}$ and outputs $\mathcal{A}$ the message m by the algorithm $\mathbf{Unsigncrypt}\ (pp,CT,sk_{(D,\varphi)})$.

Forgery: For message m^*, such that (m^*,W_s^*,W_e^*) has never been queried in $\mathcal{O}'_{SC}(m,W_e,W_s)$, $\mathcal{A}$ sends a forgery $CT^* = \{W_s^*,W_e^*,C_1^*,C_2^*,C_3^*,C_4^*,\sigma_1^*,\sigma_2^*,\sigma_3^*,\sigma_4^*\}$ for a message m^* and a decryption access structure $\mathbb{A}_d^*$ to $\mathcal{C}$, where $\mathbf{Unsigncrypt}(CT^*,sk_{\mathbb{A}_d^*}) = m^*$. Then, $\mathcal{C}$ computes

$$\vec{k}^* = (k_1^*,\cdots,k_l^*) = H_2(\sigma_2^*||\sigma_4^*||W_s^*||W_e^*)$$

and checks whether $F\left(\vec{k}^*\right) = 0$. If not, $\mathcal{C}$ aborts. Otherwise, $\mathcal{C}$ verifies the validity of CT^* though Equation (2).

If $\mathcal{A}$ wins the game, i.e. the CT^* can passe the verification, which means that

$$\begin{aligned} C_1^* &= g^{\theta}, \sigma_1^* = g^{\theta\vartheta}, \sigma_2^* = g^{\xi'}, \mu^* = H_3(C_1^*) \\ \beta^* &= H_4(\sigma_1^*||C_2^*||C_3^*||C_4^*||W_s^*||W_e^*), C_4^* = g^{\left(d\mu^*+d'\right)\theta} \\ \sigma_3^* &= g^{\alpha'+a^{n+1}}\left(T_0\prod_{att'_j\in W_s^*}T_j\right)^{\xi'}\left(g_n^{F\left(\vec{k}^*\right)}g^{J\left(\vec{k}^*\right)}\right)^{\theta}(C_4^*)^{\beta^*\vartheta} \end{aligned}$$

In order to provide a perfect simulation, the game cannot abort in **Forgery** phase. Following Selvi *et al.* [18], we have

$$\Pr[\neg\, abort] = \frac{1}{\zeta}\cdot\frac{1}{l+1} = \frac{1}{k(l+1)}.$$

Suppose that $\mathcal{A}$ wins the game with an advantage ε, $\mathcal{C}$ can solve the n-CDHE problem with the advantage $\varepsilon' = \varepsilon/(k(l+1))$. □

Appendix C

Proof of Theorem 3

Proof. Given a DL instance (g,h,f,f^{r_1},g^{r_2}), where $g,h,f\in\mathbb{G}_1$ and $r_1,r_2\in\mathbb{Z}_p$. The simulator $\mathcal{C}$ attempts to compute $h^{r_1+r_2}$ through the following game, in which, $\mathcal{C}$ plays the role of challenger and interacts with adversary $\mathcal{A}$. Here, H_4 is a one-way hash function.

Init: $\mathcal{C}$ gives the encryption attribute space $\mathcal{U}_e = \{att_1, \cdots, att_n\}$. Then, $\mathcal{A}$ chooses a challenge $W_e^* \subseteq \mathcal{U}_e$ and sends it to $\mathcal{C}$.

Setup: Given the security parameter k, $\mathcal{C}$ generates public parameters as follows:

1) Set $g^a = h, g^c = f$, where $a, c \in \mathbb{Z}_p$ are unknown.
2) Select $d \in \mathbb{Z}_p$ and compute $g^b = f^d = g^{cd}$, which implies $b = cd$.
3) For each $att_j \in \mathcal{U}_e \backslash W_e^*$, pick $\alpha_j, \beta_j \in \mathbb{Z}_p$ and set $K_j = K_0^{-1} f^{\alpha_j} g^{\beta_j}$.
4) For each $att_j \in W_e^*$, set $K_j = K_0^{-1} g^{\beta_j}$.

$\mathcal{C}$ outputs the public parameters $pp = \left(e, g, h, f, f^d, K_0, \{K_j\}_{att_j \in \mathcal{U}_e}\right)$ and sets the master key as $msk = d$.

Phase 1: $\mathcal{A}$ queries the following oracles for polynomially times:

- $\mathcal{O}_{dE}(D, \varphi)$: $\mathcal{A}$ sends an encryption access structure (D, φ) to $\mathcal{C}$, where $D = (D_{i,j})_{l_e \times k_e}$. If $W_e^* \in (D, \varphi)$, $\mathcal{C}$ outputs $\perp$. Otherwise, $\mathcal{C}$ performs as follow.

  Due to $W_e^* \notin (D, \varphi)$, there exists a vector $\vec{w} = (-1, w_2, \cdots, w_{k_e}) \in \mathbb{Z}_p^{k_e}$ such that $\vec{D}_i \vec{w} = 0$ for all $i \in [l_e]$ where $\varphi'(i) \in W_e^*$.

  1) Pick $v_2', \cdots v_{k_e}' \in \mathbb{Z}_p^{k_e}$ and set $\vec{v}_e = a\vec{w} + \vec{v}'$, where $\vec{v}' = \left(0, v_2', \cdots, v_{k_e}'\right)$.
  2) If $att_{\varphi(i)} \in W_e^*$, we have $\vec{D}_i \vec{w} = 0$. Hence, $\lambda_{\varphi(i)} = \vec{D}_i \vec{v}'$. Select a random number $\tau_i \in \mathbb{Z}_p$ and compute

$$\begin{aligned} D_{e,i} &= f^{\vec{D}_i \vec{v}'} \left(g^{\beta_{\varphi(i)}}\right)^{\tau_i} \\ &= g^{c\lambda_{\varphi(i)}} \left(K_0 K_{\varphi(i)}\right)^{\tau_i} \\ D'_{e,i} &= g^{\tau_i} \end{aligned}$$

  3) Otherwise, $\mathcal{C}$ selects $\tau_i' \in \mathbb{Z}_p$ and computes

$$\begin{aligned} D_{e,i} &= \left(g^{\lambda_{\varphi(i)}}\right)^{\frac{-\beta_{\varphi(i)}}{\alpha_{\varphi(i)}}} \left(f^{\alpha_{\varphi(i)}} g^{\beta_{\varphi(i)}}\right)^{\tau_i'} \\ &= f^{\lambda_{\varphi(i)}} \left(f^{\alpha_{\varphi(i)}} g^{\beta_{\varphi(i)}}\right)^{\tau_i' - \frac{\beta_{\varphi(i)}}{\alpha_{\varphi(i)}}} \\ D'_{e,i} &= g^{\lambda_{\varphi(i)} \left(\frac{-1}{\alpha_{\varphi(i)}}\right)} g^{\tau_i'} \end{aligned}$$

  where, τ is implicity set as $\tau_i = \tau_i' - \frac{\lambda_{\varphi(i)}}{\alpha_{\varphi(i)}}$.

  $\mathcal{C}$ returns

$$sk = \left\{(D, \varphi), \left\{D_{e,i}, D'_{e,i}\right\}_{i \in [l_e]}\right\}.$$

- $\mathcal{O}_{GT}(kw, (D, \varphi))$: $\mathcal{C}$ first runs the oracle $\mathcal{O}_{dE}(D, \varphi)$ to obtain $sk_{(D,\varphi)}$, then returns $\mathcal{A}$ a token T by the algorithm **GenToken**$(sk_{(D,\varphi)}, kw)$. If $W_e^* \in (D, \varphi)$, $\mathcal{C}$ adds kw to L_{kw}, a keyword list which is initially empty.

Challenge: $\mathcal{A}$ sends two keywords kw_0 and kw_1 to $\mathcal{C}$, where kw_0 and kw_1 do not belong to keyword set list L_{kw}. Then, $\mathcal{C}$ randomly selects a bit $b \in \{0, 1\}$ and runs the algorithm **GenIndex**(kw_b, W_e^*) to generate the index $I = \{W_0, W_1, W_2, \{W_j | att_j \in W_e^*\}\}$, where

$$W_0 = f^{r_1}, W_1 = T(f^{r_1})^{dH_4(kw_b)}, W_2 = g^{r_2}$$

For each $att_j \in W_e^*$, set $W_j = (g^{r_2})^{\beta_j}$.

Phase 2: This phase is performed as in Phase 1. The restriction is that if $W_e^* \in (D, \varphi)$, $\mathcal{A}$ cannot query the oracle $\mathcal{O}_{GT}$ with the input $((D, \varphi), kw_0)$ or $((D, \varphi), kw_1)$.

Guess: $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$. If $b' = b$, $\mathcal{C}$ outputs $T = h^{r_1 + r_2}$. Otherwise, $\mathcal{C}$ outputs $T = R \in \mathbb{G}_2$.

Suppose that $\mathcal{A}$ wins the above game with an advantage ε. In the challenge phase, if $T = h^{r_1+r_2}$, the index I is valid, then $\mathcal{A}$ outputs $b' = b$ with the probability $\frac{1}{2} + \varepsilon$. Otherwise, T is a random element in $\mathbb{G}_2$, so the index I is not valid, $\mathcal{A}$ outputs $b' = b$ with the probability $\frac{1}{2}$.

Therefore, $\mathcal{C}$ can solve the DL problem with an advantage $\frac{\varepsilon}{2}$.

□

Appendix D

Proof of Theorem 4

Proof. We utilize a challenger $\mathcal{C}$ to conduct the following keyword secrecy game.

Setup: $\mathcal{C}$ first chooses random elements $a, b, c \in \mathbb{Z}_p$, $f \in \mathbb{G}_1$. For each $att_j \in \mathcal{U}_e$, $\mathcal{C}$ selects $\alpha_j \in \mathbb{Z}_p$ and sets $K_j = g^{\alpha_j}$. Then, public parameters are set as $pk = \left(e, g, g^a, g^b, g^c, \{K_j | att_j \in \mathcal{U}_e\}\right)$ and master key is set as $mk = (a, b, c)$.

Phase: The adversary $\mathcal{A}$ queries the following two oracles for polynomial times.

- $\mathcal{O}_{dE}(D, \varphi)$: $\mathcal{C}$ runs the algorithm **dExtract**$(msk, (D, \varphi))$ to gain a secret key $sk_{(D,\varphi)}$, sends it to $\mathcal{A}$, and adds (D, φ) to the list L_{dE}.
- $\mathcal{O}_{GT}(kw, (D, \varphi))$: $\mathcal{C}$ first runs the oracle $\mathcal{O}_{dE}(D, \varphi)$ to obtain secret key $sk_{(D,\varphi)}$, then calls **GenToken**$((sk_{(D,\varphi)}), kw)$ algorithm to generate token T for $\mathcal{A}$.

Challenge: $\mathcal{A}$ first chooses an attribute set W_e^*, then $\mathcal{C}$ selects an encryption access structure (D^*, φ^*) such that $W_e^* \in (D^*, \varphi^*)$ and computes $sk^*_{(D,\varphi)}$ according to the oracle $\mathcal{O}_{dE}(msk, (D^*, \varphi^*))$. Next, $\mathcal{C}$ randomly selects a keyword kw^* and computes I^* and T^*, where $\forall\ (D, \varphi) \in L_{dE}$, $W_e^* \notin (D, \varphi)$.

Guess: $\mathcal{A}$ outputs a keyword set kw' to $\mathcal{C}$, then $\mathcal{C}$ computes I' by running the algorithm **GenIndex**(W_e^*, kw'). If **Search**$(T^*, I') = 1$, then $\mathcal{A}$ wins the game.

Suppose that $\mathcal{A}$ has issued q_{kw} different keyword sets before returning kw', and the probability of $\mathcal{A}$ winning the keyword secrecy game is at most $\frac{1}{|\mathcal{M}|-|q_{kw}|}+\varepsilon$, where $|q_{kw}|$ is denoted as the number of the different keywords. The size of remaining keyword set space is $|\mathcal{M}|-|q_{kw}|$, and H_4 is denoted as a one-way hash function which means recovering kw^* from $H_4(kw^*)$ has at most a negligible probability ε. Therefore, given $|q_{kw}|$ distinct keywords $\mathcal{A}$ has queried, $\mathcal{A}$ wins the keyword secrecy game with the probability at most $\frac{1}{|\mathcal{M}|-|q_{kw}|}+\varepsilon$. □

Biography

Zhenhua Liu received his B.S. degree from Henan Normal University, M.S., and Ph.D degrees from Xidian University, China, in 2000, 2003 and 2009, respectively. He is currently a professor with Xidian University. His research interests include cryptography and information security.

Yaqing Fan received her B.S. degree in 2015 from College of Mathematics and Information Sciences, Taiyuan Normal University. Now, she is a master degree student in Mathematics at Xidian University. Her research interests include cryptography and cloud security.