Fine-Grained Information Flow Control Using Attributes


Published in:
Information Sciences

Document Version:
Peer reviewed version

Queen's University Belfast - Research Portal:
Link to publication record in Queen's University Belfast Research Portal

Publisher rights
Copyright 2019 Elsevier. This manuscript is distributed under a Creative Commons Attribution-NonCommercial-NoDerivs License (https://creativecommons.org/licenses/by-nc-nd/4.0/), which permits distribution and reproduction for non-commercial purposes, provided the author and source are cited.

General rights
Copyright for the publications made accessible via the Queen's University Belfast Research Portal is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The Research Portal is Queen's institutional repository that provides access to Queen's research output. Every effort has been made to ensure that content in the Research Portal does not infringe any person's rights, or applicable UK laws. If you discover content in the Research Portal that you believe breaches copyright or violates any law, please contact openaccess@qub.ac.uk.

Download date:06. Mar. 2021
Fine-Grained Information Flow Control Using Attributes

Jinguang Han\textsuperscript{a,*}, Liqun Chen\textsuperscript{b}, Willy Susilo\textsuperscript{c}, Xinyi Huang\textsuperscript{d}, Aniello Castiglione\textsuperscript{e}, Kaitai Liang\textsuperscript{b}

\textsuperscript{a}Centre for Secure Information Technologies (CSIT), Queen’s University Belfast, Belfast, Northern Ireland, BT3 9DT, United Kingdom
\textsuperscript{b}Surrey Center for Cyber Security, Department of Computer Science, University of Surrey, Guildford, Surrey, GU2 7XH, United Kingdom
\textsuperscript{c}Computing and Information Technology, University of Wollongong, Wollongong, NSW 2522, Australia
\textsuperscript{d}Fujian Provincial Key Laboratory of Network Security and Cryptology, School of Mathematics and Computer Science, Fujian Normal University, Fuzhou, Fujian 350108, China
\textsuperscript{e}Department of Computer Science, University of Salerno, Via Giovanni Paolo II, 132, I-84084, Fisciano (SA), Italy

Abstract

Information flow control (IFC) mechanisms regulate where information is allowed to travel. To enhance IFC, access control encryption (ACE) was proposed where both the no write-down rule and the no read-up rule are supported. Nevertheless, there are still two issues: (1) how to determine whether a communication request should be permitted or denied was not considered; (2) the commutation cost is linear with the number of receivers.

Attribute-based system (ABS) can implement one-to-many communication and fine-grained access policies. In this paper, a new IFC scheme is proposed by combing ACE with ABS. Our scheme provides the following features: (1) IFC policies are defined over a universe set of attributes; (2) the computation cost to determine whether a communication request should be permitted or denied is constant, instead of linear with the number of required

\*Corresponding author. Tel.: +44 28 9097 1819; fax: +44 28 9097 1819
Email addresses: j.han@qub.ac.uk (Jinguang Han), liqun.chen@surrey.ac.uk (Liqun Chen), wsusilo@uow.edu.au (Willy Susilo), xyhuang81@gmail.com (Xinyi Huang), castiglione@ieee.org (Aniello Castiglione), k.liang@surrey.ac.uk (Kaitai Liang)

Preprint submitted to Information Science January 25, 2019
attributes or receivers; (3) weak attribute privacy is achieved; (4) fine-grained access policies on encrypted data are supported; (5) the communication cost is linear with the number of required attributes and is independent of the number of receivers. To the best of our knowledge, it is the first IFC scheme enforced by using attributes.

Keywords: Information flow control, Attribute-based system, Access control encryption, Security, Privacy

1. Introduction

In cloud computing [1, 2], users can share their information via network whenever and wherever. However, lots of evidence, for example data releasing, data theft, etc., show that data confidentiality should be enforced. To protect data confidentiality, various encryption schemes with distinct properties have been proposed, including symmetric encryption [42, 43], identity-based encryption (IBE) [7, 37, 52], attribute-based encryption (ABE) [22, 49], searchable encryption [8, 18], functional encryption (FE) [9, 50], fully homomorphic encryption (FHE) [21], etc. Furthermore, some access control schemes [10, 17, 55] were proposed to ensure that only authorized users can access protected data. Nevertheless, in above schemes, there is no limitation on senders’ sides.

Information flow control (IFC) [16, 48] can be used to regulate the travel of information among objects. In an IFC scheme, security requires that an entity with higher security level cannot send data to an entity with lower security level (no write-down rule) and an entity with lower security level cannot directly read a file owned by an entity with higher security level (no read-up rule) [48]. Hence, encryption schemes realize the no read-up rule, but do not consider the no write-down rule. However, no write-down rule is very important since malicious senders may release sensitive information to illegal receivers or send meaningless messages to paralysis the Internet. For example, in bank systems, tellers should not read the information held by the manager; meanwhile, the manager cannot release personal information of customers to tellers.

Introduced by Damgård et al. [14], access control encryption (ACE) is a new public-key encryption scheme where all communication between senders and receivers must go through a third party called sanitizer. In [14], a sender cannot write a message which he/she is not permitted to write and a
receiver cannot read a message which he/she is unauthorized to read. Hence, an ACE scheme supports both the no read-up rule and the no write-down rule, and can be used to enforce IFC. Some ACE schemes were proposed [4, 28, 54], but none of them addressed how the sanitizer can determine whether a communication request should be permitted or denied.

Attribute-based encryption (ABE) [47] can implement flexible and fine-grained access policies, and support one-to-many communication. For example, suppose that Alice holds a set of attributes \( A = \{\text{Profession}="\text{Student"}, \text{Department}="\text{Computer Science"}, \text{Sex}="\text{Females"}, \text{Age}=19\} \) and Bob holds a set of attributes \( B = \{\text{Profession}="\text{Student"}, \text{Department}="\text{Computer Science"}, \text{Sex}="\text{Male"}, \text{Age}=18\} \). If a message is encrypted using an access structure \( \mathcal{M} = (\text{Profession}="\text{Student"} \land \text{Department}="\text{Computer Science"}) \), both Alice and Bob can decrypt the ciphertext. Meanwhile, nobody can detect who is the real receiver/decryptor since both Alice’ attributes and Bob’ attributes satisfy the access structure.

In this paper, to implement flexible information flow policies and enforce IFC, a fine-grained IFC using attributes scheme is proposed.

### 1.1. Related Work

In this section, we review the work related to our scheme, including attribute-based systems and ACE schemes.

#### 1.1.1. Attribute-Based Systems

Introduced by Sahai and Waters [49], ABE is a variant of identity-based encryption (IBE) [7, 33, 52]. Goyal et al. [22] classified ABE schemes into two types: key-policy ABE (KP-ABE) [25, 26, 45, 49] and ciphertext-policy ABE (CP-ABE) [5, 24, 30, 58]. In a CP-ABE scheme, users’ secret keys are labelled with sets of attributes; while ciphertexts are associated with access structures. A user can decrypt a ciphertext if and only if his/her attributes satisfies the access structure associated to the ciphertext. On the converse, in a KP-ABE scheme, users’ keys are associated with access structures; while ciphertexts are labelled with sets of attributes. A user can decrypt a ciphertext if and only if the attributes labelled to the ciphertext satisfy the access structure embedded in his/her secret key. Waters [58] proposed a CP-ABE scheme where a linear secret sharing scheme matrix \( \mathcal{M} \) is used to express any monotonic access structure. A user can decrypt a ciphertext if and only if his/her attributes satisfy the access structure \( \mathcal{M} \).
This scheme is efficient and has been used to construct various application systems, including verifiable outsourced computing [29, 34], decentralized ABE [24], etc.

To reduce trust on the central authority (CA) in [49], Chase [11] proposed a multi-authority ABE (MA-ABE) scheme where the CA issues secret keys to multiple attribute authorities, and each user needs to obtain secret keys from both the CA and attribute authorities. To prevent collusion attacks, a user’s secret keys obtained from multiple authorities must be bound with her global identifier (GID). However, multiple authorities can collude to profile a user’s attributes by tracing her GID. Chase and Chow [12] proposed a privacy-preserving MA-ABE scheme. In the scheme [12], a secure two party computation technique was adopted to enable users to obtain secret keys from multiple attribute authorities without revealing their GIDs. A trusted CA is not required in [12], but multiple attribute authorities must interact to initial the system. Lewko and Waters [30] proposed a decentralized ABE scheme where the CA is not required and multiple attribute authorities can work independently without any interactions. Han et al. [24, 25] proposed two privacy-preserving decentralized ABE schemes where users can obtain secret keys from multiple attribute authorities without exposing their GIDs. There are some other ABE schemes with special features, including ABE with outsourced computing scheme [23, 32], ABE supporting revocation schemes [53, 57, 59], hierarchical ABE schemes [15, 38, 56], etc.

Maji et al. [40] first proposed an attribute-based signature scheme where a signer can sign a message with a predicate which her attributes satisfy. A signature only reveals that a signer whose attributes satisfy the predicate signed the message and hides the attributes used to sign the message.

Gagné et al. [41] and Shahandashti et al. [51] proposed attribute-based signature for threshold policies schemes. In [41] and [51], two notions of attribute privacy were considered: weak attribute privacy and full attribute privacy. In the weak attribute privacy notion, a signature is allowed to release the threshold number of a signer’s attributes, while in the full attribute privacy notion, a signature can only reveal its validity to the verifier.

To reduce trust on the CA, Li et al. [31] proposed an attribute-based signature with multiple attribute authorities scheme where each user needs to obtain secret keys from both the CA and attribute authorities. Okamoto and Takashima [44] introduced the concept of decentralized attribute-based signature where both the CA and trusted setup are not required. Furthermore, Chen et al. [13] proposed outsourced attribute-based signature schemes to
reduce the computation burden on signers’ sides.

Cloud computing provides users with on-demand computing services, and users can outsource their personal data to cloud servers. Hence, access control on outsourced data attracts lots of attentions from both academic research communities and industries. Since its can implement fine-grained access structures and privacy-preserving authentications, attribute-based systems have been applied into cloud computing to enforce access control and protect data confidentiality [35, 36, 60].

1.1.2. Access Control Encryption

Damg˚ ard et al. [14] first introduced the concept of ACE. In an ACE scheme, there are four entities: a key generator, some senders, some receivers and a sanitizer. The key generator issues secret keys to senders, receivers and the sanitizer. A sender encrypts a plaintext using his/her secret key and the receiver’s public key, and sends the ciphertext to the sanitizer. After receiving a ciphertext, the sanitizer uses his secret key to sanitize the ciphertext and send the sanitized ciphertext to the receiver. Finally, the receiver can use his/her secret key to decrypt the ciphertext if he/she is allowed to read the plaintext. Notably, the sanitizer cannot know the identities of both the senders and receivers.

In [14], four ACE schemes were proposed. The first two schemes are for single identity where only one receiver can be communicated each time. The third scheme is for multiple identities. The idea is to run the first scheme \( n \) times where \( n \) is the number of receivers. Hence, the complexity of the third scheme is linear with \( n \). To reduce the complexity in the ACE for multi-identity scheme, the fourth scheme was proposed. This scheme was based on the indistinguishability obfuscation (\( \mathcal{IO} \)) [20] and has polylogarithmic complexity in the number \( n \). In [14], the authors also left two open problems: (1) how to construct practical ACE schemes from noisy and post-quantum assumptions; (2) how to construct efficient sublinear ACE schemes even for restricted classes of predicates.

To resolve the first problem left in [14], Tang et al. [54] consturcted an ACE scheme based on the Learning with Error (LWE) assumption. In [54], the authors first proposed an ACE scheme for one-identity (1-ACE), and then extended it to support multiple identities by running the 1-ACE scheme multiple times.

Fuchsbauer et al. [19] answered the second problem left in [14] by proposing an ACE scheme for a class of predicates, including equality, comparison,
interval membership, etc. The authors first gave a generic construction of ACE schemes for equality, and then proposed a concrete construction. Furthermore, the authors discussed how to design ACE schemes for disjunction of equalities. In [19], the sanitizer does not need a secret key to sanitize ciphertexts.

Badertscher et al. [4] addressed security issues in ACE schemes and presented a general construction of ACE schemes with the following features: (1) it is secure against chosen ciphertext attacks (CCA); (2) a (unsanitized) ciphertext cannot reveal more in a policy-violating manner than the information contained in the ciphertext; (3) a legal receiver is guaranteed to receive a message.

Kim and Wu [28] proposed a generic construction of ACE schemes for general policies where functional encryption supporting randomized functionalities is adopted. Furthermore, they extended their scheme to support dynamic and more fine-grained access control policies.

In existing ACE schemes, the sanitizer sanitizes all ciphertexts from senders without any authentication. Hence, the sanitizer can prevent illegal receivers from reading plaintexts, but cannot prevent illegal senders from sending messages to receivers.

1.2. Our Contributions

To enforce IFC and implement flexible policies, a fine-grained IFC using attributes scheme is proposed in this paper. In our scheme, prior to sanitizing a ciphertext, the sanitizer first checks whether the sender’s attributes and the receiver’s attributes satisfy the IFC policy. If it is, the sanitizer sanitizes the ciphertext and sends the sanitized ciphertext to the receiver; otherwise, the communication request is stopped. Our scheme provides the following features: (1) IFC policies are defined over a universal set of attributes; (2) the computation cost to determine whether a communication request should be permitted or denied is constant, instead of linear with the number of required attributes or receivers; (3) it can provide weak attribute privacy, namely for each communication, only subsets of the sender’s attributes and the receiver’s attributes are released to the sanitizer, instead of all attributes; (4) fine-grained access structures on encrypted data are supported; (5) the communication cost is linear with the number of required attributes and is independent of the number of receivers. To the best of our knowledge, it is the first IFC scheme enforced by using attributes.
In Table 1, we compare our scheme with related schemes in terms of complexity assumption, ciphertext size, predicate, supporting multi-identity method, sanitizer’s key size. $n$ and $N$ stand for the number of identities and required attributes, respectively. $\cdots$ denotes inappropriate/not required.

1.3. Paper Organization

The remainder of this paper is organised as follows. In Section 2, the preliminaries used throughout this paper are introduced. A fine-grained IFC using attributes scheme is proposed in Section 3. The implementation of our scheme is described in Section 4. In Section 5, the security of the proposed scheme is analyzed. Finally, Section 6 concludes this paper.

2. Preliminaries

In this section, the preliminaries used throughout this paper are introduced. All syntaxes are summarized in Table 2.

2.1. Formal Definition

Fig. 1 describes the model of our scheme. In our scheme, there are four entities: central authority CA which is a trusted party, sender S, receiver R and sanitizer San which is a semi-trusted party. CA sets the systems, specifies an IFC policy $P$ defined over a universal set of attributes $U$, and issues secret keys $SK_u$ to users (receivers and senders) according to their
Table 2: Summary of Syntaxes

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>U</td>
<td>User</td>
</tr>
<tr>
<td>S</td>
<td>Sender</td>
</tr>
<tr>
<td>R</td>
<td>Receiver</td>
</tr>
<tr>
<td>San</td>
<td>Sanitizer</td>
</tr>
<tr>
<td>M</td>
<td>A message</td>
</tr>
<tr>
<td>ID_U</td>
<td>U’s identity</td>
</tr>
<tr>
<td>ID_S</td>
<td>S’s identity</td>
</tr>
<tr>
<td>ID_R</td>
<td>R’s identity</td>
</tr>
<tr>
<td>SK_U</td>
<td>U’s secret key</td>
</tr>
<tr>
<td>SK_S</td>
<td>S’s secret key</td>
</tr>
<tr>
<td>SK_R</td>
<td>R’s secret key</td>
</tr>
<tr>
<td>CA</td>
<td>Central authority</td>
</tr>
<tr>
<td>M</td>
<td>An access structure</td>
</tr>
<tr>
<td>1^ℓ</td>
<td>A security parameter</td>
</tr>
<tr>
<td>U</td>
<td>A universal of attributes</td>
</tr>
<tr>
<td>A_U</td>
<td>The set of U’s attributes</td>
</tr>
<tr>
<td>A_S</td>
<td>The set of S’s attributes</td>
</tr>
<tr>
<td>A_R</td>
<td>The set of R’s attributes</td>
</tr>
<tr>
<td>A̅_U</td>
<td>A subset of U’s attributes</td>
</tr>
<tr>
<td>A̅_S</td>
<td>A subset of S’s attributes</td>
</tr>
<tr>
<td>A̅_R</td>
<td>A subset of R’s attributes</td>
</tr>
<tr>
<td>⍵(ℓ)</td>
<td>A negligible function in ℓ</td>
</tr>
<tr>
<td>B(G(1^ℓ))</td>
<td>A bilinear group generator</td>
</tr>
<tr>
<td>A</td>
<td>= M</td>
</tr>
<tr>
<td>A ≠ M</td>
<td>The set A does not satisfy M</td>
</tr>
<tr>
<td>Ω</td>
<td>A set of attributes associated with M</td>
</tr>
<tr>
<td>x R X</td>
<td>x is randomly selected from the set X</td>
</tr>
<tr>
<td>Y(x) → y</td>
<td>y is obtained by running the algorithm Y(·) with input x</td>
</tr>
</tbody>
</table>
attributes and a secret-public key pair \((ssk, spk)\) to \(\text{San}\). All communication between \(S\) and \(R\) must pass through \(\text{San}\). To send a message \(M\) to \(R\), \(S\) selects an access structure \(\mathcal{M}\), and encrypts \(M\) under \(\mathcal{M}\). Suppose that \(\mathcal{M}\) is associated with a set of attributes \(\Omega\) and is implicitly included in the ciphertext \(CT\). \(S\) selects a subset of his attributes \(\tilde{A}_S\) with \(\mathbb{P}(\tilde{A}_S, \Omega) = 1\) and computes an authentication head \(Hd\) using his/her secret key. \(S\) sends \((CT, Hd, \tilde{A}_S)\) to \(\text{San}\). Subsequently, \(\text{San}\) checks \(\mathbb{P}(\tilde{A}_S, \Omega) \geq 1\) and validates \(Hd\). If they are correct, \(\text{San}\) sanitizes the ciphertext \(CT\) and publishes the sanitized ciphertext \(CT'\); otherwise \(\text{San}\) denies the communication request. Finally, \(R\) can decrypt \(CT'\) if and only if his/her attributes \(A_R\) satisfies the access structure \(\mathcal{M}\) \((A_R \models \mathcal{M})\).

A fine-grained IFC using attributes scheme is formalized as follows:

**Setup** \((1^\ell) \rightarrow (msk, (ssk, spk), params, U, \mathbb{P})\). This algorithm takes as input a security parameter \(1^\ell\), and outputs a master secret key \(msk\), a sanitizer’s secret-public key pair \((ssk, spk)\), the public parameters \(params\), a universal set of attributes \(U\) and an IFC policy \(\mathbb{P} : U \times U \rightarrow \{0, 1\}\).

**KeyGen** \((msk, params, A_U) \rightarrow SK_U\). This algorithm takes as input the master secret key \(msk\), the public parameters \(params\) and a set of attributes \(A_U \subseteq U\), and outputs a secret key \(SK_U\).
Enc\((params, spk, M, \mathcal{M}, SK_{S}) \rightarrow (\tilde{A}_S, Hd, CT)\). This algorithm takes as input the public parameters \(params\), the sanitizer’s public key \(spk\), a message \(M\), an access structure \(\mathcal{M}\) associated with a set of attributes \(\Omega \subseteq U\) and the sender’s secret key \(SK_{S}\), and outputs the attribute \(\tilde{A}_S (\tilde{A}_S \subseteq A_S)\) required by the no write-down rule, an authentication header \(Hd\) and a ciphertext \(CT\) which is bound with \(Hd\). Suppose that the access structure \(\mathcal{M}\) is implicitly included in \(CT\).

Sanitizer\((params, ssk, \tilde{A}_S, \Omega, Hd, CT) \rightarrow \{(1, CT'), (0, \bot)\}\). This algorithm consists of the following two sub-algorithms:

1. \(\text{Authen}(P, Hd, \tilde{A}_S, \Omega) \rightarrow \{0, 1\}\). This algorithm takes as input the IFC policy \(P\), the authentication head \(Hd\), the required sender’s attributes \(\tilde{A}_S\) and the attributes \(\Omega\) associated with the access structure \(\mathcal{M}\), and outputs 1 if \(P(\tilde{A}_S, \Omega) = 1\) and \(Hd\) is valid; otherwise it outputs 0 to indicate failure.

2. \(\text{Sanitizing}(params, ssk, CT) \rightarrow CT'\). If \(\text{Authen}(P, Hd, \tilde{A}_S, \tilde{A}_R) \rightarrow 1\), this algorithm takes as input the public parameters \(params\), the sanitizer’s secret key \(ssk\) and the ciphertext \(CT\), and outputs a sanitized ciphertext \(CT'\).

Dec\((params, CT', SK_R) \rightarrow M\). This algorithm takes as input the public parameters \(params\), the sanitized ciphertext \(CT'\) and the receiver’s secret key \(SK_R\), and outputs the plaintext \(M\) if \(A_R \models \mathcal{M}\).

\textbf{Definition 1.} A fine-grained information flow control using attributes scheme is correct if and only if

\[
\begin{align*}
Pr_{\text{Setup}(1^\ell)} \rightarrow (msk, (ssk, spk), params, U, \mathbb{P}); \\
\text{Authen}(P, Hd, \tilde{A}_S, \Omega) \rightarrow 1 \\
\text{KeyGen}(msk, params, A_R) \rightarrow SK_R; \\
\text{KeyGen}(msk, params, A_S) \rightarrow SK_S; \\
\text{Enc}(params, spk, M, \mathcal{M}, SK_{S}) \rightarrow (\tilde{A}_S, Hd, CT); \\
\mathbb{P}(A_R, \tilde{A}_S) = 1
\end{align*}
\]

and

\[
\begin{align*}
Pr_{\text{Dec}(params, CT', SK_R) \rightarrow M} \\
\text{Setup}(1^\ell) \rightarrow (msk, (ssk, spk), params, U, \mathbb{P}); \\
\text{KeyGen}(msk, params, A_R) \rightarrow SK_R; \\
\text{KeyGen}(msk, params, A_S) \rightarrow SK_S; \\
\text{Enc}(params, spk, M, \mathcal{M}, SK_{S}) \rightarrow (\tilde{A}_S, Hd, CT); \\
\text{Sanitizing}(params, ssk, CT) \rightarrow CT'; \\
A_R \models \mathcal{M}
\end{align*}
\]

= 1.
where the probability is taken over random bits consumed by all algorithms.

2.2. Security Model

The security models of fine-grained IFC using attributes schemes are defined by using the following two games executed between a challenger \( \mathcal{C} \) and an adversary \( \mathcal{A} \).

**No Read-Up Rule/Sender’s Security Game.** This game is used to formalize the no read-up rule/sender’s security, namely only legal receivers can read the plaintext. This game is as follows:

**Setup.** \( \mathcal{C} \) runs \( \text{Setup}(1^\ell) \rightarrow (\text{msk},(\text{ssk},\text{spk}),\text{params},\mathcal{U},\mathcal{P}) \) and sends \( (\text{spk}, \text{params}, \mathcal{U}, \mathcal{P}) \) to \( \mathcal{A} \).

**Phase 1.** \( \mathcal{A} \) can adaptively make the following queries multiple times.

- **Key Generation Query.** \( \mathcal{A} \) submits a set of attributes \( \mathcal{A}_U \) (including senders’ and receivers’ attributes). \( \mathcal{C} \) runs \( \text{KeyGen}(\text{msk},\text{params},\mathcal{A}_U) \rightarrow \text{SK}_U \), and sends \( \text{SK}_U \) to \( \mathcal{A} \). Let \( Q_K \) be a family consisting of the sets of attributes selected by \( \mathcal{A} \) to query secret keys.

- **Encryption Query.** \( \mathcal{A} \) submits a message \( M \), a set of attributes \( \tilde{\mathcal{A}}_S \) and an access structure \( \mathcal{M} \). \( \mathcal{C} \) runs \( \text{KeyGen}(\text{msk},\text{params},\tilde{\mathcal{A}}_S) \rightarrow \text{SK}_S \) and \( \text{Enc}(\text{params},\text{spk},M,\mathcal{M},\text{SK}_S) \rightarrow (\tilde{\mathcal{A}}_S,\text{Hd},\text{CT}) \). \( \mathcal{C} \) sends the ciphertext \( (\tilde{\mathcal{A}}_S,\text{Hd},\text{CT}) \) to \( \mathcal{A} \). Let \( Q_E \) be a set consisting of \( (\tilde{\mathcal{A}}_S,\text{CT}) \).

- **Sanitizer Query.** \( \mathcal{A} \) submits \( (\tilde{\mathcal{A}}_S,\Omega,\text{Hd},\text{CT}) \). \( \mathcal{C} \) runs \( \text{Authen}(\mathcal{P},\text{Hd},\tilde{\mathcal{A}}_S,\Omega) \). If \( \text{Authen}(\mathcal{P},\text{Hd},\tilde{\mathcal{A}}_S,\Omega) = 0 \), \( \mathcal{C} \) quits; otherwise, \( \mathcal{C} \) runs \( \text{Sanitizing}(\text{params},\text{ssk},\text{CT}) \rightarrow \tilde{\text{CT}} \) and sends \( \tilde{\text{CT}} \) to \( \mathcal{A} \).

**Challenge.** \( \mathcal{A} \) submits two messages \( M_1 \) and \( M_2 \) with same length, an access structure \( \mathcal{M}^* \) and a set of attributes \( \tilde{\mathcal{A}}_{S^*} \) with the limitation \( \mathcal{A}_U \notin \mathcal{M}^* \) for all \( \mathcal{A}_U \in Q_K \). \( \mathcal{C} \) flips an unbiased coin with \( \{0,1\} \) and obtains a bit \( b \in \{0,1\} \). \( \mathcal{C} \) runs \( \text{KeyGen}(\text{msk},\text{params},\tilde{\mathcal{A}}_{S^*}) \rightarrow \text{SK}_{S^*} \), \( \text{Enc}(\text{params},\text{spk},M_b,\mathcal{M}^*,\text{SK}_{S^*}) \rightarrow (\tilde{\mathcal{A}}_{S^*},\text{Hd}^*,\text{CT}^*) \) and \( \text{Sanitizing}(\text{params},\text{ssk},\text{CT}) \rightarrow \tilde{\text{CT}}^* \). \( \mathcal{C} \) sends the challenged ciphertext \( \tilde{\text{CT}}^* \) to \( \mathcal{A} \).

**Phase 2.** \( \mathcal{A} \) can adaptively make the following queries multiple times.

- **Key Generation Query.** This query is the same as in **Phase 1** with the restrict that \( \mathcal{A}_U \notin \mathcal{M}^* \).

- **Encryption Query.** This query is the same as in **Phase 1**.
Sanitizer Query. This is the same as in Phase 1.

**Guess.** \( A \) outputs his guess \( b' \) on \( b \). \( A \) wins the game if \( b' = b \).

**Definition 2.** A fine-grained information flow control using attributes scheme is \( \epsilon(\ell) \) no read-up/sender secure if for all probable polynomial-time (PPT) adversaries \( A \) can win the above game with the advantage

\[
\text{Adv}_A = \left| \Pr[b' = b] - \frac{1}{2} \right| \leq \epsilon(\ell).
\]

We say that a fine-grained information flow control using attributes scheme is no read-up/sender selectively secure if an initialization phase Initial is added before the Setup phase, where \( A \) must submit an access structure \( \mathcal{M}^* \) prior to seeing the public parameters \( \text{params} \).

**No Write-Down Rule/Receiver’s Security Game.** This game is used to formalize the receiver’s security, namely only the legal senders can send messages to the receivers. This game is as follows:

**Setup.** \( C \) runs \( \text{Setup}(1^\ell) \rightarrow (\text{msk}, (\text{ssk}, \text{spk}), \text{params}, \mathcal{U}, \mathcal{P}) \) and sends \( (\text{spk}, \text{params}, \mathcal{U}, \mathcal{P}) \) to \( A \).

**Phase 1.** \( A \) can adaptively make the following queries multiple times.

**Key Generation Query.** \( A \) submits a set of attributes \( A_U \) (including senders’ and receivers’ attributes). \( C \) runs \( \text{KeyGen}(\text{msk}, \text{params}, A_U) \rightarrow S\text{K}_U \), and sends \( S\text{K}_U \) to \( A \). Let \( Q_K \) be a family consisting of the sets of attributes selected by \( A \) to query secret keys.

**Encryption Query.** \( A \) submits a messages \( M \), a set of attributes \( \tilde{A}_S \) and an access structure \( \mathcal{M} \) associated with a set of attributes \( \Omega \). \( C \) runs \( \text{KeyGen}(\text{msk}, \text{params}, \tilde{A}_S) \rightarrow S\text{K}_S \) and \( \text{Enc}(\text{params}, \text{spk}, M, \mathcal{M}, S\text{K}_S) \rightarrow (\tilde{A}_S, H\text{d}, CT) \). \( C \) sends the ciphertext \( (\tilde{A}_S, H\text{d}, CT) \) to \( A \). Let \( Q_E \) be a set consists of \( (\tilde{A}_S, CT) \).

**Sanitizer Query.** \( A \) submits \( (\tilde{A}_S, \Omega, H\text{d}, CT) \). \( C \) runs \( \text{Authen}(\mathcal{P}, H\text{d}, \tilde{A}_S, \Omega) \). If \( \text{Authen}(\mathcal{P}, H\text{d}, \tilde{A}_S, \Omega) = 0 \), \( C \) quits; otherwise, \( C \) runs \( \text{Sanitizing}(\text{params}, \text{ssk}, CT) \rightarrow CT' \), and sends \( CT' \) to \( A \).

**Challenge.** \( A \) submits a message \( M^* \), an access structure \( \mathcal{M}^* \) associated with a set of attributes \( \Omega^* \) and a set of attributes \( \tilde{A}_{S^*} \) with the following limitations \( \tilde{A}_{S^*} \notin Q_K \). \( C \) runs \( \text{KeyGen}(\text{msk}, \text{params}, \tilde{A}_{S^*}) \rightarrow S\text{K}_{S^*} \),
\[ \text{Enc}(\text{params}, \text{spk}, M^*, \mathcal{M}, SK_S^*) \rightarrow (\tilde{A}_S^*, CT^*) \text{ where } (\tilde{A}_S^*, CT^*) \notin Q_E. \] \[ \mathcal{C} \text{ sends the challenged ciphertext } (\tilde{A}_S^*, CT^*) \text{ to } \mathcal{A}. \]

Phase 2. \( \mathcal{A} \) can adaptively make the following queries multiple times.

Key Generation Query. This query is the same as in Phase 1 with the limitation \( \tilde{A}_S \notin Q_K. \)

Encryption Query. This query is the same as in Phase 1 with \( (\tilde{A}_S^*, CT^*) \notin Q_E. \)

Sanitizer Query. This is the same as in Phase 1.

Output. \( \mathcal{A} \) outputs an authentication header \( Hd^* \). \( \mathcal{A} \) wins the game if \( \text{Authen}(P, Hd^*, \tilde{A}_S^*, \Omega^*) \rightarrow 1. \)

**Definition 3.** A fine-grained information flow control using attributes scheme is \( \epsilon(\ell) \) no write-down/receiver secure if for all probable polynomial-time (PPT) adversaries \( \mathcal{A} \) can win the above game with the advantage

\[ \text{Adv}_\mathcal{A} = \Pr \left[ \text{Authen}(P, Hd^*, \tilde{A}_S^*, \Omega^*) \rightarrow 1 \right] \leq \epsilon(\ell). \]

We say that a fine-grained information flow control using attributes scheme is no write-down/receiver weakly secure if an initialization phase \text{Initial} is added before the \text{Setup} phase, where \( \mathcal{A} \) must submit the attributes which he wants to query the \text{KeyGen}, \text{Enc} and \text{Sanitizing} oracles.

**2.3. Bilinear Map and Complexity assumptions**

Let \( G_1, G_2 \) and \( G_\tau \) be three cyclic groups with prime order \( p \) and \( g, h \) be generator of \( G_1 \) and \( G_2 \), respectively. A map \( e : G_1 \times G_2 \rightarrow G_\tau \) is a bilinear map if the following properties can be satisfied:

1. **Bilinearity.** For all \( g \in G_1, h \in G_2 \) and \( x, y \in \mathbb{Z}_p \), \( e(g^x, h^y) = e(g^y, h^x) = e(g, h)^{xy}; \)
2. **Non-degeneracy.** \( e(g, h) \neq 1_\tau \), where \( 1_\tau \) is the identity element of \( G_\tau; \)
3. **Efficiency.** For all \( g \in G_1 \) and \( h \in G_2 \), there is an efficient algorithm to compute \( e(g, h). \)

We say that a bilinear map is symmetric if \( G_1 = G_2 \). Let \( BG(1^\ell) \rightarrow (e, p, G, G_\tau) \) be a generator of bilinear group which takes as input a security parameter \( 1^\ell \) and outputs a bilinear group \( (e, p, G, G_\tau) \) with prime order \( p \) and \( e : G \times G \rightarrow G_\tau. \)
Definition 4. (Decisional Parallel Bilinear Diffie-Hellman Exponent (DPBDHE) Assumption [58]) Let $BG(1^\ell) \rightarrow (e, p, G, G_\tau)$, $g$ be a generator of $G$ and $a, s, x_1, x_2, \ldots, x_q \xleftarrow{R} \mathbb{Z}_p$. Suppose that

$$Y = (g, g^a, g^{a^2}, \ldots, g^{a^{q+2}}, \ldots, g^{a^{2q}},$$

$$\forall 1 \leq j \leq q, g^{as_x^j}, g^{as_y^j}, \ldots, g^{as_t^j}, \ldots, g^{as_k^j},$$

$$\forall 1 \leq k, j \leq q, k \neq j, g^{a_s x_k^j}, \ldots, g^{a_s x_k^j}).$$

We say that the DPBDHE assumption holds on the bilinear group $(e, p, G, G_\tau)$ if for all probable polynomial-time (PPT) adversary $A$ can distinguish $e(g, g)^{a^{q+1}s}$ from a random $T \in G_\tau$ with negligible advantage, namely

$$\text{Adv}_A = \left| \Pr[A(Y, e(g, g)^{a^{q+1}s}) = 1] - \Pr[A(Y, T) = 1] \right| \leq \epsilon(\ell).$$

Definition 5. ($q$-Strong Diffie-Hellman (SDH) Assumption [6]) Let $BG(1^\ell) \rightarrow (e, p, G, G_\tau)$, $g$ be a generator of $G$ and $x \xleftarrow{R} \mathbb{Z}_p$. Given $(p, g, g^x, g^{x^2}, \ldots, g^{x^q})$, we say that the $q$-SDH assumption holds on $G$ if for all probable polynomial-time (PPT) adversary $A$ can output $(c, g^{\frac{1}{x+c}})$ with a negligible advantage, namely

$$\text{Adv}_A = \Pr[A(p, g, g^x, g^{x^2}, \ldots, g^{x^q}) \rightarrow (x, g^{\frac{1}{x+c}})] \leq \epsilon(\ell),$$

where $c \in \mathbb{Z}_p - \{0\}.$

2.4. Waters Attribute-Based Encryption [58]

This ABE scheme was proposed by Waters [58] and works as follows.

- Setup. Let $U = \{att_1, att_2, \ldots, att_N\}$ be a universal set of attributes. The central authority $CA$ takes as input a security parameter $1^\ell$ and runs $BG(1^\ell) \rightarrow (p, g, G, G_\tau)$. Let $g, h_1, h_2, \ldots, h_N$ be generators of $G$. $CA$ selects $\alpha, a \xleftarrow{R} \mathbb{Z}_p$, and computes $h = e(g, g)^\alpha$ and $\tilde{h} = g^a$.

The master secret key is $\text{msk} = (\alpha, a)$ and the public parameters are $\text{params} = (p, e, g, G, G_\tau, g, h, \tilde{h}, h_1, h_2, \ldots, h_N)$.

- Encryption. To encrypt a message $M \in G_\tau$, the encryptor selects an access structure $(\mathcal{M}, \rho)$, where $\mathcal{M}$ is a $k \times n$ matrix and $\rho : M_i \rightarrow \text{attr}_{\rho(i)} \in U$ is a map which maps the $i$-th row $M_i$ of $\mathcal{M}$ to an attribute
att\_{\rho(i)} in U. The encryptor selects \( \vec{v} = (s, v_2, v_3, \cdots, v_n) \xleftarrow{\$} \mathbb{Z}_p^n \) and \((r_1, r_2, \cdots, r_k) \xleftarrow{\$} \mathbb{Z}_p^k\), and computes \( C = M \cdot h^s, \quad C' = g^s, \quad (C_1 = g^{a\lambda_i} h^{r_1}_{\rho(1)}, D_1 = g^{r_1}), \cdots, (C_k = g^{a\lambda_i} h^{r_k}_{\rho(k)}, D_k^{a\lambda_i}) \), where \( \lambda_i = \vec{v} \cdot M_i \) for \( i = 1, 2, \cdots, k \). The ciphertext is \( CT = (C, C', (C_1, D_1), \cdots, (C_k, D_k)) \).

- **KeyGen.** To generate a secret key for a user U with a set of attribute \( A_U \), CA selects \( t_u \xleftarrow{\$} \mathbb{Z}_p \), and computes \( K_U = g^{a^{t_u}}, \quad L_U = g^{t_u} \) and \((K_x = h_x^{t_u})_{\text{att}_{\rho(i)}} \in A_U \). The secret key is \( SK_U = (K_U, L_U, (K_x)_{\text{att}_{\rho(i)}}) \).

- **Decryption.** To decrypt a ciphertext \( CT \), a decryptor takes as input his secret key \( SK_D = (K_D, L_D, (K_x)_{\text{att}_{\rho(i)}}) \), and computes \( M = \prod_{i \in \{1, 2, \cdots, k\}} e(C_i, K_D)^{w_i} \), where \( I = \{1, 2, \cdots, k\} \) is defined as \( I = \{i : \rho(i) \in A_D\} \) and \( w_i \in \mathbb{Z}_p \) such that \( \sum_{i=1}^{k} w_i \lambda_i = s \).

**Theorem 1.** (Waters [58]) This ABE scheme is selectively secure if the DPBDHE assumption holds on \((e, p, G, G_T)\).

### 2.5. Boneh-Boyen Short Signature Scheme [6]

This short signature scheme was proposed by Boneh and Boyen [6], and works as follows.

- **KeyGen.** Let \( \mathcal{BG}(1^\ell) \rightarrow (e, p, G, G_T) \) and \( g \) be a generator of \( G \). The signer selects \( \alpha \xleftarrow{\$} \mathbb{Z}_p \) and computes \( Y = g^\alpha \). The secret key is \( sk = \alpha \) and the public parameters are \( params = (e, p, G, G_T, g, Y) \).

- **Signing.** To sign a message \( m \in \mathbb{Z}_p \), the signer computes \( \sigma = g^{\frac{1}{\alpha + m}} \). The signature of \( m \) is \((m, \sigma)\).

- **Verification.** To verify a signature \((m, \sigma)\), the verifier checks \( e(\sigma, Y g^m) = e(g, g) \). If the equation holds, \((m, \sigma)\) is a valid signature; otherwise, it is invalid.

**Theorem 2.** (Boneh and Boyen [6]) This signature scheme is secure against existential forgery under a weak chosen message attack if the q-SDH assumption holds on \((e, p, G, G_T)\).
3. Construction

In this section, a concrete construction of our fine-grained IFC using attributes scheme is described. We first introduce a high-level overview of our scheme, and then present a formal construction.

3.1. High-Level Overview

In our scheme, CA publishes an IFC policy \( P \) defined over a set of attributes. Each user’s (sender and receiver) identity is bound with a set of attributes. When joining the system, each \( U \) obtains a secret key for his/her attributes \( A_U \) from CA. \( SK_U \) can implement two functions: (1) it can be used to decrypt ciphertexts \( CT \) if \( A_U \models M \) where \( M \) is the access structure embed in \( CT \); (2) it can be used to generate authentication headers for any subset \( \tilde{A}_U \subseteq A_U \) with \( P(\tilde{A}_U, \Omega) = 1 \) where \( \Omega \) is the set of attributes associated with an access structure \( M \).

Specifically, the no read-up rule is implemented by encrypting a message \( M \) under an access structure \( M \) and the San’s public key so that a receiver \( R \) can decrypt a ciphertext \( CT' \) if \( A_R \models M \) and \( CT' \) is the sanitized ciphertext generated by San using his/her secret key and the ciphertext \( CT \). Furthermore, the no write-down rule is realized by using authentication headers \( Hd \) which are computed by senders \( S \) using their secret keys. Prior to sanitizing a ciphertext, San first determines whether \( S \) is authorized to communicate with \( R \) (\( A_R \models M \)) by checking \( P(\tilde{A}_S, \Omega) \models 1 \) and validating \( Hd \). If so, San sanitizes the ciphertext and publishes it so that all receivers \( R \) with \( A_R \models M \) can decrypt it; otherwise, San stops the communication request. Finally, \( R \) can decrypt \( CT' \) if \( A_R \models M \).

3.2. Our Construction

Our formal construction is presented in Fig. 2.

**Setup.** CA runs the Setup algorithm to generate the public parameters \( \text{parms} \) and a master secret key \( msk \); a secret-public key pair \( (\vartheta, \tilde{Y}) \) for San, publishes a universal of attributes \( U \) and an IFC policy \( P : U \times U \to \{0, 1\} \) specifying the relationships between the attributes holden by senders and receivers.

**KeyGen.** When a user \( U \) (receiver/sender) with identity \( ID_U \) and attributes \( A_U \) joining the system, CA generates a secret key \( SK_U \) for him/her, where \( SK_U \) is bound with \( ID_U \) and \( A_U \) by using the signature scheme [6]. Notably,
$SK_U$ can be used to decrypt ciphertexts (no read-up rule) and compute authentication headers (no write-down rule).

**Encryption.** This algorithm works as follows: (1) $S$ selects an access structure $(M, \rho)$ associated with a set of attributes $\Omega$, and encrypts $M$ under $(M, \rho)$ and San’s public key $\tilde{Y}$; (2) $S$ selects a subset of his/her attributes $\tilde{A}_S$ with $P(\tilde{A}_S, \Omega) = 1$ and computes an authentication header $Hd$ by using his/her secret key $SK_S$. Especially, to prevent senders from combing authentication headers, the ciphertext $CT$ is bound with $Hd$. Finally, $S$ sends $(\tilde{A}_S, Hd, CT)$ to $San$. As in the ABE scheme [5], we suppose that the access structure $(M, \rho, \Omega)$ is included $CT$.

**Sanitizer.** This algorithm consists of two sub-algorithms: Authentication and Sanitizing. In the Authentication phase, after receiving $CT$ and $Hd$, $San$ first checks whether $P(\tilde{A}_S, \Omega) = 1$ and $Hd$ is valid. If so, $San$ goes to the next step; otherwise, $San$ stops the communication request. In the Sanitizing phase, $San$ uses her secret key to sanitize the ciphertext $CT$ according to the access structure $(M, \rho)$ and obtains a sanitized ciphertext $CT'$. Finally, $San$ publishes $CT'$.

**Decryption.** After receiving $CT'$, a receiver $R$ can use his/her secret key $SK_R$ to decrypt $CT'$ and obtain the plaintext $M$ if $A_R \models M$.

### 3.3. Correctness Analysis

To verify the correctness of our scheme in Fig. 2, the following two cases are considered: *Case-I* and *Case-II*. *Case-I* is used to verify the correctness of the Authentication algorithm; while *Case-II* is used to verify the correctness of the Decryption algorithm.

**Case-I.** We have

\[
e(K, g) = e(K_S \tilde{h}^t, g) = e(g^\alpha g^\eta^\gamma h^{\frac{1}{t}+H(ID_S)}, g)
\]

\[
= e(g, g)^\alpha \cdot e(g^\eta, g^{\frac{1}{t}+H(ID_S)})
\]

\[
= h \cdot e(\tilde{h}, L_1) \cdot e(\eta^\frac{1}{z}, g^{\frac{1}{t}+H(ID_S)})
\]

\[
= h \cdot e(\tilde{h}, L_1) \cdot e(R_2, R_1),
\]

\[
e(R_2, g) = e(\eta^\frac{1}{z}, g) = e(\eta, g^\frac{1}{z}) = e(\eta, R_3),
\]

\[
e(R_4, g) = e((\tilde{h}_2 g^{H(ID_S)})^{\frac{1}{z}}, g) = e(\tilde{h}_2, g^\frac{1}{z}) \cdot e(g, g)^{H(ID_S)} = e(\tilde{h}_2, R_3) \cdot R_5,
\]
Setup. Let $\mathcal{U} = \{attr_1, attr_2, \cdots, attr_N\}$ be a universal set of attributes. CA publishes an IFC policy $\mathbb{P}: \mathcal{U} \times \mathcal{U} \rightarrow \{0,1\}$ which specifies the relationship between senders’ attributes and receivers’ attributes, for example hierarchy, inclusion, etc. CA runs $\mathcal{BG}(1^k) \rightarrow (p, e, \mathbb{G}, \mathbb{G}_t)$. Let $g$ and $\eta$ be generators of $\mathbb{G}$ and $H: \{0,1\}^* \rightarrow \mathbb{Z}_p$ be a cryptographic hash function. CA selects $\alpha, \gamma, a, \vartheta \leftarrow \mathbb{R} \mathbb{Z}_p$ and $h_1, h_2, \cdots, h_N \leftarrow \mathbb{R} \mathbb{G}$, and computes $h = e(g, g)^\alpha$, $\tilde{h}_1 = g^a$, $\tilde{h}_2 = g^\gamma$ and $\tilde{Y} = g^\vartheta$. The master secret key is $\text{msk} = (\alpha, a, \gamma)$, the secret-public key pair of $\text{San}$ is $(\vartheta, \tilde{Y})$, and the public parameters are $\text{params} = (e, p, \mathbb{G}, \mathbb{G}_t, g, \eta, h, \tilde{h}_1, h_2, \cdots, h_N, \tilde{Y})$. CA sends $(\vartheta, \tilde{Y})$ to the sanitizer $\text{San}$.

KeyGen. To generate a secret key for a user $U$ (receiver/sender) with an identity $ID_U$ and a set of attributes $A_U$, CA first selects $t_u \leftarrow \mathbb{R} \mathbb{Z}_p$, and computes

$$K_u = g^\alpha g^{at_u \eta^{-1 + \frac{1}{2n}H(ID_U)}}, \; L_u = g^{t_u}, \; R_u = g^{-\frac{1}{2n}H(ID_U)}, \; \{K_u^x = h_x^{t_u}\}_{attr \in A_U}.$$  

The secret key for $U$ is $SK_U = (K_u, L_u, R_u, (K_v^x)_{attr \in A_U})$.

Encryption. Suppose that a sender $S$ holds a set of attributes $A_S$ and obtains a secret key $SK_S = (K_S, L_S, R_S, (K_v^x)_{attr \in A_S})$. Let $\mathcal{M}$ be an $k \times n$ access matrix, $\mathcal{M}_i$ be the vector corresponding to the $i$-th row of $\mathcal{M}$ and $\rho$ be the function which maps rows of $\mathcal{M}$ to attributes in $\mathcal{U}$. Let $\Omega = \{\rho(1), \rho(2) \cdots, \rho(k)\}$ be the set of attributes corresponding to the matrix $\mathcal{M}$.

To encrypt a message $M \in \mathbb{G}_t$ under $\mathcal{M}$ and $\tilde{Y}$, $S$ selects a random vector $\vec{v} = (s, y_2, \cdots, y_n) \leftarrow \mathbb{Z}_p^n$, $r_1, r_2, \cdots, r_k \leftarrow \mathbb{R} \mathbb{Z}_p$, and computes

$$C = M \cdot h^s \cdot e(g, \tilde{Y})^{-s}, \; C' = g^s, \; C'' = \eta^s \tilde{Y}^s,$$

$$(C_1 = g^{a_1} h_{\rho(1)}^{-r_1} \tilde{Y}^{-r_1}, D_1 = g^{r_1}), \; \cdots, \; (C_k = g^{a_k_1} h_{\rho(k)}^{-r_k} \tilde{Y}^{-r_k}, D_k = g^{r_k})$$

where $\lambda_i = \vec{v} \cdot \mathcal{M}_i, i = 1, 2, \cdots, \ell$. The ciphertext is $CT = (C, C', C'', (C_1, D_1), \cdots, (C_k, D_k), (\mathcal{M}, \rho, \Omega))$.

To convince $\text{San}$ that he/she is allowed to communicate with receivers $A_R \models \mathcal{M}$, $S$ selects a subset of his/her attributes $\mathcal{A}_S \subseteq A_S$ with $\mathbb{P}(\mathcal{A}_S, \Omega) = 1$. Then, $S$ selects $t, w, z \leftarrow \mathbb{R} \mathbb{Z}_p$, and computes
\[ K = K_S h^L_{i_1} = g^a g^{a(t_s+t)} \eta^{\frac{1}{R_3}h^{(ID_S)}} \], L_1 = L_S g^t = g^{t_s+t}, L_2 = L_1 h^{(CT)} \eta^{w},
L_3 = g^w, R_1 = R_5^\eta = g^{\frac{1}{R_3}h^{(ID_S)}}, R_2 = \eta^{\frac{1}{R_3}}, R_3 = g^{\frac{1}{R_3}}, R_4 = (h_2 g^{h^{(ID_S)}})^\frac{1}{R_3},
R_5 = e(g, g)^{h^{(ID_S)}} \] and \[ R_6 = \prod_{att \in \tilde{A}_S} K_x h^t_x = \prod_{att \in \tilde{A}_S} h^{t_x+t}. \]

The authentication header is \( Hd = (K, L_1, L_2, L_3, R_1, R_2, R_3, R_4, R_5, R_6, \tilde{A}_S) \). \( S \) sends \( (\tilde{A}_S, Hd, CT) \) to \( San \).

**Sanitizer.** After receiving \( (\tilde{A}_S, Hd, CT) \) from \( S \), \( San \) works as follows:

1. **Authentication.** To verify whether \( S \) is authorized to send a message to receivers with \( A_R \models \mathcal{M}, San \) checks:

\[ \mathbb{P}(\tilde{A}_S, \Omega) = 1, e(K, g)^\frac{1}{R_2} = h \cdot e(L_1, h) \cdot e(R_1, R_2), e(R_2, g)^\frac{1}{R_2} = e(\eta, R_3),
\]
\[ e(R_4, g)^\frac{1}{R_2} = e(h_2, R_3) \cdot R_3, e(R_1, R_4)^\frac{1}{R_2} = e(g, g),
\]
\[ e(L_2, g)^\frac{1}{R_2} = e(L_1, g)^{h^{(CT)}} \cdot e(L_3, \eta) \] and \( e(R_6, g)^\frac{1}{R_2} = e(\prod_{att \in \tilde{A}_S} h_x, L_1). \)

If above equations hold, \( San \) goes to the next step; otherwise, \( San \) stops the communication.

2. **Sanitizing.** \( San \) selects \( \bar{r} \leftarrow \mathbb{Z}_p \) and computes \( \tilde{C} = C \cdot e(g, C'^{\eta})^\bar{r} = M \cdot h^{s+\bar{r}}, \tilde{C}_1 = C' g^\bar{r} = g_{s+\bar{r}}, \tilde{C}_2 = C'^{\eta} \cdot h^{\bar{r}}, \tilde{C}_3 = \tilde{h} \cdot g^\bar{r}, \tilde{C}_4 = C_1 D_1^\eta \cdot h_{\rho(1)}^{-\bar{r}}, \tilde{D}_1 = D_1 \cdot g = g_{(r_1+\bar{r})}, \ldots, \tilde{C}_k = C_k D_k^\eta \cdot h_{\rho(k)}^{-\bar{r}}, \tilde{D}_k = D_k \cdot g = g_{(r_k+\bar{r})}. \)

\( San \) publishes \( CT = (\tilde{C}, \tilde{C}_1, \tilde{C}_2, \tilde{C}_3, (\tilde{C}_1, \tilde{D}_1), \ldots, (\tilde{C}_k, \tilde{D}_k), (\mathcal{M}, \rho, \Omega)). \)

**Decryption.** Suppose that a receiver \( R \) holds a secret key \( SK_R = (K_R, L_R, R_R, (K_R^{\eta_{att}} \text{ att} \in A_R)) \) and \( A_R \models \mathcal{M} \). Given a sanitized ciphertext \( CT = (\tilde{C}, \tilde{C}_1, \tilde{C}_2, \tilde{C}_3, (\tilde{C}_1, \tilde{D}_1), \ldots, (\tilde{C}_k, \tilde{D}_k), (\mathcal{M}, \rho, \Omega)) \). \( R \) computes \( D = \frac{e(\mathcal{C}_1, K_R)}{\prod_{att \in \Omega}(e(\mathcal{C}_1, L_R) \cdot e(D_1, K_R^{\eta_{att}})) \cdot e(L_R, \tilde{C}_3) \cdot e(R_R, \tilde{C}_2)} = h^{s+\bar{r}} \) and \( M = \frac{\mathcal{C}_1}{D}. \)

Figure 2: Fine-grained Information Flow Control Using Attributes Scheme
\[ e(R_1, R_2) = e(g^{\frac{a}{\lambda' H(D_S^0)}}, g^{\frac{\lambda H(D_S^0)}}) = e(g, g), \]
\[ e(L_2, g) = e(L_1^{H(CT)}, g) \cdot e(\eta^w, g) = e(L_1, g)^{H(CT)} \cdot e(\eta, L_3) \]

and

\[ e(R_6, g) = e(\prod_{\text{att}_x \in \tilde{A}_S} h_x^{t_x + \tilde{t}}, g) = e(\prod_{\text{att}_x \in \tilde{A}_S} h_x, g^{t_x + \tilde{t}}) = e(\prod_{\text{att}_x \in \tilde{A}_S} h_x, L_1). \]

**Case-II.** We have

\[ e(\tilde{C}_1, K_R) = e(g^{s+\tilde{f}}, g^\alpha g^{\text{att}_e} \eta^{\frac{1}{\lambda' H(VR)}}) = e(g, g)^{\alpha(s+\tilde{f})} \cdot e(g, g)^{\text{att}_e(s+\tilde{f})} \cdot e(\eta, \eta)^{\frac{1}{\lambda' H(VR)}} = h^{s+\tilde{f}} \cdot e(g, g)^{\text{att}_e(s+\tilde{f})} \cdot e(\eta, \eta)^{\frac{1}{\lambda' H(VR)}}, \]

\[ e(R_R, \tilde{C}_2) = e(g^{\frac{1}{\lambda' H(VR)}}, \eta^{s+\tilde{f}}) = e(g, \eta)^{\frac{s+\tilde{f}}{\lambda' H(VR)}}, \]

\[ \prod_{\text{att}_t \in \Omega} \left( e(\tilde{C}_t, L) e(\tilde{D}_t, K_R^{\rho(1)}) \right)^{\omega_t} = \prod_{\text{att}_t \in \Omega} e(g^\alpha h_{r_i + \tilde{f}}, g^{\omega_i} \cdot e(g, g^{-(r_i + \tilde{f})}, h_{r_i}^{\omega_i}) = \prod_{\text{att}_t \in \Omega} e(g, g)^{\text{att}_e \lambda_i \omega_i} = e(g, g)^{\lambda_i \omega_i}, \]

\[ e(L_R, \tilde{C}_3) = e(g^{t_r}, g^{\tilde{f}}) = e(g, g)^{\text{att}_e \tilde{f}} \]

and

\[ D = \frac{e(\tilde{C}_1, K_R)}{\prod_{\text{att}_t \in \Omega} (e(\tilde{C}_t, L) \cdot e(\tilde{D}_t, K_R^{\rho(1)}))^\omega_t \cdot e(L_R, \tilde{C}_3) \cdot e(R_R, \tilde{C}_2)} = \frac{h^{s+\tilde{f}} \cdot e(g, g)^{\text{att}_e(s+\tilde{f})} \cdot e(\eta, \eta)^{\frac{1}{\lambda' H(VR)}} = h^{s+\tilde{f}} \cdot e(g, g)^{\text{att}_e(s+\tilde{f})} \cdot e(g, \eta)^{\frac{1}{\lambda' H(VR)}}}{e(g, g)^{\text{att}_e(s+\tilde{f})} \cdot e(\eta, g)^{\frac{1}{\lambda' H(VR)}}} = h^{s+\tilde{f}}. \]
Hence,
\[
\frac{\tilde{C}}{\tilde{D}} = \frac{M \cdot h^{s+\varphi}}{h^{s+\varphi}} = M.
\]

Remark. Platform-as-a-Service (PaaS) is a cloud computing model where users’ applications are isolated to guarantee data confidentiality and integrity. IFC has been introduced into PaaS to manage, specify, and control requirements for isolation and data sharing, and enforce them [3]. Data sharing can be implemented by encrypting data according to application policy. Meanwhile, strict isolation should be enforced when flows are not permitted [46]. Our scheme proposed in Fig. 2 supports both the no read-up rule and the no write-down rule, hence it can be potentially used to enforce data sharing and isolation in PaaS where the PaaS router [27] can work as the sanitizer.

4. Performance

In this section, we evaluate our scheme in Fig. 2 by using the code from the pairing-base cryptography (PBC) library [39]. When choosing an elliptic curve, two important factors should be considered: the group size \( l \) and the embedding degree \( d \). To achieve the same security as the 1024-bit RSA, we require that \( l \times d \geq 1024 \). Our scheme is evaluated on the Type A curve \( y^2 = x^3 + x \), where \( p \) is 160 bits, \( l = 512 \) and \( \mathbb{G}_1 = \mathbb{G}_2 \). The size of one element in \( \mathbb{G} \) and \( \mathbb{G}_\tau \) are 512 bits and 1024 bits, respectively. SHA - 1 is selected as a hash function.

4.1. Benchmark Time

The running time consumed by different operations on the bilinear group \((e, p, \mathbb{G}, \mathbb{G}_\tau)\) is obtained on a DELL E630 laptop with Intel(R) CoreTM 2 Duo CPU (T8100@2.10 GHz) and 2 GB RAM by running Ubuntu 9.10. The running time is computed by calculating the average of running each operation 10 times with random inputs using the test code from the PBC library [39]. Table 3 summarizes the running times of different operations on the bilinear group from Type A curve. By \( P, PP, E, PE, E_\tau \) and \( PE_\tau \), we denote the time of executing one pairing operation, executing one pairing operation with preprocessing, executing one exponent operation on \( \mathbb{G} \), executing one exponent operation on \( \mathbb{G}_\tau \) with preprocessing, executing one exponent operation on \( \mathbb{G}_\tau \), executing one exponent operation on \( \mathbb{G}_\tau \) with preprocessing, respectively.
Table 3: Benchmark Time of Different Operations on Type A Curve and SHA-1 (ms)

<table>
<thead>
<tr>
<th>Curve</th>
<th>$P$</th>
<th>$PP$</th>
<th>$E$</th>
<th>$PE$</th>
<th>$E_\tau$</th>
<th>$PE_\tau$</th>
<th>SHA-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type A</td>
<td>1.289</td>
<td>0.585</td>
<td>2.026</td>
<td>0.273</td>
<td>0.195</td>
<td>0.038</td>
<td>0.004</td>
</tr>
</tbody>
</table>

4.2. Performance Evaluation

The computation costs and communication costs of different algorithms in our scheme are presented in Fig. 3 and Fig. 4, respectively.

![Figure 3: The Computation Cost of Our Scheme](image1)

![Figure 4: The Communication Cost of Our Scheme](image2)

In summary, the Encryption algorithm is the most time consuming algorithm and has highest communication cost. The Decryption algorithm and the Sanitizing algorithm are the second and third most time consuming algorithms. Comparatively, the Authentication algorithm, the KeyGen algorithm and the Setup algorithm are less time consuming algorithms.

4.3. Efficiency Analysis

In our scheme, it takes about 0.8570ms to setup the system and 17.541 ms to determine whether a sender can communicate with a receiver, hence the computation cost of both the setup algorithm and the authentication algorithm are constant, instead of linear with the number of required attributes or
receivers in the system. Furthermore, the computation costs of the KeyGen algorithm, Encryption algorithm, Sanitizing algorithm and Decryption algorithm are linear with the number of required attributes and independent of the number of receivers/senders in the system. Meanwhile, the communication costs of Setup algorithm, KeyGen algorithm, Encryption algorithm and Sanitizing algorithm are linear with the number of required attributes and independent of the number of receivers/senders in the systems.

Nevertheless, the ciphertext size in our scheme is 704 bytes longer than that in [58] since an authentication header $H_d$ is required. The size of $H_d$ is constant, but the computation cost of $H_d$ is linear with the number of required attributes. The computation costs of the Decryption algorithm in our scheme is similar as that in [58] since only one additional pairing is required.

5. Security Analysis

In this section, the security of our scheme in Fig. 2 is formally proven. Specifically, the no read-up rule/sender security is reduced to the DPBDHE assumption, while the no write-down rule/receiver security is reduced to the $q$-SDH assumption.

5.1. No Read-Up Rule/Sender’s Security

**Theorem 3.** Our scheme is $\epsilon'(\ell)$ no read-up/sender selectively secure if the $\epsilon(\ell)$-DPBDHE assumption hold on the bilinear group $(e, p, \mathbb{G}, \mathbb{G}_\tau)$, where $\epsilon(\ell) = \frac{\epsilon'(\ell)}{2}$.

**Proof.** Suppose that there exists an adversary $A$ who can break the no read-up rule/sender’s security of our scheme, we can construct an algorithm $B$ that can use $A$ to break the DPBDHE assumption as follows.

Let

$$Y = (e, p, \mathbb{G}, \mathbb{G}_\tau, g, g^a, g^{a^2}, \ldots, g^{a^q}, g^{a^{q+2}}, \ldots, g^{a^{q^2}},$$

$$\forall 1 \leq j \leq q: g^{x_j}, g^{a x_j}, \ldots, g^{a^{q-2} x_j}, g^{a^{q+2} x_j}, \ldots, g^{a^{q^2-2} x_j},$$

$$\forall 1 \leq k, j \leq q, k \neq j: g^{a^k x_j}, \ldots, g^{a^q x_j} x_j).$$

The challenger $C$ flips an unbiased coin with $\{0, 1\}$, and obtains a bit $b \in \{0, 1\}$. If $b = 0$, it sends $(Y, T = e(g, g)^{a^{q+1} s})$ to $B$; otherwise, it sends $(Y, T = R)$ to $B$ where $R \leftarrow \mathbb{G}_\tau$. $B$ will output his guess $b'$ on $b$. 

23
Initial. $A$ submits a challenged access structure $(\mathcal{M}^*, \rho^*)$, where $\mathcal{M}^*$ is a $k^* \times n^*$ matrix and $\rho^*(i)$ maps the rows of $\mathcal{M}^*$ to attributes $\Omega^* = \{\rho^*(1), \rho^*(2), \cdots, \rho^*(k^*)\}$.

Setup. Let $X$ be the set of indices $i$, such that $\rho^*(i) = att_x$. $B$ selects $\alpha', \gamma, \nu \xleftarrow{R} \mathbb{Z}_p$, and sets $\alpha = \alpha' + a^{q+1}$. $B$ computes $h = e(g, g)^\alpha = e(g, g)^{a^{q+1}} \cdot e(g, g)^{\alpha'}, \hat{h}_1 = g^\alpha, \hat{h}_2 = g^\gamma$ and $\eta = g^\nu$. Furthermore, $B$ selects $t_x \xleftarrow{R} \mathbb{Z}_p$ and computes $h_x = g^{t_x} \prod_{i \in X} g_{\nu_i}^{a_{M^*_{i1}}} \cdot g_{\nu_i}^{a_{M^*_{i2}}} \cdots g_{\nu_i}^{a_{M^*_{in^*}}}$. Hence, $h_x = g^{t_x}$ if $X = \Phi$. Finally, $B$ selects $\beta, \vartheta \xleftarrow{R} \mathbb{Z}_p$ and computes $\hat{h}_x - \beta$ and $\tilde{Y} = g^\vartheta$. $B$ sends $\text{params} = (e, p, g, \mathbb{G}, \mathbb{G}_T, g, h, \eta, \hat{h}_1, \hat{h}_2, h_1, \cdots, h_N, \tilde{Y})$ to $A$.

Phase 1. $A$ can make the following queries adaptively.

Key Generation Query. $A$ submits an identity $ID_U$ with a set of attributes $A_U \neq \mathcal{M}^*$. $B$ selects $r_u \xleftarrow{R} \mathbb{Z}_p$ and a vector $\vec{w} = (w_1, w_2, \cdots, w_{n^*})$ where $w_1 = -1$ and $\vec{w} \cdot M^*_i = 0$. Let $t = r_u + w_1 a + w_2 a^{q-1} + w_{n^*} a^{q-n^*+1}$, $L_U = g^t = g^r \prod_{i=1}^{n^*} (g^{a^{q-j+1}})^{w_i}$, $K_U = g^{t'} \prod_{j=2}^{n^*} (g^{a^{q-j+2}})^{w_i}$, $\eta^{-1} \prod_{i \in X} (\eta^{v_i} g_{\nu_i}^{a_{M^*_{i1}}} \cdots g_{\nu_i}^{a_{M^*_{in^*}}})$ and $R_U = g^{t''} \prod_{i \in X} (\nu_i^{v_i} g_{\nu_i}^{a_{M^*_{i1}}} \cdots g_{\nu_i}^{a_{M^*_{in^*}}})$. $B$ computes:

1. For $\text{att}_i \in A_U$ and $\rho(i) \neq att_x$, $K_i = L^{t_x}$;
2. For $\text{att}_i \in A_U$ and $\rho(i) = att_x$,

$$K_i = L^{t_x} \prod_{i \in X} \prod_{j=1}^{n^*} \left( g^{a_{M^*_{ij}}^{(s_j)}} \right)^{w_i} \prod_{d=1}^{n^*} \left( g^{a_{M^*_{ij}}^{(s_j+d)}} \right)^{w_k}.$$ $B$ sends $(K_U, L_U, R_U, (K_i)_{i \in A_U})$ to $A$.

Encryption Query. $A$ submits a message $M$, an identity $ID_S$ with a set of attributes $\tilde{A}_S$ and an access structure $(\mathcal{M}, \rho)$ where $\mathcal{M}$ is an $k \times n$ matrix and $\Omega = \{\rho(1), \rho(2), \cdots, \rho(k)\}$ is a set of attributes associated to the rows of $\mathcal{M}$. $B$ first computes a secret key $(K_{S_1}, L_{S_1}, R_{S_1}, (K_{i})_{i \in S_1})$ as above. Then, $B$ selects a random vector $\vec{v} = (s', y_2, \cdots, y_n) \xleftarrow{R} \mathbb{Z}_p$ and $r_1, r_2, \cdots, r_l \xleftarrow{R} \mathbb{Z}_p$, and computes $s' = M \cdot e(g, g)^{s' \alpha} \cdot e(g, \tilde{Y})^{s'}, C' = g^{s'}, C'' = \eta^{s'} \tilde{Y}^{s'}, C_1 = g^{a_{M_1}^{(s_1)}} \tilde{Y}^{-r_1}, D_1 = g^{r_1}, \cdots, C_k = g^{a_{M_k}^{(s_k)}} \tilde{Y}^{-r_k}, D_k = g^{r_k}$, where $\lambda_i = \vec{v} \cdot \mathcal{M}_i, i = 1, 2, \cdots, k$. The ciphertext is $CT = (C, C', C'', (C_1, D_1), \cdots, (C_k, D_k), (\mathcal{M}, \rho, \Omega))$. 


\[ B \text{ selects } t', w', z', \xi \sim \mathbb{Z}_p \text{ and computes, } K = K_S \cdot \tilde{h}^{t'} = g^a g^{(t_s + t')} \eta^{-1} h^{\tilde{h}_{1,S}} \]

\[ L_1 = L_S \cdot g^{w'} = g^{w'} \cdot L_2 = L_1 H(CT) \eta^{w'}, \quad L_3 = g^{w'}, \quad R_1 = R_s' = \eta^{-1} h^{\tilde{h}_{1,S}}, \quad R_2 = \eta^{-1} h^{\tilde{h}_{1,S}}, \quad R_3 = g^{\tilde{h}_{1,S}}, \quad R_4 = (h_2 g^{1D_S})^{\frac{1}{2}}, \quad R_5 = e(g, g) \frac{H(D_S)}{\sigma} \text{ and } R_6 = \prod_{a_t(i) \in A_s} K_1. \]

\[ h_i^{t'} = \prod_{a_t(i) \in A_s} h_i^{t + t'} \]. The authentication header is \( H_d = (K, L_1, L_2, L_3, R_1, R_2, R_3, R_4, R_5, R_6, \tilde{A}_s) \). Finally, \( B \) sends \((\tilde{A}_s, H_d, CT)\) to \( A \).

**Sanitizer Query.** \( A \) submits a ciphertext \( CT = (C, C', C'', (C_1, D_1), \ldots, (C_k, D_k), (M, \rho, \Omega)) \). \( B \) selects \( \tilde{r} \overset{R}{\leftarrow} \mathbb{Z}_p \) and computes \( C = C \cdot e(g, C') \cdot h^{\tilde{r}}, \quad \tilde{C}_1 = C' g^{\tilde{r}}, \quad \tilde{C}_2 = C'' g^{\tilde{r}} \eta^{\tilde{r}}, \quad \tilde{C}_3 = \tilde{h}^{\tilde{r}}, \quad \tilde{C}_1 = C_1 \cdot D_1^{\tilde{r}} \cdot h^{\tilde{r}} \rho(1) = g^{\tilde{r}} h^{\rho(1)}, \quad \tilde{D}_1 = D_1 \cdot g^{(r_1 + \tilde{r})}, \quad \tilde{C}_k = C_k \cdot D_k^{\tilde{r}} \cdot h^{\rho(k)} = g^{\tilde{r}} \rho(k) \cdot g^{-r_k}, \quad \tilde{D}_k = D_k \cdot g^{(r_1 + \tilde{r})} \rho(k) \cdot g^{-r_k} \). The sanitized ciphertext is \( \tilde{C}T = (\tilde{C}, \tilde{C}_1, \tilde{C}_2, \tilde{C}_3, (\tilde{C}_1, \tilde{D}_1), \ldots, (\tilde{C}_k, \tilde{D}_k, (M, \rho)) \).

**Challenge.** \( A \) submits two messages \( M_1 \) and \( M_2 \). \( B \) flips an unbiased coin with \( \{0, 1\} \) and obtains a bit \( \tilde{b} \). \( B \) selects \((\tilde{y}_2, \tilde{y}_3, \ldots, \tilde{y}_n, \tilde{y}) \overset{R}{\leftarrow} \mathbb{Z}_p^{* - 1} \) and sets \( \nu^* = (s, sa + y_2, sa^2 + y_3, \ldots, sa^{n-1} + y_n) \). \( B \) selects \((\tilde{r}_1, \tilde{r}_2, \ldots, \tilde{r}_n, \tilde{r}) \overset{R}{\leftarrow} \mathbb{Z}_p^* \) and sets \( \tilde{R}_i = (j | \rho(i) = \rho(j) \land i \neq j) \) for \( i = 1, 2, \ldots, n^* \). \( B \) computes

\[ C^* = M_{\tilde{b}} \cdot T \cdot e(g^s, g^{\tilde{r}}), \quad E^* = g^s, \quad F^* = (g^s)^v, \quad D_i^* = g^{-\tilde{r}_i}, g^{\tilde{r}_i}, \]

and

\[ C_i^* = h^{\tilde{r}_i} \cdot \left( g^{\tilde{y}^s(i)} \right)^{-2 \nu^*(i)} \cdot \left( \prod_{j=2}^{n^*} (g^a)^{M_j \cdot \tilde{y}_i} \right) \cdot \left( \prod_{j \in \tilde{R}_i} \prod_{k \leftarrow 1} g^{a \cdot \tilde{y}_i} \cdot g^{b \cdot \tilde{y}_i} \cdot g^{c \cdot \tilde{y}_i} \right) \]

\( B \) sends \( \tilde{C}T^* = (C^*, E^*, F^*, (C_i^*, D_i^*)_{\nu^*(i) \in \{1, 2, \ldots, n^*\}}, (M^*, \rho^*)) \) to \( A \).

**Phase 2.** This is the same as in Phase 1.

**Guess.** \( A \) outputs his guess \( \tilde{b}' \) on \( b \). If \( \tilde{b}' = b \), \( B \) outputs \( b' = 0 \); if \( \tilde{b}' \neq b \), \( B \) outputs \( b' = 1 \).

If \( b = 0 \), \( CT^* \) is a correct ciphertext, so \( A \) can output \( \tilde{b}' = \tilde{b} \) with \( \Pr[\tilde{b}' = b | b = 0] \geq \frac{1}{2} + \epsilon(\ell) \). When \( A \) outputs \( \tilde{b}' = b \), \( B \) outputs \( b' = 0 \). Hence, \( \Pr[\tilde{b}' = b | b = 0] \geq \frac{1}{2} + \epsilon(\ell) \). If \( b = 1 \), \( CT^* \) are random numbers, so \( A \) can output \( \tilde{b}' \neq b \) with \( \Pr[\tilde{b}' \neq b | b = 1] = \frac{1}{2} \). When \( A \) outputs \( \tilde{b}' \neq b \), \( B \) outputs \( b' = 1 \). Hence, \( \Pr[\tilde{b}' = b | b = 1] = \frac{1}{2} \). Therefore, the advantage with which \( B \)
can break the PBDHE assumption is
\[
Adv_B^{DPBDHE} = \left| \frac{1}{2} \Pr[b' = b|b = 0] - \frac{1}{2} \Pr[b' = b|b = 1] \right|
\]
\[
\geq \frac{1}{2} \left( \frac{1}{2} + \epsilon(\ell) \right) - \frac{1}{2} \times \frac{1}{2} = \frac{\epsilon(\ell)}{2}.
\]

5.2. No Write-Done Rule/Receiver's Security

**Theorem 4.** Our scheme is \((q_1, q_2, \epsilon(\ell))\) no write-done/receiver weakly secure if the \((q, \epsilon(\ell))\)-SDH assumption holds on the group \(G\), where \(\epsilon(\ell) = \epsilon(\ell)\), \(q \geq q_1 + q_2 + 1\), \(q_1\) and \(q_2\) are the numbers of key generation queries and encryption queries made by the adversary \(A\), respectively.

**Proof.** Suppose that there exists an adversary \(A\) who can break the no write-done/receiver’s security of our scheme, we can construct an algorithm \(B\) that can use \(A\) to break the \((q, \epsilon(\ell))\)-SDH assumption as follows.

Let \(Y = (e, p, G, G_T, \tilde{g}, \tilde{g}^x, \tilde{g}^{x^2}, \ldots, \tilde{g}^{x^q})\). The goal of \(C\) is to output \((c, \tilde{g}^{\frac{1}{x^q}})\) where \(c \in \mathbb{Z}_p - \{0\}\).

**Initial.** \(A\) submits \(q'\) identities \(\Gamma = (ID_{U_1}, ID_{U_2}, \ldots, ID_{U_{\ell}})\).

**Setup.** Let \(q \geq q' + 1\), \(f(x) = \prod_{i=1}^{q'} (x + H(ID_{U_i})) = \sum_{i=0}^{q'} a_ix^i\), \(f'(x) = f(x)^2 = \sum_{i=0}^{2q'} c_i x^i\) and \(f_s(x) = \frac{f(x)}{H(ID_{U_{\ell}})}\). \(B\) selects \(a, \gamma, v, x_1, x_2, \ldots, x_N, y_1, y_2, \ldots, y_{q'} \overset{\$}{\leftarrow} \mathbb{Z}_p\), and computes \(g = \tilde{g}^{f(x)} = \prod_{i=0}^{q'} (\tilde{g}^{x^i})^{a_i}, \eta = g^v = \tilde{g}^{f(x)}, h = e(g, g)^{\gamma} = e(\tilde{g}^{f(x)}, \tilde{g}^{f(x)})^{\gamma} = e(\prod_{i=0}^{q'} (g^{x^i})^{a_i}, \prod_{i=0}^{q'} (g^{x^i})^{a_i})^{\gamma}, \tilde{h}_1 = g^{a}, \tilde{h}_2 = g^v = \tilde{g}^{f(x)} = \prod_{i=1}^{q+1} (g^{x^i})^{a_i-1}, \eta = g^v, h_1 = \tilde{g}^{f(x)}, \ldots, h_N = g^{x^N}\). \(B\) selects \(\tilde{v} \overset{\$}{\leftarrow} \mathbb{Z}_p\) and computes \(\tilde{Y} = g^\tilde{v}\). \(B\) sends the public parameters \(params = (e, p, G, G_T, g, h, \eta, h_1, h_2, h_1, \ldots, h_N, \tilde{Y})\) to \(A\).

**Key Generation Query.** \(A\) submits an identity \(ID_U \in \Gamma\). \(B\) selects \(t \overset{\$}{\leftarrow} \mathbb{Z}_p\) and computes \(K_U = g^t g^{at^{f(x)}} = g^t g^{at^{f(x)}e^{-H(ID_{U_i})}} = g^t g^{at^{e^{-H(ID_{U_i})}}} = g^t g^{\eta^{e^{-1}(ID_{U_i})}}, L_U = g', R_U = \tilde{g}^{f_s(x)} = \tilde{g}^{e^{-H(ID_{U_{\ell}})}}, K_i = h_i^{t} = h_i^{e^{-1}(ID_{U_i})}\), \(\{K_i = h_i^{t}\}_{att_i \in AU}\). \(B\) sends \((K_U, L_U, R_U, (K_i)_{att_i \in AU})\) to \(A\). Let \(Q_K\) be a set consisting of the identities selected by \(A\) to query secret keys.

**Encryption Query.** \(A\) submits a message \(M\), an identity \(ID_S \in \Gamma\) with a set of attributes \(A_S\) and an access structure \((M, \rho)\), where \(M\) is an \(k \times n\) matrix, \(\rho : M_i \rightarrow \rho(i)\) maps the \(i\)-th row of \(M\) to an attribute.
\(\rho(i) \in U\), and \(\Omega = \{\rho(1), \rho(2), \cdots , \rho(k)\}\). \(\mathcal{B}\) first computes a secret key \((K_S, L_S, R_S, (K_i)_{att_i \in A_S})\) as above. Then, \(\mathcal{B}\) selects a random vector \(\vec{v} = (s, y_2, \cdots , y_n) \xleftarrow{\$} \mathbb{Z}_p^n\) and \(r_1, r_2, \cdots , r_k \xleftarrow{\$} \mathbb{Z}_p\), and computes \(C = M \cdot (g, \bar{Y})^s \cdot e(g, g)^{s \sigma}, C' = g^s, C'' = \eta^s \bar{Y}^s, (C_1 = g^{a\lambda_1} h_{\rho(1)} \bar{Y}^{-r_1}, D_1 = g^r), \cdots , (C_k = g^{a\lambda_k} h_{\rho(k)} \bar{Y}^{-r_k}, D_k = g^r)\) where \(\lambda_i = \vec{v} \cdot \mathcal{M}_i, i = 1, 2, \cdots , k\). The ciphertext is \(CT = (C, C', C'', (C_1, D_1), \cdots , (C_k, D_k), (\mathcal{M}, \rho, \Omega))\).

\(\mathcal{B}\) selects \(t', w', z' \xleftarrow{\$} \mathbb{Z}_p\) and computes \(K = K_S \cdot \tilde{h}^{t'} = g^a g^{a(t'+t')} \eta^{\frac{t'}{\mathbb{M}(\mathbb{M}_S)}}, L_1 = L_S \cdot g^{t'} = g^{s+t'}, L_2 = L_1^{H(CT)} \eta^{w'}, L_3 = g^{w'}, R_1 = R_s^{t'}, R_2 = \eta^\frac{t'}{\mathbb{M}(\mathbb{M}_S)^2}, R_3 = g^\frac{t'}{2}, R_4 = (\tilde{h}_2 g^{ID_S})^{\frac{t'}{2}}, R_5 = e(g, g)^{\frac{t'}{ID_S}} \) and \(R_6 = \prod_{att_i \in A_S} K_i \cdot h^{t'} = \prod_{att_i \in A_S} h^{t'+t'}\). The authentication head is \(Hd = (K, L_1, L_2, L_3, R_1, R_2, R_3, R_4, R_5, R_6, A_S)\). Finally, \(\mathcal{B}\) sends \((\tilde{A}_S, Hd, CT)\) to \(\mathcal{A}\). Let \(Q_E\) be a set consisting of \((A_S, CT)\).

Sanitizer Query. \(\mathcal{A}\) submits a ciphertext \(CT = (C, C', C'', (C_1, D_1), \cdots , (C_k, D_k), (\mathcal{M}, \rho, \Omega))\). \(\mathcal{B}\) selects \(\vec{r} \xleftarrow{\$} \mathbb{Z}_p\) and computes \(\vec{C} = C \cdot e(g, C')^\vec{r} \cdot h^{\vec{r}}, \vec{C}_1 = C'g^{\vec{r}}, \vec{C}_2 = C'' \cdot C'^\vec{r} \cdot \eta^{\vec{r}}, \vec{C}_3 = \tilde{h}^{\vec{r}}, \vec{C}_1 = C_1 \cdot D_1^{\vec{r}} = h_{\rho(1)}^{-r_1+t'} = g^{a\lambda_1} h_{\rho(1)}^{r_1+t'}, D_1 = D_1 \cdot g^{r_1}, \cdots \), \(\vec{C}_k = C_k \cdot D_k^{\vec{r}} \cdot h_{\rho(k)}^{r_1+t'} = g^{a\lambda_k} h_{\rho(k)}^{r_1+t'}, D_k = D_k \cdot g^{r_1+t'} + g^{-(r_1+t')}\). The sanitized ciphertext is \(\vec{CT} = (\vec{C}, \vec{C}_1, \vec{C}_2, \vec{C}_3, (\vec{C}_1, \vec{D}_1), \cdots , (\vec{C}_k, \vec{D}_k), (\mathcal{M}, \rho, \Omega))\).

Challenge. \(\mathcal{A}\) submits a set of attributes \(A_S\) with \(ID_{S'} \notin \Gamma\), a message \(M^*\) and an access structure \(\mathcal{M}^*\) associated with a set of attributes \(\Omega^*\). \(\mathcal{B}\) selects a random vector \(\vec{v}^* = (s^*, y_2^*, \cdots , y_n^*) \xleftarrow{\$} \mathbb{Z}_p^n\) and \(r_1, r_2, \cdots , r_k \xleftarrow{\$} \mathbb{Z}_p\), and computes \(C = M \cdot e(g, g)^{s^*} \cdot e(g, \bar{Y})^{-s^*}, C' = g^{s^*}, C'' = \eta^s \bar{Y}^{s^*}, (C_1 = g^{a\lambda_1} h_{\rho(1)}^{r_1} \bar{Y}^{-r_1}, D_1 = g^{r_1}), \cdots , (C_k = g^{a\lambda_k} h_{\rho(k)}^{r_1} \bar{Y}^{-r_1}, D_k = g^{r_1})\) where \(\lambda_i = \vec{v}^* \cdot \mathcal{M}_i, i = 1, 2, \cdots , k\). The ciphertext is \(CT^* = (C, C', C'', (C_1, D_1), \cdots , (C_k, D_k), (\mathcal{M}, \rho, \Omega))\). \(\mathcal{B}\) sends \((A_{S^*}, CT^*)\) to \(\mathcal{A}\).

Phase 2. This is the same as in Phase 1 with the limitation that \(ID_{S^*} \notin Q_K\) and \((A_{S^*}, CT^*) \notin Q_E\).

Output. \(\mathcal{A}\) outputs an authentication header \(Hd^* = (K^*, L_1^*, L_2^*, L_3^*, R_1^*, R_2^*, R_3^*, R_4^*, R_5^*, R_6^*, A_{S^*})\) for \((A_{S^*}, CT^*)\). If \(Hd^*\) is valid, \(K^* = g^a g^{at^*} \eta^{\frac{t^*}{\mathbb{M}(A_{S^*})}}\) and \(L_1^* = g^{t^*}\). Hence, \(\Psi = \frac{K^*}{g^{2t^* L_1^*}} = \eta^{\frac{1}{\mathbb{M}(A_{S^*})}}\). Let \(f(x) = \mu(x)(x + H(A_{S^*})) + \rho\) and \(\mu(x) = \mu_0 + \mu_1 x + \cdots + \mu_{2t^*} x^{2t^* - 2}\). We have \(\frac{f(x)}{x + H(A_{S^*})} = \mu(x) + \frac{\rho}{x + H(A_{S^*})}\), \(\Psi^\frac{1}{2} = g^{\frac{1}{\mathbb{M}(A_{S^*})}} = g^{\frac{f(x)}{x + H(A_{S^*})}} = g^{\mu(x)g^{\frac{2t^*}{H(A_{S^*})}}}\), \(g^{\frac{1}{\mathbb{M}(A_{S^*})}} = (\Psi^\frac{1}{2} \cdot g^{\mu(x)})^\frac{1}{2} = g^{\frac{1}{\mathbb{M}(A_{S^*})}} = (\Psi^\frac{1}{2} \cdot g^{\mu(x)})^\frac{1}{2} = g^{\frac{1}{\mathbb{M}(A_{S^*})}}\).
\(\Psi^{\frac{1}{2}} \cdot \prod_{i=0}^{q-2} (\tilde{g}^{-i})^{\mu_i}\). Therefore, \(\mathcal{B}\) can use \((H(A_{S^*}), \tilde{g}^{-H(A_{S^*})})\) to break the \(q\)-SDH assumption with the advantage \(\epsilon(\ell)\).

**Weak Attribute Privacy.** In our scheme, to generate an authentication header, \(S\) selects a subset of his/her attributes \(\tilde{A}_S \subset A_S\) and \(\mathbb{P}(\tilde{A}_S, \Omega) = 1\), and proves that he/she holds these attributes by randomizing his/her secret key. Since the attributes included in \(Hd\) are sent to \(\text{San}\) in plaintext, \(\text{San}\) knows a subset of attributes of \(S\). We assume that a subset of attributes is not enough to identify a real user. This security notion is similar as that introduced in [41, 51].

6. Conclusion

In existing ACE schemes, how to determine whether a communication request should be permitted or denied was not considered. Furthermore, the commutation cost is linear with the number of receivers.

To express flexible policies and enhance IFC, a fine-grained IFC using attributes scheme is proposed in this paper. Our scheme provides the following features: (1) IFC policies are defined over a universe set of attributes; (2) the computation cost to determine whether a communication should be permitted or denied is constant, instead of linear with the number of required attributes or receivers; (3) it can provide weak attribute privacy; (4) fine-grained access structures on encrypted data are supported; (5) the communication cost is linear with the number of required attributes and is independent of the number of receivers.

7. Future Work

In this paper, weak attribute privacy is considered, but the sanitizer can know subsets of attributes of both senders and receivers. Hence, it is interesting to construct a fine-grained IFC using attributes scheme where both the no read-up rule and the no write-down rule are supported and attributes can be fully hidden. We leave this as an open problem and our future work.

Acknowledgements

This work was partially supported by the National Key Research and Development Program of China (Grant No. 2017YFD0401002-3), the Six Talent
Peaks Project of Jiangsu Province, China (Grant No. 2015-DZXX-020) and the National Natural Science Foundation of China (Grant No. 61822202, 61872089, 61300213).

References


