An efficient attribute-based authentication scheme with multiple authorities in public cloud

Xin Liu1,2*, Man Guo3, Bin Zhang1,2 and Xuzhou Li1,2

1 School of Information Engineering, Shandong Youth University of Political Science, Jinan, Shandong, 250013, China
2 Key Laboratory of Information Security and Intelligent Control in Universities of Shandong (Shandong Youth University of Political Science), Shandong Youth University of Political Science, Jinan, Shandong, 250013, China
3 Public Course Teaching Department, Shandong University of Art & Design, Jinan, Shandong, 250300, China

*Corresponding author’s e-mail: lxonne@163.com

Abstract. Currently, attribute-based authentication provides a feasible solution for fine-grained access control in cloud environment. However, the existing schemes can not solve the following problems at the same time, that is, how to ensure that the computation cost of the client does not depend on the size of underlying access structure, and how to introduce distributed authorities to manage and maintain the attribute universe. To solve the above problems, an efficient multi-authority attribute-based authentication scheme is proposed. The new scheme uses the technique of distributed attribute-based encryption to realize the access control of anonymous users, and reduces users’ computation burden by optimizing the standard implementation zero-knowledge proof and outsourcing users’ computing tasks in the authentication stage. Under the new definition of security, it can be proved that the new scheme is secure and satisfies many attractive properties, such as introducing distributed authorities, supporting outsourcing computation, satisfying attribute anonymity.

1. Introduction

In the cloud environment, service providers (SP) often need to implement fine-grained access control on various cloud resources (such as online video, music, etc.). Specifically, they want to use attributes to define access control policies. For potential users, access to cloud resources can only be allowed when their attributes satisfy the access policies of these resources. On the other hand, users hope that SP can only know the fact that they have access qualifications, but cannot know what attributes they have. Fortunately, attribute-based authentication provides a satisfactory solution for these applications.

In [1], Zhou et al. proposed an attribute-based authentication scheme for distributed medical system. In order to protect their privacy (such as illness), patients can use the attribute tree mechanism to authorize doctors’ access. In [2], Lian et al. considered the dynamic change of authentication strategies in large-scale systems. For this reason, Lian et al. designed a general adaptive attribute-based authentication scheme based on Attribute-based Signature (ABS). In [3], Li et al. provided a joint attribute-based authentication scheme, which solved the problems of cooperative authentication and attribute combination of multiple users belonging to the same department in the enterprise computing environment. In [4], Li et al. put forward a multi-authority attribute-based authentication scheme.
However, their scheme requires the distributed attribute authorities to perform a large number of interactions in the setup phase. In [5], Yang et al. constructed a traceable attribute-based authentication scheme. This scheme was obtained by combining the ciphertext policy attribute-based encryption scheme of Bethencourt et al. [6] with group signature. Although traceability is achieved, Yang et al.’s scheme requires the group manager to be honest. Otherwise, anonymity cannot be guaranteed even if users remain honest. In [7], Yang et al. proposed an attribute-based authentication scheme based on dynamic attribute tree. The advantage of their scheme is that the underlying attribute tree is constructed in a “bottom-up” way, so when the attribute tree changes, there is no need to reissue the attribute credentials for users. However, users are required to provide the subset of the used attributes to SP in the authentication stage, thus their privacy is revealed. In [8], Liu et al. designed a k-times anonymous attribute-based authentication scheme based on direct anonymous attestation, which supports member revocation and attribute update.

1.1. Related research
Recently, researchers have also proposed a type of related technique, called attribute-based access control schemes (such as [9]). This kind of scheme is based on Attribute-based Encryption (ABE), and it is mainly used to implement fine-grained access control for big data. In the design of access control schemes, researchers need to consider how to avoid users’ computation cost depending on the size of underlying access control policy. Therefore, many researches focus their attention on how to outsource the heavy decryption tasks of the underlying ABE scheme (such as [10]). It should be noted that performing bilinear pairing operations is a key factor affecting users’ computation efficiency. Feasible solutions are to design a lightweight ABE scheme (e.g., [11]) that does not need to perform bilinear pairing operation, or to use the optimization techniques such as the pairing delegation operation protocol, so as to avoid performing bilinear pairing operations (e.g., [12]).

1.2. Contribution
We believe that the existing schemes have the following shortcomings. (1) Most schemes require users to fully trust Attribute Authority (AA), that is, users cannot verify the attribute key provided by AA. (2) In the authentication phase of most schemes, the computation cost of users has linear dependency on the size of attribute-based access structure. (3) There is only one AA in most schemes, so it is easy to cause single point failure. Inspired by Wang et al. [13], we construct an efficient multi-authority attribute-based authentication scheme based on distributed ABE. Our scheme has the following advantages. (1) There is no need to assume that AA is trustworthy. (2) In the setup phase, there is no need for multiple AAs to perform a large number of interactive operations. (3) Users are allowed to outsource most of their computation tasks in the authentication phase, which makes their computation cost independent of the size of access structure.

1.3. Orgnization of the paper
In Section 2, we introduce the key techniques used in the construction of the new scheme. In Section 3, we propose a syntax definition and a security definition for multi-authority attribute-based authentication scheme that supports outsourcing decryption. In Section 4, we provide a detailed description of the new scheme. In Section 5, we provide the details of the underlying zero-knowledge proofs. Then, we give security analysis and performance analysis in Section 6 and section 7 respectively. Finally, we summarize in Section 8.

2. Preliminaries

2.1 Bilinear group
The new scheme is constructed on bilinear groups. Specifically, let $\mathbb{G} = \langle g \rangle, \mathbb{G}_T$ denote the cyclic multiplicative group of order $p$, which is a prime number. We call the mapping $\hat{e}: \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ a
bilinear pairing if it satisfies the following properties: (1) Bilinear, that is, for all elements \( a, b \in \mathbb{Z}_p \), \( \hat{e}(g^a, g^b) = \hat{e}(g, g)^{ab} \) is satisfied. (2) Non degeneracy, i.e. \( \hat{e}(g, g) \neq 1 \). Here, 1 represents the unit element of group \( G_T \). (3) Computability, that is, for any \( g_1, g_2 \in G \), \( \hat{e}(g_1, g_2) \) can be calculated effectively.

2.2. Distributed attribute-based encryption

ABE is a kind of one-to-many public key encryption scheme, which is conducive to actualize fine-grained access control in the cloud environment. ABE can be divided into two types: Key Policy-ABE (KP-ABE) and Ciphertext Policy-ABE (CP-ABE). In KP-ABE scheme, user’s key is generated according to the access control policy, and the ciphertext is generated according to the attribute set, which needs to be embedded in the ciphertext. In CP-ABE scheme, user’s key is generated according to his own attribute set, and the ciphertext is generated according to the access control policy, which needs to be embedded in the ciphertext. In [14], Müller et al. proposed the concept of Distributed ABE (DABE). Compared with the standard ABE scheme (such as [6]), the scheme introduces a central authority and multiple AAs. The central authority is responsible for generating system parameters, and each AA can independently determine the number, structure, and semantics of their managed attributes. In order to encrypt messages, the encrypting party needs to define an access policy in DNF (Disruptive Normal Form), i.e., \( A = \bigvee_i (\land_{A_i \in S} A_i) \). For the decryptor, only when his/her attribute set satisfies one of the disjunctive paradigms can he/she successfully decrypt.

2.3. Zero-knowledge proof

In this paper, we use a special kind of zero-knowledge proof of knowledge (zero-knowledge proof for short). A standard zero-knowledge proof is a three-round interactive protocol executed by a prover \( \mathcal{P} \) and a verifier \( \mathcal{V} \), which is divided into three stages, i.e., commitment phase, challenge phase and response phase. In the challenge phase, \( \mathcal{V} \) sends its randomly selected challenge string \( cha \) to \( \mathcal{P} \). In the zero-knowledge proof of this paper, \( cha \) is not selected by \( \mathcal{V} \), but generated by \( \mathcal{P} \) using collision-resistant hash function. In describing such protocols, we use the symbols invented by Camenich et al. [15]. For example, \( \pi = PK\{\alpha : y = g^x\} \) represents the following zero-knowledge proof, i.e., \( \mathcal{P} \) proves that he/she owns the discrete logarithm of \( y \) based on \( g \).

3. Definition of multi-authority attribute-based authentication scheme

3.1. Participants

We assume that there are the following participants in a multi-authority authentication scheme. \( \mathcal{CA} \) is the central authority of the system, and its specific responsibilities are as follows: (1) To generate system public parameters. (2) To generate the master key of the underlying encryption scheme. (3) To help users generate personal public/private key pairs and member credentials. \( AA \)s are attribute authorities, who are responsible for generating attribute keys for users. Here, \( AA \) refers to one of the multiple attribute authorities, namely \( AA_i \). For the sake of simplicity, we will omit the subscript \( i \) in the following description. \( CS \) is a cloud server, which can provide users with outsourcing decryption services. \( SP \) is a service provider, which is responsible for providing cloud resources to users and setting access structure \( A \) for cloud resources. For user \( U \), to access cloud resources, he/she needs to first register with \( CA \) (to become a legal member), then apply for attribute key from \( AA \), and finally perform anonymous authentication with \( SP \). The participants and their interactions are shown in Figure 1.
3.2. Syntax definition

A multi-authority authentication scheme consists of the following algorithms and protocols:

- **CASetup**. Taking the security parameter $\kappa$ as input, the algorithm outputs the system public parameter $\text{params}$ and the master key $MK$.

- **AASetup**. Taking the system public parameter $\text{params}$ as input, the algorithm outputs $\mathcal{A}_A$'s private key $SK_{\mathcal{A}_A}$.

- **Registration**. This is a two-party protocol executed by $\mathcal{U}$ and $\mathcal{C}_A$, in which $\mathcal{U}$ takes the system public parameter $\text{params}$ and identity $ID_\mathcal{U}$ as input, and $\mathcal{C}_A$ takes $\text{params}$ and the master key $MK$ as input. After the end of the protocol, $\mathcal{U}$ outputs his/her public/private key pair $(PK_\mathcal{U}, SK_\mathcal{U})$ and membership credential $\text{cred}_\mathcal{U}$.

- **AKeyGen**. This is a two-party protocol executed by $\mathcal{U}$ and $\mathcal{A}_A$, in which $\mathcal{U}$ takes the system public parameter $\text{params}$, attribute set $S$, public/private key pair $(PK_\mathcal{U}, SK_\mathcal{U})$ and membership credential $\text{cred}_\mathcal{U}$ as input, and $\mathcal{A}_A$ takes $\text{params}$ and private key $SK_{\mathcal{A}_A}$ as input. After the end of the protocol, $\mathcal{U}$ outputs his/her attribute key $\{PK_{\mathcal{A}_A}, SK_{\mathcal{A}_A}\}_i$ of $S$.

- **Authentication**. This is a two-party protocol executed by $\mathcal{U}$ and $\mathcal{S}_P$, in which $\mathcal{U}$ takes the system public parameter $\text{params}$, attribute set $S$, his/her public/private key pair $(PK_\mathcal{U}, SK_\mathcal{U})$, membership credential $\text{cred}_\mathcal{U}$ and attribute key $\{PK_{\mathcal{A}_A}, SK_{\mathcal{A}_A}\}_i$ as input, and $\mathcal{S}_P$ takes $\text{params}$ as input. If the protocol ends successfully, $\mathcal{S}_P$ allows $\mathcal{U}$ to pass the authentication.

- **CreateTK**. Taking $(PK_{\mathcal{U}_i}, SK_{\mathcal{A}_A}_i, CT)$ as the input, the algorithm outputs the transformation key $TK$, the retrieving key $RK$ and $CT_\mathcal{U}$, where $CT_\mathcal{U}$ represents a part of the ciphertext $CT$.

- **Transform**. Taking $(PK, TK_\mathcal{U}, CT)$ as the input, the algorithm outputs the outsourcing decryption result $V$.

- **Decrypt**. Taking $(A, CT, V, RK)$ as the input, the algorithm outputs the plaintext.

3.3. Security definition

Figure 1. Participants in an attribute-based authentication scheme.
A secure multi-authority attribute-based authentication scheme should satisfy the following properties:

- **Correctness.** The registered user $\mathcal{U}$ can use his/her public/private key pair $(PK_\mathcal{U}, SK_\mathcal{U})$ and membership credential $\text{cred}_\mathcal{U}$ to apply to $\mathcal{AA}$ for the attribute key $\{PK_A, SK_A\}_{i=1}^{\mid S\mid}$ of the attribute set $S$. For a given access structure $\mathbb{A} = \bigvee_{i=1}^{m} (\wedge_{A \in s} A)$, as long as $\text{cred}_\mathcal{U}$ is legal and any conjunction of $\mathbb{A}$ is satisfied by $\mathcal{U}$, $\mathcal{U}$ can successfully execute the Authentication protocol with $\mathcal{AO}$. During the execution of Authentication protocol, $\mathcal{U}$ uses his/her attribute key to decrypt the underlying DABE ciphertext $CT$ generated by $\mathcal{AO}$ using $\mathbb{A}$. As an alternative, $\mathcal{U}$ invokes the CreateTK algorithm to generate the transmation key $TK$ and the retrieving key $RK$. Then $\mathcal{U}$ provides $TK$ and the partial ciphertext $CT$ to $\mathcal{CS}$. After receiving the outsourcing decryption result $V$ returned by $\mathcal{CS}$, $\mathcal{U}$ uses $RK$ to invoke the outDecrypt algorithm to recover the plaintext.

- **Anonymity.** In the Authentication protocol, even if $\mathcal{AO}$ colludes with $\mathcal{AA}$ and $\mathcal{CA}$, the real identity of $\mathcal{U}$ cannot be determined.

- **Unlinkability.** Even if $\mathcal{AO}$ colludes with $\mathcal{AA}$ and $\mathcal{CA}$, it is impossible for them to judge whether the execution transcript of two Authentication protocols corresponds to the same user. In addition, they cannot associate the execution transcript of an Authentication protocol with that of a AKeyGen or a Registration protocol.

- **Ciphertext security.** The underlying DABE scheme should at least satisfy the CPA security, i.e., it is secure under the chosen-plaintext attack.

- **Verifiability.** After receiving the outsourcing decryption result $V$ returned by $\mathcal{CS}$, $\mathcal{U}$ can verify the correctness of $V$.

4. **Specific description of the new scheme**

In this section, instantiation of our new scheme is described.

**CASetup.** Taking the security parameter $\kappa$ as the input, $\mathcal{CA}$ performs the following operations:

1. Defines the bilinear group $(\mathbb{G}, \mathbb{G}_T, p, \hat{e})$, where $\hat{e} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$.
2. Defines the parameters of the underlying Boneh-Boyen scheme. Specifically, selects $g_0, h \in \mathbb{G}$, selects $p \in \mathbb{Z}_p$ and sets $\hat{e}(g, g) = h^\gamma$.
3. Defines the parameters of the underlying DABE scheme. Specifically, selects $g, \overline{g} \in \mathbb{G}$, selects $y \in \mathbb{Z}_p$ and sets $Y = \hat{e}(g, g^{\gamma})$.
4. Defines the collision-resistant hash functions $\mathcal{H}_0 : \{0, 1\}^* \rightarrow \mathbb{Z}_p^\ast$, $\mathcal{H}_1 : \{0, 1\}^* \times \mathbb{G} \rightarrow \mathbb{Z}_p^\ast$, $\mathcal{H}_2 : \mathbb{G}_T \rightarrow \mathbb{Z}_p^\ast$, $\mathcal{H}_3 : \mathbb{G}_T^* \times \{0, 1\}^* \rightarrow \mathbb{Z}_p^\ast$.
5. Sets the public system parameters $\text{params} = (\mathbb{G}, \mathbb{G}_T, p, \hat{e}, g, h, w, g, \overline{g}, Y, \mathcal{H}_0, \mathcal{H}_1, \mathcal{H}_2, \mathcal{H}_3)$. At the same time, keeps $MK = (h^\gamma, \gamma)$ in secret.

**AASetup.** $\mathcal{AA}$ does the following operations: selects $x_{\mathcal{AA}} \in \mathbb{Z}_p$ and sets $SK_{\mathcal{AA}} = x_{\mathcal{AA}}$.

**Registration.** $\mathcal{U}$ and $\mathcal{CA}$ perform the following steps:

1. $\mathcal{U}$ sets $PK_\mathcal{U} = g^{mk_\mathcal{U}}, R = \overline{g}^{mk_\mathcal{U}}$, where $mk_\mathcal{U} = \mathcal{H}_2(ID_\mathcal{U})$. It should be noted that $ID_\mathcal{U}$ is the unique identity of $\mathcal{U}$ (such as ID number, IP address or mailbox address).
2. $\mathcal{U}$ generates $\pi'_\mathcal{U} = PK_\mathcal{U}(\{mk_\mathcal{U}\} : PK_\mathcal{U} = g^{mk_\mathcal{U}} \wedge R = \overline{g}^{mk_\mathcal{U}})$. Then, $\mathcal{U}$ sends $PK_\mathcal{U}, R, \pi'_\mathcal{U}$ to $\mathcal{CA}$.
(3) If $\pi^1_\mu$ is valid, $CA$ calculates $SK_\mu = MK \cdot R = MK \cdot g^{mk_\mu}$. $CA$ selects $e \in \mathbb{Z}_p$ and calculates $A = (g_0 \cdot PK_\mu)^{i(e,\gamma)}$. $CA$ generates $\pi^1_{\mu, A} = PK\{(g) : A = (g_0 \cdot PK_\mu)^{i(e,\gamma)} \land w = h^e\}$ and returns $A, e, \pi^1_{\mu, A}$ to $U$.

(4) If $\pi^1_{\mu, A}$ is valid, $U$ keeps the public / private key pair $(PK_\mu, SK_\mu)$ and membership credential $cert_\mu = (A, e)$.

\textbf{AKeyGen.} $U$ and $AA$ do the following steps:

(1) $U$ constructs a zero-knowledge proof $\pi^1_\mu = PK\{(A, e, mk_\mu) : A = (g_0 \cdot g^{mk_\mu})^{i(e,\gamma)}\}$, and sends the requested attribute set $S$, public key $PK_\mu$ and $\pi^1_\mu$ to $AA$.

(2) If $\pi^1_\mu$ is valid, for each attribute $A_i \in S$, $AA$ generates $PK'_{A_i} = g^{\gamma_i(i(e,\mu))}$, $PK'_{A_i} = \hat{e}(g_0, g)^{i(e,\mu)}$, $SK_A = PK_{\mu}^{i(e,\mu)}$ and sets $PK_A = (PK'_{A_i}, PK'_{A_i})$. Finally, $AA$ returns $\{PK_A, SK_A\}_{\mu,i}^{|S|}$ to $U$.

(3) For $i = 1, \ldots, |S|$, $U$ verifies if the equation $\hat{e}(SK_\mu, PK'_{A_i}) = PK''_{A_i} \cdot \hat{e}(g_0, SK_\mu)$ holds. If so, $U$ stores $\{PK_A, SK_A\}_{\mu,i}^{|S|}$ as his / her attribute key.

\textbf{Authentication.} $U$ and $SP$ do the following steps:

(1) $U$ sends the authentication request $Req$ to $SP$.

(2) $SP$ defines the access structure $A = \vee_{i=1}^{n_\mu} (A_{i\mu} \land A_{i\mu})$. $SP$ selects a random number $nonce \in \mathbb{G}_T$. For $t = 1, \ldots, n$, $SP$ calculates $h_t = \mathcal{H}(nonce \parallel t), E_t = nonce \cdot \prod_{i=1}^{t} PK'_{A_i}^{\gamma_i(i(e,\mu))}, E'_t = \prod_{i=1}^{t} PK'_{A_i}^{\gamma_i(i(e,\mu))}$. $SP$ then sends the challenge ciphertext $CT = (CT_1, \ldots, CT_n)$ to $U$, where $CT_i = (E_t, E'_t, E''_t)$.

(3) For $t = 1, \ldots, n$, $U$ checks if there exists $S_t$ such that $S_t \subseteq S$. If so, $U$ calculates $nonce = E_{t} \cdot \hat{e}(E'_t, \prod_{i=1}^{t} SK_{A_i}) \cdot \hat{e}(E''_t, SK_{\mu})^{-1}$. At the same time, $U$ verifies whether the equation $E'_t = g^{\gamma_t(nonce||t)}$ holds. If so, $U$ generates $\pi^1_t = PK\{(A, e, mk_\mu) : A = (g_0 \cdot g^{mk_\mu})^{i(e,\gamma)}\}$. Finally, $U$ sends $\pi^1_t$ to $SP$. It should be noted that $U$ can also use the following method to obtain $nonce$. Specifically, $U$ executes $(TK_{S_t}, E'_t, E''_t) \leftarrow CreateTK(PK, SK_{\mu}, \{SK_A\}_{\mu,i}^{|S|}, CT)$. Then, $U$ sends $(PK, E'_t, E''_t, TK_{S_t})$ to $CS$, who returns $V \leftarrow Transform(PK, E'_t, E''_t, TK_{S_t})$. Finally, $U$ performs $nonce \leftarrow Decrypt_{out}(A, CT, V, RK_{S_t})$.

(4) If $\pi^1_t$ is valid, $SP$ accepts $U$’s authentication.

\textbf{CreateTK.} Taking $(PK, SK_\mu, \{SK_A\}_{\mu,i}^{|S|}, CT)$ as input, the algorithm performs the following steps:

(1) For $A = \vee_{i=1}^{n_\mu} (A_{i\mu} \land A_{i\mu})$, checks if there exists $S_t (r \in [1, n])$, such that $S_t \subseteq S$. If not, aborts.

(2) Selects $z \in \mathbb{Z}_p^*$, calculates $SK'_{\mu} = SK_{\mu}^{i/2}, \{SK'_{A_i}\}_{A_{i\mu}} = \{SK_{A_i}\}_{A_{i\mu}}^{i/2}$, and sets $TK_{S_t} = (SK'_{\mu}, \{SK'_{A_i}\}_{A_{i\mu}}, RK_{S_t} = z$.

(3) Outputs $TK_{S_t}, E'_t, E''_t$.

\textbf{Transform.} Taking $(PK, TK_{S_t}, E'_t, E''_t)$ as input, the algorithm performs the following operations: calculates $V = \hat{e}(E'_t, \prod_{i=1}^{t} SK'_{A_i}) \cdot \hat{e}(E''_t, SK_{\mu})^{-1}$, and outputs $V$.

\textbf{Decrypt_{out}.} The algorithm takes $(A = \vee_{i=1}^{n_\mu} (A_{i\mu} \land A_{i\mu}), CT, V, RK_{S_t})$ as input, the algorithm performs the following operations:

(1) Separates $CT$ into the form of $CT = (CT_1, \ldots, CT_n)$.

(2) Calculates $nonce' = E_z \cdot V^{RK_{S_t}}$. 

6
(3) Checks if \( E'_i = g^{h(nonce') \pi^i_u} \) is true. If so, outputs \( nonce' \), otherwise, outputs \( \perp \).

5. Details of the underlying zero-knowledge proofs
In the scheme of section 4, participants need to perform multiple zero knowledge proofs, in which \( \pi^i_u \) is constructed by the standard technique that proves “two discrete logarithms are equal”. \( \pi^i_{CA} \) is constructed with the technique of Canard et al. [16], which allows \( CA \) to prove to \( U \) that “the membership credential provided is valid”, and does not require \( U \) to perform bilinear pairing operations. \( \pi^i_2 \) and \( \pi^i_3 \) are constructed by using the technique of Camenich et al. [17], which avoids \( U \) performing bilinear pairing operations. The difference between the two is that the construction of \( \pi^i_3 \) requires the use of the random number \( nonce \) obtained by decryption.

6. Security analysis
In this section, we prove that the new scheme satisfies all the security properties defined in Section 3.

Correctness. Due to limited space, we focus on the following cases: (I) Assuming that \( U \)'s attribute set \( S \) satisfies the access structure \( \mathcal{A} \), \( U \) can always decrypt the challenging ciphertext \( CT \) successfully.

\[
E_i = nonce \cdot (\prod_{A \in S} PK^*_{A})^{h} = nonce \cdot \hat{e}(g, g)^{h \sum_{A \in S} y_{i}(A)_{A \in \mathcal{A}}} \\
\hat{e}(E'_i, \prod_{A \in S} SK^*_{A}) = \hat{e}(g, g)^{h \sum_{A \in S} m_{i}(A)_{A \in \mathcal{A}}} \cdot \hat{e}(g, \overline{g})^{h \sum_{A \in S} y_{i}(A)_{A \in \mathcal{A}}} \\
\hat{e}(E'_i, SK^*_{U}) = \hat{e}(g, g)^{h \sum_{A \in S} y_{i}(A)_{A \in \mathcal{A}}} \cdot \hat{e}(g, \overline{g})^{h \sum_{A \in S} m_{i}(A)_{A \in \mathcal{A}}}.
\]

According to formula (1), (2), (3), we have

\[
E_i \cdot \hat{e}(E'_i, \prod_{A \in S} SK^*_{A}) \cdot \hat{e}(E'_i, SK^*_{U})^{-1} = nonce \cdot \hat{e}(g, g)^{h \sum_{A \in S} y_{i}(A)_{A \in \mathcal{A}}} \cdot \hat{e}(g, \overline{g})^{h \sum_{A \in S} m_{i}(A)_{A \in \mathcal{A}}}. \\
\hat{e}(g, g)^{-h \sum_{A \in S} y_{i}(A)_{A \in \mathcal{A}}} \cdot \hat{e}(g, \overline{g})^{-h \sum_{A \in S} m_{i}(A)_{A \in \mathcal{A}}} = nonce
\]

(II) Suppose that \( U \)'s attribute set \( S \) satisfies the access structure \( \mathcal{A} \). Assuming that there is \( S_{i', s} \subseteq S, U \) can successfully decrypt the challenging ciphertext \( CT \). Specifically, \( U \) provides the transformation key \( TK_{i'} \) and part of ciphertext \( E'_i, E''_i \) to \( CSA \), and obtains the returned value \( V \), s.t.,

\[
V = \frac{\hat{e}(E'_i, \prod_{A \in S} SK'_{A})}{\hat{e}(E'_i, SK'_{U})} = \frac{\hat{e}(g, \overline{g})^{h \sum_{A \in S} y_{i}(A)_{A \in \mathcal{A}}} \cdot \hat{e}(g, \overline{g})^{\frac{1}{2} \sum_{A \in S} m_{i}(A)_{A \in \mathcal{A}}}}{\hat{e}(g, \overline{g})^{h \sum_{A \in S} y_{i}(A)_{A \in \mathcal{A}}} \cdot \hat{e}(g, \overline{g})^{\frac{1}{2} \sum_{A \in S} m_{i}(A)_{A \in \mathcal{A}}}} = \hat{e}(g, \overline{g})^{h \sum_{A \in S} y_{i}(A)_{A \in \mathcal{A}}}.
\]

According to formula (5), we have

\[
E_i \cdot V^{PK_{i'}} = nonce' \cdot \hat{e}(g, g)^{h \sum_{A \in S} y_{i}(A)_{A \in \mathcal{A}}} \cdot \hat{e}(g, g)^{\frac{1}{2} \sum_{A \in S} m_{i}(A)_{A \in \mathcal{A}}} = nonce'.
\]

Anonymity & Unlinkability. These properties are obviously satisfied because \( U \) only provides \( \pi^i_u \) to \( SP \) in the Authentication protocol. According to the simulatability of \( \pi^i_u \), it is impossible for \( SP \) to obtain any information about the real identity of users according to \( \pi^i_u \). Similarly, \( SP \) cannot associate the user’s access behavior with his/her previous behaviour of registration or key application through the collusion with \( CA \) and \( AA \).
Ciphertext security & verifiability. We can define a selective CPA security experiment performed by challenger $C$ and attacker $A$. If $A$ wins in the experiment, it is equivalent to breaking through the CPA security of the underlying DABE scheme. Similarly, we can also define a verifiability experiment performed by challenger $C$ and attacker $A$. If $A$ wins in the experiment, it is equivalent to breaking the collision-resistant property of hash function $h_i$.

7. Performance Analysis

First, we compare the properties of the new scheme with those of the existing similar schemes in table 1. First of all, except the new scheme and the scheme in [4], other schemes only consider one attribute authority. In the authentication phase, only the new scheme and scheme in [8] allow most of the computation tasks to be outsourced to cloud server. In terms of the underlying access policy types, each scheme is different. It should be acknowledged that policies based on LSSS (Linear Secret Sharing Scheme) and attribute tree are more flexible. In the scheme in [7], users are required to provide a subset of attributes to the verifier in the authentication phase, so it does not satisfy attribute anonymity. In addition, both the new scheme and the scheme in [8] scheme allow users to verify the attribute key provided by AA.

```
| Scheme | Type of AA | Outsourcing Computation | Access Policy | Attribute Anonymity | Verifiability |
|--------|------------|--------------------------|---------------|---------------------|---------------|
| [1]    | Centralized | No                        | Threshold     | Yes                 | No            |
| [2]    | Centralized | No                        | LSSS          | Yes                 | No            |
| [3]    | Centralized | No                        | LSSS          | Yes                 | No            |
| [4]    | Distributed | No                        | Threshold     | Yes                 | No            |
| [5]    | Centralized | No                        | Attribute Tree| Yes                 | No            |
| [7]    | Centralized | No                        | Attribute Tree| No                  | No            |
| [8]    | Centralized | Yes                       | LSSS          | Yes                 | Yes           |
| Ours   | Distributed | Yes                       | DNF           | Yes                 | Yes           |
```

Next, we analyze the computation efficiency of the new scheme. Let $|S|$ represent the size of user’s attribute set. Let $Exp(G)$ represent one exponential operation performed in group $G$. Let $Exp(G_r)$ represent one exponential operation performed in group $G_r$. Let $P$ represent one bilinear pairing operation. In the new scheme, the computation cost of $U$ in the Registration protocol, the AKeyGen protocol and the Authentication protocol is $6Exp(G)(2|S|+1)Exp(G)+|S|P$ and $3Exp(G)+Exp(G_r)$ respectively. Obviously, the operation cost of $U$ in the authentication stage is more satisfactory. Although the operation cost of $U$ in this stage is linear with that of $|S|$, it is the price that $U$ must pay to perform the verification on the attribute key.

8. Conclusion

In order to overcome the shortcomings of existing attribute-based authentication schemes, we propose a new multi-authority attribute authentication scheme, which takes DABE as its core underlying module. The new scheme not only meets the requirements of service providers to implement fine-grained access control on cloud resources, but also allows multiple authorities to cooperate to maintain the attribute universe. The future research directions include further improving the computation efficiency of the user, upgrading the underlying access policy to LSSS type, and so on.

Acknowledgments

This work was supported in part by the Project of Shandong Province Higher Educational Science and Technology Program under Grant J17KA081, and in part by the Undergraduate Education Reform.
Project of Higher Education in Shandong Province under Grant M2018X245, in part by the Scientific Research Project of Shandong Youth University of Political Science under Grant XJPY2021, and in part by the Project of Educational Reform and Research in Shandong Youth University of Political Science under Grant JG201905.

References

[1] Zhou, J., Cao, Z. (2012) PSCPA: Patient self-controllable privacy-preserving cooperative authentication in distributed m-healthcare systems. https://eprint.iacr.org/2012/044.pdf

[2] Lian, Y., Huang, X., Mu, Y., et al. (2014) SA 3: Self-adaptive anonymous authentication for dynamic authentication policies. Future Generation Computer Systems, 30: 133-139.

[3] Li, M., Huang, X., Liu, J. K., et al. (2015) Cooperative attribute-based access control for enterprise computing system. International Journal of Embedded Systems, 7: 191-202.

[4] Li, J., Chen, X., Huang, X. (2015) New attribute-based authentication and its application in anonymous cloud access service. International Journal of Web and Grid Services, 11: 125-141.

[5] Yang, H., Oleshchuk, V. A. (2015) An efficient traceable attribute-based authentication scheme with one-time attribute trees. In: Proceedings of NordSec 2015. Stockholm. pp. 123-135.

[6] Bethencourt, J., Sahai, A., Waters, B. (2007) Ciphertext-policy attribute-based encryption. In: Proceedings of IEEE symposium on security and privacy (S&P 2007). Oakland. pp. 321-334.

[7] Yang, H., Oleshchuk, V. A. (2015) A dynamic attribute-based authentication scheme. In: Proceedings of C2SI 2015. Rabat. pp. 106-118.

[8] Liu, X., Xu, Q., Zhang, B., et al (2018) k-times attribute-based authentication scheme using direct anonymous attestation. Journal on Communications, 39: 113-133.

[9] Zhang, Y., Zheng, D., Deng, R. H. (2018) Security and privacy in smart health: Efficient policy-hiding attribute-based access control. IEEE Internet of Things Journal, 5:2130-2145.

[10] Li, J., Wang, Y., Zhang, Y., et al (2017) Full verifiability for outsourced decryption in attribute based encryption. IEEE Transactions on Services Computing. Doi: 10.1109/TSC.2017.2710190

[11] Yao, X., Chen, Z., Tian, Y. (2015) A lightweight attribute-based encryption scheme for the Internet of Things. Future Generation Computer Systems, 49: 104-112.

[12] Liu, X., Xu, Q., Zhang, B. (2016) Lightweight multi-coupon system for multi-merchant environments with DAA. Journal on Communications, 37: 30-45.

[13] Wang, H., He, D., Han, J. (2017) VOD-ADAC: Anonymous distributed fine-grained access control protocol with verifiable outsourced decryption in public cloud. IEEE Transactions on Services Computing, doi: 10.1109/TSC.2017.2687459

[14] Müller, S., Katzenbeisser, S., Eckert, C. (2009) Distributed attribute-based encryption. In: Proceedings of ICISC 2008. Seoul. pp. 20-36.

[15] Camenisch, J., Chaabouni, R., Shelat, A. (2008) Efficient protocols for set membership and range proofs. In: Proceedings of ASIACRYPT 2008. Melbourne. pp. 234-252.

[16] Canard, S., Coisel, I., Jambert, A., et al. (2014) New results for the practical use of range proofs. In: Proceedings of EuroPKI 2013, Egham. pp. 47–64.

[17] Camenisch, J., Drijvers, M., Hajny, J. (2016) Scalable revocation scheme for anonymous credentials based on n-times unlinkable proofs. In: Proceedings of ACM WPES 2016, Vienna. pp. 123-133.