A Verifiable and Fair Attribute-Based Proxy Re-Encryption Scheme for Data Sharing in Clouds

Chunpeng Ge, *Member, IEEE*, Willy Susilo, *Fellow, IEEE*, Joonsang Baek, *Member, IEEE*, Zhe Liu, *Senior Member, IEEE*, Jinyue Xia, and Liming Fang, *Member, IEEE*

Abstract—To manage outsourced encrypted data sharing in clouds, attribute-based proxy re-encryption (ABPRE) has become an elegant primitive. In ABPRE, a cloud server can transform an original recipient's ciphertext to a new one of a shared user's. As the transformation is computation consuming, a malicious cloud server may return an incorrect re-encrypted ciphertext to save its computation resources. Moreover, a shared user may accuse the cloud server of returning an incorrect re-encrypted ciphertext to refuse to pay the cost of using the cloud service. However, existing ABPRE schemes do not support a mechanism to achieve verifiability and fairness. In this article, a novel verifiable and fair attribute-based proxy re-encryption (VF-ABPRE) scheme is introduced to support verifiability and fairness. The verifiability enables a shared user to verify whether the re-encrypted ciphertext returned by the server is correct and the fairness ensures a cloud server escape from malicious accusation if it has indeed conducted the re-encryption operation honestly. Additionally, we conduct a performance experiment to show the efficiency and practicality of the new VF-ABPRE scheme.

Index Terms—Attribute-based proxy re-encryption, data sharing, cloud computing, verifiability, fairness

1 Introduction

Cloud computing, which provides adequate storage and computation capability, has been a prevalent information infrastructure. Instead of managing data locally, cloud services provide an opportunity for users outsourcing their data on the cloud without data management concerns [1]. While it is so convenient to use, data security and privacy especially access control on shared data become a concern since the cloud service is usually provided by third parties [2] such as the Amazon cloud services and Alibaba cloud. Recently, attribute-based encryption (ABE) [3] has been adopted to support flexible access control on user's credential. In a typical ABE scheme, a user's credential and ciphertext are associate with an attribute set and an access policy. The ciphertext can only be decrypted when the attribute set satisfies the access policy.

However, ABE lacks the ability of encrypted data sharing which is critical in the situation of collaboration needed. Therefore, attribute-based proxy re-encryption (ABPRE) was used to enable direct transformation from a ciphertext to another one. And all the cloud server needs to deal with is the encrypted data and a re-encryption key. The ciphertext under one policy would be transformed by re-encryption to a new ciphertext associate with another policy. During the transformation, the cloud server cannot reveal the underlying plaintext. In such a manner, the recipient with access policy P_1 is able to share its outsourced encrypted data with a shared user with access policy P_2. However, a main issue of ABPRE is that the recipient cannot ensure the validity of the re-encrypted ciphertext returned by the cloud server. Since the re-encryption computation's complexity is linear with the size of the access policy, the computation can be very expensive. As a result, the cloud server may reuse a previously generated re-encrypted ciphertext or even a random re-encrypted ciphertext to save its computation resource [4]. Moreover, the current ABPRE schemes cannot achieve the fairness property which prevents the cloud server from a malicious accusation of returning an incorrect re-encrypted ciphertext if it has indeed behaved honestly.

We use the following genome data research system to illustrate the aforementioned problem. A volunteer uploads his personal genome data to an online genomics research center. To protect the confidentiality of his genome data, the

- *Chunpeng Ge is with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211100, China, and with the State Key Laboratory of Cryptology, Beijing 100878, China, with the Science and Technology on Parallel and Distributed Processing Laboratory, Changsha 410000, China, and also with the School of Computing and Information Technology, University of Wollongong, Wollongong, NSW 2522, Australia. E-mail: gecp@nuaa.edu.cn.*
- *Willy Susilo and Joonsang Baek are with the School of Computing and Information Technology, University of Wollongong, Wollongong, NSW 2522, Australia. E-mail: {wsusilo, baek}@uow.edu.au.*
- *Zhe Liu is with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211100, China, and also with the State Key Laboratory of Cryptology, Beijing 100878, China. E-mail: zheliu@nuaa.edu.cn.*
- *Jinyue Xia is with IBM, Durham, NC 27709 USA. E-mail: jinyue.xia@ibm.com.*
- *Liming Fang is with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211100, China. E-mail: fangliming@nuaa.edu.cn.*

Manuscript received 26 Aug. 2020; revised 11 Mar. 2021; accepted 20 Apr. 2021. Date of publication 29 Apr. 2021; date of current version 2 Sept. 2022. (Corresponding authors: Willy Susilo; Zhe Liu.) Digital Object Identifier no. 10.1109/TDSC.2021.3076580

volunteer encrypts his genome data with an access policy P_1: $''Californiacenter''AND(''Brain''OR''Neurology'')AND''Researchscientist''$ that indicates a brain or neurology research scientist from California center can access the genome data. After encryption is completed, he uploads it to the cloud server. Later, another user Alice who satisfies P_1 may need to share this genome data with her colleagues with access policy P_2: $''Washingtoncenter''AND(''Brain''OR''Neurology'')AND''Professor$ to conduct some joint research. With ABPRE, the cloud server can re-encrypt the original ciphertext to a shared user's ciphertext. The scenario is depicted in Fig. 1. The concern is that the shared user cannot assure that the returned re-encrypted ciphertext is a correct re-encrypted ciphertext of the original ciphertext [4]. The cloud server may return an incorrect re-encrypted ciphertext to save its computation cost. Incorrect genome data may lead to a disaster to the research result. Another issue is that, the shared user may accuse the cloud serve of returning an incorrect re-encrypted ciphertext even if the re-encrypted ciphertext is correct [5]. By doing so, the shared user may refuse to pay for the cloud service which is a critical problem for the commercial cloud service system.

At first glance, one may think this issue can be resolved through verifying the re-encryption if the server publishes the re-encryption key and the original ciphertext. By doing so, everyone can repeat the re-encryption operation to verify the correctness of the ciphertext returned by the server. However, such a manner is impractical as the re-encryption operation cost is expensive, it is not suitable for a verifier to repeat this computation costly work. Another trivial solution is that a shared user can reveal his private key so that the original recipient can decrypt the returned re-encrypted ciphertext and then he can validate the plaintext is original. However, revealing private key may result in critical security issues to the shared user.

1.1 Motivations

Although the existing ABPRE schemes can protect data confidentiality and enable fine-grained data sharing for outsourced encrypted data, they cannot provide verifiability and fairness. Thus, we need a mechanism to achieve verifiability and fairness for ABPRE while keeping its confidentiality. More specifically, we need to design an ABPRE scheme that:

1) enables fine-grained encrypted data sharing while keeping the confidentiality of the underlying data;
2) enables the shared user to verify the correctness of the re-encrypted ciphertext returned from the cloud server;
3) protects the cloud server to escape from a malicious accusation if it has returned a correct re-encrypted ciphertext.

Therefore, we introduce a verifiable and fair attribute-based proxy re-encryption (VF-ABPRE) scheme for cloud computing, whereby the shared user can verify the correctness of the re-encrypted ciphertext, and the server can protect itself from a malicious accusation.

1.2 Related Work

*ABE.*In a typical ABE [3] setup, a user only can access the plaintext from a ciphertext if his credential satisfies the access policy. According to the manner of a secret key/ciphertext is generated, ABE is categorized into ciphertext-policy ABE (CP-ABE) [6] and key-policy ABE (KP-ABE) [7]. In CP-ABE, a ciphertext is generated with an access policy and a secret key is with attribute set, while KP-ABE is in contrast. As illustrated in [7], KP-ABE is suitable for the scenarios where the authority decides who are the authorized users, such as the pay-TV system [7]. In contrast, CP-ABE fits well for the cloud sharing system as the user can appoint the access to the recipients [6]. Following their work, the work on ABE have addressed concerns on the efficiency [8], [9], [10], [11], [12], [13], security [14], [15], [16], [17], [18] and anonymity [19], [20], [21], [22].

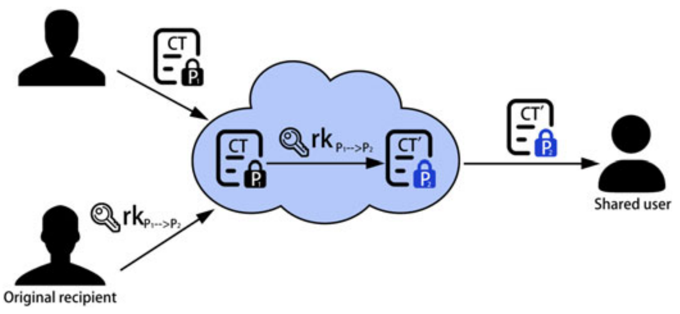


Fig. 1. System model of ABPRE.

ABPRE. In proxy re-encryption [23], [24], a ciphertext is transformed to a new one of another user's by a proxy using a re-encryption key. It allows a user to share its encryption to others without disclosing the plaintext to the proxy. To detect whether the proxy has the maliciously behaved, Ohata *et al.* [25] presented a mechanism to verify the output of the proxy. However, their scheme cannot provide a fine-grained access control on encrypted data. Moreover, in their scheme, the original recipient's public key is required for the verification of the re-encrypted ciphertext, which also means the shared user must identify each re-encrypted ciphertext with the original recipient. What's important is that it lacks a mechanism to protect the proxy from malicious accusation. In the meanwhile, Liang *et al.* [26] introduced this notion to the attribute-based setting and thus achieves a flexible description on the encrypted data. Following their work, many ABPRE schemes were proposed to extend the expressiveness of the access policy [27], [28], [29] and enhance the security model [30], [31], [32]. Unfortunately, none of above ABPRE works have considered the verifiability and the fairness properties which are critical for an outsourced re-encryption system.

1.3 Our Contribution

We present a VF-ABPRE scheme, and our contributions are:

- First, we present a formal definition of VF-ABPRE which considers the verifiability and fairness of attribute-based encrypted data sharing in cloud computing.
- Then, we construct a concrete VF-ABPRE scheme and prove its confidentiality, verifiability and fairness.
- Finally, we conduct an implementation to evaluate the performance of our proposed scheme to demonstrate its practicality and efficiency.

2 System Architecture and Definitions

We start with the system architecture and work flow of the proposed VF-APPRE scheme. The algorithms and security model are illustrated afterwords. Note that we take the ciphertext-policy setting into consideration, however, our method is applicable in the key-policy setting.

2.1 System Architecture

A verifiable and fair ciphertext-policy attribute-based proxy re-encryption (VF-CP-ABPRE) system includes the following entities, data owner, original recipient, a cloud server act as a proxy and shared recipients. In addition, there is an authority center that initializes the system and issues private keys for users.

- The authority center initializes the system by generating system parameters and issuing private keys for participants.
- A user encrypts his personal data under an access policy and stores the resulting ciphertext on cloud.
- The original recipient, who wants to share encrypted data that originally sent to him with some other shared users, generates a re-encryption key which will be used by the cloud server to convert his ciphertext for the shared users'.
- The cloud server (e.g., AWS) stores the ciphertexts, and conducts the re-encryption operations.
- The shared user can decrypt a re-encrypted ciphertext as normal. Additionally, he can verify whether the re-encrypted ciphertext is correct. If it is incorrect, he is able to present a proof to show that the re-encryption is an invalid one.

2.2 Threat Model

In our treat model, we consider the following threats:

- An adversary including the cloud server without a valid private key may want to reveal the plaintext from a ciphertext (including the original ciphertext and the re-encrypted ciphertext). This is type of attack is viewed as a threat to data confidentiality.
- A malicious may want to return an incorrect re-encrypted ciphertext without been detected by the shared user. This type of attack is viewed as a threat to the verifiability.
- A shared user may want to make a wrong claim to accuse that the cloud server has returned an incorrect re-encrypted ciphertext even if the cloud server has returned a correct one. This type of attack is viewed as a threat to the fairness.

2.3 Linear Secret Sharing Scheme

A Linear secret sharing scheme (LSSS) Π is linear over a set of parties (over Z_p) if:

- every party's share is a vector over Z_p.
- There exits an $l \times n$ matrix $\mathbb{A}$, and a function ρ, where $\rho(j) \in \mathcal{P}$, $j \in \{1, \ldots, l\}$. Denote r is the shared secret, and $r_2, \ldots, r_n \in Z_p$ are random values from Z_p. Let $\mu = (r, r_2, \ldots, r_n)$ is a column vector, then $\mathbb{A}_j \cdot \mu$, $j \in [1, l]$ is l shares of r related to Π, and $\mathbb{A}_j \cdot \mu$ belongs to $\rho(j)$.

The linear reconstruction property of LSSS scheme is defined as follows. Suppose S is an authorized attribute set. J is the set that $J = \{j : \rho(j) \in S\} \subset \{1, \ldots, l\}$. Then there exist constants $\{\theta_j\}_{j \in J}$, such that $\sum_{j \in J} \theta_j \cdot \mathbb{A}_j = (1, 0, \ldots, 0)$, and $\sum_{j \in J} \theta_j \mathbb{A}_j \cdot \mu = r$.

2.4 VF-CP-ABPRE Definition

Definition 1. *A VF-CP-ABPRE scheme is composed of the following steps.*

- *$Setup(\lambda, U)$: A trusted authority executes $Setup$ to initialize the system. A security parameter λ and the attribute universe U are the input to $Setup$. The system's public parameter PP and master secret key msk are generated at this phase.*
- *$KeyGen(msk, S)$: $KeyGen$ is also executed by the authority party. This step outputs the secret keys for participants. It takes the master key msk and an attribute set S as the input, and generates the secret key sk for S.*
- *$Enc(m, (\mathbb{A}, \rho))$: This step encrypts a message m with an access policy $(\mathbb{A}, \rho)$ and generates an original ciphertext CT.*
- *$ReKeyGen(sk, (\widetilde{\mathbb{A}}, \widetilde{\rho}))$: This algorithm uses a private key sk of attribute set S and an access policy $(\widetilde{\mathbb{A}}, \widetilde{\rho})$, where $R(S, (\widetilde{\mathbb{A}}, \widetilde{\rho})) = 0$[1] and outputs a re-encryption key rk.*
- *$ReEnc(rk, CT)$: A re-encryption key rk and an original ciphertext CT are the inputs, $ReEnc(rk, CT)$ outputs a re-encrypted ciphertext CT'.*
- *$Dec_{or}(sk, CT)$: This step is to decrypt an original ciphertext CT. Inputs a secret key sk and an original ciphertext CT, outputs the plaintext m if CT is a valid ciphertext or abort $\perp$ otherwise.*
- *$Dec_{re}(sk, CT', CT)$: The Dec_{re} algorithm is leveraged to decrypt a re-encrypted ciphertext CT'. Inputs a secret key sk, an original ciphertext CT and its re-encrypted ciphertext CT', outputs the plaintext m if CT' is a valid re-encrypted ciphertext of CT or $\perp$ otherwise.*
- *$Claim(CT, rk, CT', \pi)$: The $Claim$ algorithm is executed between a shared user and a public verifier. The input of the shared user and the public verifier is (CT, rk, CT'). The shared user outputs a proof π to claim that CT' returned by the cloud server is an incorrect re-encrypted ciphertext of CT under rk. The public verifier verifies the claim and outputs $true$ if the claim is correct which means CT' is an incorrect re-encrypted ciphertext of CT. Otherwise, outputs $flase$ that indicates the claim is incorrect.*

In the above definition, there is the public parameter as part of the input for each algorithm and is omitted for simplicity.

2.5 Security Definitions

As illustrated in the threat model, there are three types of security properties which are confidentiality, verifiability and fairness need to be considered. First, we use the semantic

1. $R(S, (\mathbb{A}, \rho)) = 0$ means the attribute set S doesn't satisfy the access policy $(\mathbb{A}, \rho)$, and $R(S, (\mathbb{A}, \rho)) = 1$ means S satisfies $(\mathbb{A}, \rho)$.

security of the ciphertext to describe the confidentiality. Then, we user the verifiability and fairness game to describe the verifiability and fairness respectively. In our security model, we adopted the selective model, which means that the adversary needs to commit the challenge policy before the security game, as in [15].

Semantic Security. In a VF-CP-ABPRE scheme, there are two types of ciphertexts: the original ciphertext and the re-encrypted ciphertext. We will user the Semantic-Or security game to describe the semantic security of the original ciphertext, and Semantic-Re to describe the semantic security of the re-encrypted ciphertext.

Game Semantic-Or. A VF-CP-ABPRE scheme is original ciphertext semantic secure in the selective model if the advantage of an adversary $\mathcal{A}$ in the following game is negligible.

- Init. The attacker $\mathcal{A}$ chooses $(\mathbb{A}^*, \rho^*)$ as the challenge access policy.
- Setup. In this step, a challenger C runs the $Setup$ to generate PP and msk. Then, C passes public parameters to $\mathcal{A}$.
- Query phase I. $\mathcal{A}$ queries:
  1) $\mathcal{O}_{sk}(S)$: $\mathcal{A}$ queries on attribute set S, challenger $\mathcal{C}$ executes $sk = KeyGen(msk, S)$ and returns sk to $\mathcal{A}$.
  2) $\mathcal{O}_{rk}(S, (\widetilde{\mathbb{A}}, \widetilde{\rho}))$: $\mathcal{A}$ makes a re-encryption key query on $(S, (\widetilde{\mathbb{A}}, \widetilde{\rho}))$, where $R(S, (\widetilde{\mathbb{A}}, \widetilde{\rho})) = 0$. $\mathcal{C}$ runs $sk = KeyGen(msk, S)$ and $rk = ReKeyGen(sk, (\widetilde{\mathbb{A}}, \widetilde{\rho}))$, returns rk to $\mathcal{A}$.
  3) $\mathcal{O}_{re}(CT, S, (\widetilde{\mathbb{A}}, \widetilde{\rho}))$: $\mathcal{A}$ makes a re-encryption query on $(CT, S, (\widetilde{\mathbb{A}}, \widetilde{\rho}))$, $\mathcal{C}$ executes $sk = KeyGen(msk, S)$, $rk = ReKeyGen(sk, (\widetilde{\mathbb{A}}, \widetilde{\rho}))$ and $CT' = ReEnc(CT, rk)$, returns CT' to $\mathcal{A}$.

During Query phase I, $\mathcal{A}$ cannot make the following queries:
  1) $\mathcal{O}_{sk}(S)$ if $R(S, (\mathbb{A}^*, \rho^*)) = 1$;
  2) $\mathcal{O}_{rk}(S, (\widetilde{\mathbb{A}}, \widetilde{\rho}))$ if $R(S, (\mathbb{A}^*, \rho^*)) = 1$ and $\mathcal{A}$ has queried $\mathcal{O}_{sk}(\widetilde{S})$ where $R(\widetilde{S}, (\widetilde{\mathbb{A}}, \widetilde{\rho})) = 1$.
- Challenge. $\mathcal{A}$ chooses two plaintext (m_0, m_1) with equal length and sends them to the challenger $\mathcal{C}$. $\mathcal{C}$ computes the challenge ciphertext $CT^* = Enc(m_\sigma, (\mathbb{A}^*, \rho^*))$ where $\sigma \in \{0, 1\}$, and returns it to $\mathcal{A}$.
- Query phase II. $\mathcal{A}$ queries like in Query phase I except:
  1) $\mathcal{O}_{sk}(S)$ if $R(S, (\mathbb{A}^*, \rho^*)) = 1$;
  2) $\mathcal{O}_{rk}(S, (\widetilde{\mathbb{A}}, \widetilde{\rho}))$ and $\mathcal{O}_{sk}(\widetilde{S})$, if $R(S, (\mathbb{A}^*, \rho^*)) = 1$ and $R(\widetilde{S}, (\widetilde{\mathbb{A}}, \widetilde{\rho})) = 1$.
  3) $\mathcal{O}_{re}(CT^*, S, (\widetilde{\mathbb{A}}, \widetilde{\rho}))$ and $\mathcal{O}_{sk}(\widetilde{S})$, if $R(S, (\mathbb{A}^*, \rho^*)) = 1$ and $R(\widetilde{S}, (\widetilde{\mathbb{A}}, \widetilde{\rho})) = 1$.
- Guess. $\mathcal{A}$ outputs its guess σ'.

The adversary $\mathcal{A}$'s advantage to win the semantic-or game is defined as

$$Adv_{\mathcal{A}}^{Sem-Or}(\lambda) = |Pr[\sigma' = \sigma] - 1/2|.$$

Game Semantic-Re. A VF-CP-ABPRE scheme is re-encrypted ciphertext semantic secure in the selective model if the advantage of an adversary $\mathcal{A}$ in the following game is negligible.

- Init. $\mathcal{A}$ chooses $(\mathbb{A}^*, \rho^*)$ as the challenging access policy.
- Setup. In this step, a challenger C runs the $Setup$ to generate PP and msk.. Then, C passes public parameters to $\mathcal{A}$.
- Query phase I. $\mathcal{A}$ queries:
  1) $\mathcal{O}_{sk}(S)$: $\mathcal{A}$ queries on attribute set S, challenger $\mathcal{C}$ executes $sk = KeyGen(msk, S)$, then sends sk to $\mathcal{A}$.
  2) $\mathcal{O}_{rk}(S, (\widetilde{\mathbb{A}}, \widetilde{\rho}))$: $\mathcal{A}$ makes a re-encryption key query on $(S, (\widetilde{\mathbb{A}}, \widetilde{\rho}))$, where $R(S, (\widetilde{\mathbb{A}}, \widetilde{\rho})) = 0$. $\mathcal{C}$ executes $sk = KeyGen(msk, S)$ and $rk = ReKeyGen(sk, (\widetilde{\mathbb{A}}, \widetilde{\rho}))$, sends rk to $\mathcal{A}$.

During Query phase I, $\mathcal{A}$ cannot make the $\mathcal{O}_{sk}(S)$ query if $R(S, (\mathbb{A}^*, \rho^*)) = 1$.

- Challenge. $\mathcal{A}$ chooses two plaintext (m_0, m_1) with equal length and sends them to the challenger $\mathcal{C}$. $\mathcal{C}$ computes the challenge ciphertext $CT'^* = ReEnc(Enc(m_\sigma, (\mathbb{A}, \rho)), rk)$, where $\sigma \in \{0, 1\}$, $rk = ReKeyGen(S, (\mathbb{A}^*, \rho^*))$, $R(S, (\mathbb{A}, \rho)) = 1$, and returns it to $\mathcal{A}$.
- Query phase II. $\mathcal{A}$ queries like in Query phase I except the query $\mathcal{O}_{sk}(S)$ if $R(S, (\mathbb{A}^*, \rho^*)) = 1$;
- Guess. $\mathcal{A}$ outputs its guess σ'.

Note that, as there is no limitation on the re-encryption, anyone can answer the re-encryption query by issuing a re-encryption key query first to get the relevant re-encryption key and then answer the re-encryption query. Hence, we omit the re-encryption query in the above game.

The adversary $\mathcal{A}$'s advantage to win the semantic-re game is defined as

$$Adv_{\mathcal{A}}^{Sem-Re}(\lambda) = |Pr[\sigma' = \sigma] - 1/2|.$$

Definition 2 (Semantic Security) *A VF-CP-ABPRE scheme is semantic secure if it is Semantic-Or secure and Semantic-Re secure.*

Verifiability. The verifiability of a VF-CP-ABPRE scheme is captured by the following game.

- Setup. In this step, a challenger C runs the $Setup$ algorithm and gets the public parameters and master secret key. Then, C passes public parameters to $\mathcal{A}$.
- Query phase I. $\mathcal{A}$ queries:
  1) $\mathcal{O}_{sk}(S)$: $\mathcal{A}$ queries on attribute set S, challenger $\mathcal{C}$ executes $sk = KeyGen(msk, S)$, and sends sk to $\mathcal{A}$.
  2) $\mathcal{O}_{rk}(S, (\widetilde{\mathbb{A}}, \widetilde{\rho}))$: $\mathcal{A}$ makes a re-encryption key query on $(S, (\widetilde{\mathbb{A}}, \widetilde{\rho}))$, where $R(S, (\widetilde{\mathbb{A}}, \widetilde{\rho})) = 0$. $\mathcal{C}$ executes $sk = KeyGen(msk, S)$ and $rk = ReKeyGen(sk, (\widetilde{\mathbb{A}}, \widetilde{\rho}))$, sends rk to $\mathcal{A}$.
  3) $\mathcal{O}_{claim}(CT, rk, CT', \pi)$: $\mathcal{A}$ makes a claim key query on (CT, rk, CT', π). The challenger $\mathcal{C}$ interacts with the adversary $\mathcal{A}$ and outputs the result of algorithm $Claim(CT, rk, CT', \pi)$.
- Challenge. $\mathcal{A}$ chooses a message m^* and access policy $(\mathbb{A}^*, \rho^*)$ and sends them to $\mathcal{C}$. $\mathcal{C}$ computes $CT^* = Enc(m^*, (\mathbb{A}^*, \rho^*))$ and sends CT^* to $\mathcal{A}$.
- Query phase II. $\mathcal{A}$ queries like in Query phase I.

- Guess. $\mathcal{A}$ outputs an attribute set S^* where $R(S^*, (\mathbb{A}^*, \rho^*)) = 1$, and a re-encrypted ciphertext CT'^*. The adversary $\mathcal{A}$ wins if $Dec_{re}(S^*, CT^*, CT'^*) \notin \{m^*, \perp\}$.

The adversary $\mathcal{A}$'s advantage to win the verifiability security game is defined as

$$Adv_{\mathcal{A}}^{Ver}(\lambda) = |Pr[\mathcal{A}\ wins|Ver].$$

Definition 3 (Verifiability) *A VF-CP-ABPRE scheme is verifiable if for all PPT adversary $\mathcal{A}$, its advantage $Adv_{\mathcal{A}}^{Ver}$ is negligible.*

Fairness. The Fairness of a VF-CP-ABPRE scheme is captured by the following game.

- Setup. In this step, a challenger C runs the $Setup$ algorithm and gets the public parameters and master secret key. Then, C passes public parameters to $\mathcal{A}$.
- Query phase I. $\mathcal{A}$ queries:
  1) $\mathcal{O}_{sk}(S)$: $\mathcal{A}$ queries on attribute set S, challenger $\mathcal{C}$ executes $sk = KeyGen(msk, S)$, and sends sk to $\mathcal{A}$.
  2) $\mathcal{O}_{rk}(S, (\widetilde{\mathbb{A}}, \widetilde{\rho}))$: $\mathcal{A}$ makes a re-encryption key query on $(S, (\widetilde{\mathbb{A}}, \widetilde{\rho}))$, where $R(S, (\widetilde{\mathbb{A}}, \widetilde{\rho})) = 0$. $\mathcal{C}$ executes $sk = KeyGen(msk, S)$ and $rk = ReKeyGen(sk, (\widetilde{\mathbb{A}}, \widetilde{\rho}))$, sends rk to $\mathcal{A}$.
  3) $\mathcal{O}_{claim}(CT, rk, CT', \pi)$: $\mathcal{A}$ makes a claim key query on (CT, rk, CT', π). The challenger $\mathcal{C}$ interacts with the adversary $\mathcal{A}$ and outputs the result of algorithm $Claim(CT, rk, CT', \pi)$.
- Challenge. $\mathcal{A}$ chooses a message m^* and access policy $(\mathbb{A}^*, \rho^*)$ and sends them to $\mathcal{C}$. $\mathcal{C}$ computes $CT^* = Enc(m^*, (\mathbb{A}^*, \rho^*))$ and $CT'^* = ReEnc(CT^*, rk^*)$. Then $\mathcal{C}$ sends CT^* and CT'^* to $\mathcal{A}$. Note that, rk^* is a re-encryption key from $(\mathbb{A}^*, \rho^*)$ to a new access policy $(\widetilde{\mathbb{A}}, \widetilde{\rho})$.
- Query phase II. $\mathcal{A}$ queries like in Query phase I.
- Guess. $\mathcal{A}$ outputs a proof π and wins the game if $Claim(CT^*, CT'^*, \pi) = true$.

The adversary $\mathcal{A}$'s advantage to win the verifiability security game is defined as

$$Adv_{\mathcal{A}}^{Fair}(\lambda) = |Pr[\mathcal{A}\ wins|Fair].$$

Definition 3 (Fairness) *A VF-CP-ABPRE scheme is fair if for all PPT adversary $\mathcal{A}$, its advantage $Adv_{\mathcal{A}}^{Fair}$ is negligible.*

3 Preliminaries

3.1 Negligible Function

If for $\forall t > 0$, there exists a x_t such that $f(x) < 1/x^t$ for $\forall x > x_t$, the $f(x)$ is a negligible function.

3.2 Bilinear Pairing

A bilinear pairing is a tuple (e, G, G_T, g, p) where

- $e(U^s, V^t) = e(U, V)^{st}$ for all $U, V \in G$ and $s, t \in Z_p^*$,
- $e(U, V) \neq 1$,
- $e(U, V)$ can be efficiently computed for all $U, V \in G$,

where G and G_T are multiplicative cyclic groups with generator g and order p.

3.3 Cryptographic Hash Function

A hash function H is a cryptographic hash function, if:

1) given a value y, the probability of finding a value x, where $H(x) = y$ is negligible;
2) given $H(a)$, the probability of finding a value b where $b \neq a$ and $H(b) = H(a)$ is negligible;
3) the probability of finding two values a, b such that $b\ / = a$ and $H(a) = H(b)$ is negligible;

3.4 Complex Assumption

Let (e, G, G_T, g, p) is a bilinear pairing and $a, r \in Z_p$, $g, \in G$ are random elements, given an instance $\vec{v} =$

$$g, g^r, g^a, \ldots, g^{a^q}, g^{a^{q+2}}, \ldots, g^{a^{2q}},$$

$$\forall j \in [1, q] \quad g^{r \cdot b_j}, g^{a/b_j}, \ldots, g^{a^q/b_j}, g^{a^{q+2}/b_j}, \ldots, g^{a^{2q}/b_j},$$

$$\forall j, t \in [1, q], t \neq j, \quad g^{a \cdot r \cdot b_t/b_j}, \ldots, g^{a^q \cdot r \cdot b_t/b_j},$$

it is hard to distinguish $e(g, g)^{r \cdot a^{q+1}}$ from a random value $T \in G_T$. Formally, the advantage of a PPT algorithms $\mathcal{B}$

$$|Pr[\mathcal{B}(\vec{v}, e(g, g)^{r \cdot a^{q+1}}) = 1] - Pr[\mathcal{B}(\vec{v}, T) = 1]|,$$

is negligible.

The decisional q-parallel BDHE assumption holds if no polynomial probability time adversary can solve the above problem with a non-negligible advantage.

3.5 Discrete Logarithm (DL) Assumption

Let (e, G, G_T, g, p) is a bilinear pairing, given a tuple $(e, G, G_T, p, g, g^\beta)$ where $g \in G, \beta \in Z_p^*$, the discrete logarithm assumption means that the advantage of a probability polynomial time (PPT) adversary $\mathcal{A}$ to find the integer β is negligible. Formally, the advantage of a PPT adversary $\mathcal{A}$

$$Pr[Adv_{DL}] = Pr[\mathcal{A}(e, G, G_T, p, g, g^\beta) = \beta],$$

is negligible.

4 Proposed VF-CP-ABPRE Construction

To begin with, we describe what challenges we are facing and then introduce our techniques that will be used in our VF-CP-ABPRE construction. Finally, we prove its semantic security, verifiability and fairness.

4.1 Challenges and Our Techniques

In an ABE scheme, if a shared user needs to verify whether the re-encrypted ciphertext returned by the cloud server is a correct one, a straightforward way is that the original recipient and the shared user can reveal their private key. Thus, anyone can check whether the returned re-encrypted is valid. However, this violates the rule of data confidentiality. To address this, we leverage the technique of commitment. In our scheme, when generating the original ciphertext, a commitment of the plaintext is also generated. This commitment with the re-encryption will be sent to the shared user.

Thus, the shared user can first retrieve the plaintext by normal decryption and then check the plaintext against the commitment. In such a manner, the shared user can verify the validity of the re-encryption. However, this approach assumes the shared user's private key is available for the verification. The malicious attacker may issue an incorrect plaintext m and claim that m is the underlying plaintext of the re-encrypted ciphertext. The problem is that a verifier cannot assure that m is indeed the corresponding plaintext as no one else could make the decryption except the shared user. If the shared user publishes his private key, the data confidentiality will be destroyed.

In our approach, we leverage the message-lock encryption (MLE) [33] technique to prevent the shared user from revealing an unrelated m. MLE guarantees that the same plaintext corresponds to the identical re-encrypted ciphertext. Thus, a plaintext m was bind with a re-encrypted ciphertext. Whenever a shared user returns a plaintext m, it can be publicly verified that m is indeed the plaintext of the re-encrypted ciphertext.

4.2 Our VF-CP-ABPRE Construction

- $Setup(\lambda, U)$: The authority center generates a bilinear pairing tuple (e, G, G_T, G, p). Denote the message space as $\{0,1\}^k$. Randomly chooses $\alpha, a \in Z_p^*$, $g, f_1, \ldots, f_U, \phi, \varphi, Q \in G$. The authority center also chooses two cryptographic hash functions $H_1 : G_T \rightarrow \{0,1\}^{2k}$, $H_2 : \{0,1\}^* \rightarrow Z_p^*$, and a message-lock encryption algorithm MLE. Sets $msk = g^\alpha$ and $PP = (e, G, G_T, g, f_1, \ldots, f_U, \phi, \varphi, Q, g^a, e(g,g)^\alpha, H_1, H_2, MLE)$.

  Note that, the hash function H_1 and H_2 can be constructed based on the standard cryptographic hash functions such as SHA-1. Specifically, given an element $x \in G_T$ as the input of H_1, H_1 first converts x into a string type. Then, it takes the string as input of SHA-1 and thus gets a fixed length output. For H_2, it first calls SHA-1 to get a fixed length string, and then converts the string into an element in Z_p.
- $KeyGen(msk, S)$: Randomly choose $s \in Z_p^*$, and sets

$$sk = (S,\ K_1 = g^\alpha g^{as},\ K_2 = g^s,\ \forall x \in S\ K_x = f_x^s).$$

- $Enc(m, (\mathbb{A}, \rho))$: Let $(\mathbb{A}, \rho)$ is an $l \times n$ matrix and ρ associates each row of $\mathbb{A}$ with an attribute in U. Randomly choose $R \in \{0,1\}^k$, and computes $r = H_2(MLE(m||R))$ with the message-lock encryption algorithm MLE. Sets a random vector $\vec{\mu} = (r, y_2, \ldots, y_n) \in Z_p^n$, where $y_2, \ldots, y_n$ are randomly chosen from Z_p. For each row $\mathbb{A}_j$ of $\mathbb{A}$, computes $\lambda_j = \vec{\mu} \cdot \mathbb{A}_j$, $j \in [1, l]$. Randomly chooses $r_j \in Z_p$ for each $j \in [1, l]$. Then computes

$$C = (m||R) \oplus H_1(e(g,g)^{\alpha r}), \quad C_1 = g^r, \quad C_2 = Q^r,$$

$$C_{3,j} = g^{a\lambda_j} f_{\rho(j)}^{-r_j}, \quad C_{4,j} = g^{r_j}\ \forall j \in [1, l],$$

$$\overline{C} = \phi^{H_2(m)} \varphi^{H_2(R)}.$$

  Outputs the ciphertext as $CT = ((\mathbb{A}, \rho), C, C_1, C_2, \{C_{3,j}, C_{4,j}\}_{j \in [1,l]}, \overline{C})$.
- $ReKeyGen(sk, (\widetilde{\mathbb{A}}, \widetilde{\rho}))$: On input $sk = (S, K_1, K_2, K_x\ x \in S)$ and an access policy $(\widetilde{\mathbb{A}}, \widetilde{\rho})$, where $\widetilde{\mathbb{A}}$ is a $\widetilde{l} \times \widetilde{n}$ matrix. Randomly chooses $\chi \in G_T$, $\delta \in Z_p^*$ and computes

$$rk_0 = e(g,g)^{\alpha H_2(\chi)}, \qquad rk_1 = K_1^{H_2(\chi)} \cdot Q^\delta,$$

$$rk_2 = g^\delta, \qquad rk_3 = K_2^{H_2(\chi)},$$

$$rk_{4,x} = K_x^{H_2(\chi)}\ \forall x \in S.$$

  Selects a random vector $\widetilde{\vec{\mu}} = (\widetilde{r}, \widetilde{y_2}, \ldots, \widetilde{y_{\widetilde{n}}}) \in Z_p^{\widetilde{n}}$. For each row $\widetilde{\mathbb{A}}_j$ of $\widetilde{\mathbb{A}}$, computes $\widetilde{\lambda}_j = \widetilde{\vec{\mu}} \cdot \widetilde{\mathbb{A}}_j$, $j \in [1, \widetilde{l}]$. Randomly chooses $\widetilde{r_j} \in Z_p$ for each $j \in [1, \widetilde{l}]$. Then computes

$$rk_5 = \chi \cdot e(g,g)^{\alpha \widetilde{r}}, \quad rk_6 = g^{\widetilde{r}},$$

$$rk_{7,j} = g^{a\widetilde{\lambda}_j} f_{\widetilde{\rho}(j)}^{-\widetilde{r}_j}, \quad rk_{8,j} = g^{\widetilde{r}_j}\ \forall j \in [1, \widetilde{l}].$$

  Output the re-encryption as key $rk = (rk_0, rk_1, rk_2, rk_3, \{rk_{4,x}\}_{x \in S}, rk_5, rk_6, \{rk_{7,j}, rk_{8,j}\}_{j \in [1,\widetilde{l}]}, (\widetilde{\mathbb{A}}, \widetilde{\rho}))$.
- $ReEnc(rk, CT)$: On input an original ciphertext $CT = ((\mathbb{A}, \rho), C, C_1, C_2, \{C_{3,j}, C_{4,j}\}_{j \in [1,l]}, \overline{C})$ and a re-encryption key $rk = (rk_0, rk_1, rk_2, rk_3, \{rk_{4,x}\}_{x \in S}, rk_5, rk_6, \{rk_{7,j}, rk_{8,j}\}_{j \in [1,\widetilde{l}]}, (\widetilde{\mathbb{A}}, \widetilde{\rho}))$.

  If $R(S, (\mathbb{A}, \rho)) = 0$, outputs $\perp$. Otherwise, let $J = \{j : \rho(j) \in S\} \subset \{1, \cdots l\}$, finds elements $\theta_j \in Z_p^*$, such that $\sum_{j \in J} \theta_j \cdot \mathbb{A}_j = (1, 0, \ldots, 0)$. Then, computes

$$C_0' = \frac{e(rk_1, C_1) \cdot e(rk_2, C_2)^{-1}}{\prod_{j \in J}(e(rk_3, C_{3,j}) \cdot e(rk_{4,\rho(j)}, C_{4,j}))^{\theta_j}}.$$

  Sets $\overline{C}' = \overline{C}$, $C' = C$, $C_1' = rk_5$, $C_2' = rk_6$, $C_{3,j}' = rk_{7,j}, C_{4,j}' = rk_{8,j}\ j \in [1, \widetilde{l}]$, $C_5' = rk_0$. Outputs the re-encrypted ciphertext $CT' = (C', \overline{C}', C_0', C_1', C_2', \{C_{3,j}', C_{4,j}'\}_{j \in [1,\widetilde{l}]}, C_5', (\widetilde{\mathbb{A}}, \widetilde{\rho}))$.
- $Dec_{or}(sk, CT)$: Inputs a secret key $sk = (S, K_1, K_2, K_x\ x \in S)$ and an original ciphertext $CT = ((\mathbb{A}, \rho), C, C_1, C_2, \{C_{3,j}, C_{4,j}\}_{j \in [1,l]}, \overline{C})$. Checks whether $R(S, (\mathbb{A}, \rho)) = 1$. If not, outputs $\perp$. Otherwise, let $J = \{j : \rho(j) \in S\} \subset \{1, \ldots l\}$, finds elements $\theta_j \in Z_p^*$, such that $\sum_{j \in J} \theta_j \cdot \mathbb{A}_j = (1, 0, \ldots, 0)$. Then, computes

$$\Phi = \frac{e(K_1, C_1)}{\prod_{j \in J}(e(K_2, C_{3,j}) \cdot e(K_{\rho(j)}, C_{4,j}))^{\theta_j}}, \tag{1}$$

  and

$$m||R = C \oplus H_1(\Phi).$$

  Outputs m if $\overline{C} = \phi^{H_2(m)} \varphi^{H_2(R)}$. Otherwise, outputs $\perp$.
- $Dec_{re}(sk, CT', CT)$: On input a private key $sk = (S, K_1, K_2, K_x\ x \in S)$, a re-encrypted ciphertext $CT' =$

$(C', \overline{C}', C'_0, C'_1, C'_2, \{C'_{3,j}, C'_{4,j}\}_{j\in[1,\widetilde{l}]}, C'_5, (\widetilde{\mathbb{A}}, \widetilde{\rho}))$ and an original ciphertext $CT = ((\mathbb{A}, \rho), C, C_1, C_2, \{C_{3,j}, C_{4,j}\}_{j\in[1,l]}, \overline{C})$. Checks whether $R(S, (\widetilde{\mathbb{A}}, \widetilde{\rho})) = 1$. If not, outputs $\perp$. Otherwise, let $J = \{j : \widetilde{\rho}(j) \in S\} \subset \{1, \dots \widetilde{l}\}$, finds elements $\theta_j \in Z_p^*$, such that $\sum_{j\in J} \theta_j \cdot \widetilde{\mathbb{A}}_j = (1, 0, \dots, 0)$. Then, computes

$$\chi = C'_1 / \frac{e(K_1, C'_2)}{\prod_{j\in J}(e(K_2, C'_{3,j}) \cdot e(K_{\widetilde{\rho}(j)}, C'_{4,j}))^{\theta_j}}, \quad (2)$$

and

$$m||R = C' \oplus H_1(C_0'^{\frac{1}{H_2(\chi)}}).$$

Outputs m if $\overline{C}' = \overline{C} = \phi^{H_2(m)} \varphi^{H_2(R)}$. Otherwise, outputs $\perp$.

- $Claim(CT, rk, CT', \pi)$: On input an original ciphertext $CT = ((\mathbb{A}, \rho), C, C_1, C_2, \{C_{3,j}, C_{4,j}\}_{j\in[1,l]}, \overline{C})$, a re-encryption key $rk = (rk_0, rk_1, rk_2, rk_3, \{rk_{4,x}\}_{x\in S}, rk_5, rk_6, \{rk_{7,j}, rk_{8,j}\}_{j\in[1,\widetilde{l}]}, (\widetilde{\mathbb{A}}, \widetilde{\rho}))$, a re-encrypted ciphertext $CT' = (C', \overline{C}', C'_0, C'_1, C'_2, \{C'_{3,j}, C'_{4,j}\}_{j\in[1,\widetilde{l}]}, C'_5, (\widetilde{\mathbb{A}}, \widetilde{\rho}))$ and a proof $\pi = (m, R)$ that published by a shared user. The verifier first checks whether

$$\begin{cases} \overline{C}' \overset{?}{=} \overline{C} \\ C' \overset{?}{=} C, \\ C'_1 \overset{?}{=} rk_5, \\ C'_2 \overset{?}{=} rk_6, \\ C'_{3,j} \overset{?}{=} rk_{7,j} \ \forall j \in [1, \widetilde{l}], \\ C'_{4,j} \overset{?}{=} rk_{8,j} \ \forall j \in [1, \widetilde{l}], \\ C'_5 \overset{?}{=} rk_0. \end{cases} \quad (3)$$

If one of the above equations in (3) does not hold, abort and outputs $ture$.

Otherwise, computes $r = H_2(MLE(m||R))$ and check whether

$$\overline{C}' = \overline{C} = \phi^{H_2(m)} \varphi^{H_2(R)}, \quad (4)$$

$$C'_0 \neq C_5'^{r}. \quad (5)$$

If (4) and (5) hold, outputs $true$. Otherwise, outputs $flase$.

Note that, Equation (3) ensures that all the elements except C'_0 in a re-encrypted ciphertext CT' are correct. These elements are duplicated from the original ciphertext and re-encryption key and didn't consume computation resources. Thus, usually, the cloud serve has no incentive to revise these elements and return incorrect ones. Equation (4) ensures that m in the proof $\pi = (m, R)$ is indeed the underlying plaintext of the original ciphertext CT, and Equation (5) ensures that m is not the underlying plaintext of the re-encrypted ciphertext CT', which means the part C'_0 is not correctly generated.

Correctness. Next, we explain the correctness of the decryption algorithm.

When CT is an original ciphertext, then Equation (1) is as

$$\begin{aligned} \Phi &= \frac{e(K_1, C_1)}{\prod_{j\in J}(e(K_2, C_{3,j}) \cdot e(K_{\rho(j)}, C_{4,j}))^{\theta_j}} \\ &= \frac{e(g^\alpha g^{as}, g^r)}{\prod_{j\in J}(e(g^s, g^{a\lambda_j} f_{\rho(j)}^{-r_j}) \cdot e(f_{\rho(j)}^s, g^{r_j}))^{\theta_j}} \\ &= \frac{e(g,g)^{\alpha r} \cdot e(g^{as}, g^r)}{e(g^s, g^a)^{\sum_{j\in J}\lambda_j\theta_j}} \\ &= e(g,g)^{\alpha r}. \end{aligned}$$

Then

$$\begin{aligned} & C \oplus H_1(\Phi) \\ &= [(m||R) \oplus H_1(e(g,g)^{\alpha r})] \oplus H_1(e(g,g)^{\alpha r}) \\ &= m||R. \end{aligned}$$

When CT' is a re-encrypted ciphertext, then

$$\begin{aligned} C'_0 =& \frac{e(rk_1, C_1) \cdot e(rk_2, C_2)^{-1}}{\prod_{j\in J}(e(rk_3, C_{3,j}) \cdot e(rk_{4,\rho(j)}, C_{4,j}))^{\theta_j}} \\ =& \frac{e(sk_1^{H_2(\chi)} \cdot Q^\delta, g^r) \cdot e(g^\delta, Q^r)^{-1}}{\prod_{j\in J}(e(sk_2^{H_2(\chi)}, g^{a\lambda_j} f_{\rho(j)}^{-r_j}) \cdot e(sk_{\rho(j)}^{H_2(\chi)}, g^{r_j}))^{\theta_j}} \\ =& e(g,g)^{\alpha r H_2(\chi)}. \end{aligned}$$

χ in Equation (2) can be verified in the same way as in Equation (1). Thus

$$\begin{aligned} & C' \oplus H_1(C_0'^{\frac{1}{H_2(\chi)}}) \\ &= [(m||R) \oplus H_1(e(g,g)^{\alpha r})] \oplus H_1(e(g,g)^{\alpha r H_2(\chi)/H_2(\chi)}) \\ &= m||R. \end{aligned}$$

4.3 Security Proof

Here will prove the semantic security, verifiability and fairness of our proposed VF-CP-ABPRE construction.

Theorem 1. *Our proposed VF-CP-ABPRE scheme is semantic secure under the q-parallel BDHE assumption.*

Proof. According to semantic security that is defined in Definition 1, we prove its Semantic-Or and Semantic-Re security through the following two lemmas. □

Lemma 1: VF-CP-ABPRE scheme is Semantic-Or secure under the q-parallel BDHE assumption.

Proof. Assume a PPT adversary $\mathcal{A}$ can break the Semantic-Or security of the proposed scheme with a non-negligible probability ϵ, we can construct a simulator $\mathcal{B}$ that can solve the q-parallel BDHE assumption with an advantage ϵ. $\mathcal{B}$ is given a q-parallel BDHE instance $(\vec{v}, T)$ as illustrate in Section 3.4, its goal is to decide whether T equals to $e(g,g)^{r\cdot\alpha^{q+1}}$ or T is randomly chosen from G_T. □

Initially, $\mathcal{B}$ maintains private key and re-encryption key lists which are initially empty.

- $List_{sk}$: stores a tuple of (S, sk_S).
- $List_{rk}$ stores a tuple of $(S, (\widetilde{\mathbb{A}}, \widetilde{\rho}), rk, index)$, where $index \in \{0, 1\}$. $index = 1$ indicates that rk is a correct

re-encryption key, while $index = 0$ indicates rk is a random one.

- Init. $\mathcal{A}$ outputs a challenge access policy $(\mathbb{A}^*, \rho^*)$, where $\mathbb{A}^*$ is a $l^* \times n^*$ matrix.
- Setup. $\mathcal{B}$ chooses a random $\beta, \varpi \in Z_p$, $\phi, \varphi \in G$ and sets $e(g,g)^{\alpha} = e(g,g)^{\beta} \cdot e(g^a, g^{a^q})$, $Q = g^{\varpi}$. This implicitly sets $\alpha = a^{q+1} + \beta$ for some unknown α. For each $x \in U$, chooses a random value $t_x \in Z_p$. Denote $X = \{j : \rho^*(j) = x\}$, $\mathcal{B}$ computes
$$f_x = g^{t_x} \prod_{j \in X} g^{a\mathbb{A}^*_{j,1}/b_j} \cdot g^{a^2\mathbb{A}^*_{j,2}/b_j} \ldots g^{a^{n^*}\mathbb{A}^*_{j,n}/b_j}.$$
If X is an empty set, then $f_x = g^{t_x}$. $\mathcal{B}$ also chooses two cryptographic hash functions H_1, H_2, and a message-lock encryption algorithm MLE. Finally, $\mathcal{B}$ outputs the public parameter $PP = (e, G, G_T, g, f_1, \ldots, f_U, \phi, \varphi, Q, g^a, e(g,g)^{\alpha}, H_1, H_2, MLE)$.
- Query phase I. $\mathcal{A}$ queries:
  1) $\mathcal{O}_{sk}(S)$: $\mathcal{A}$ queries on attribute set S. $\mathcal{B}$ verifies that $R(S, (\mathbb{A}^*, \rho^*)) = 0$. If not, outputs $\perp$. Otherwise, $\mathcal{B}$ first finds a vector $\vec{\theta} = (\theta_1, \ldots, \theta_{n^*})$ such that $\theta_1 = -1$ and for all j where $\rho^*(j) \in S$, then $\vec{\theta} \cdot \mathbb{A}^*_j = 0$.[2] $\mathcal{B}$ randomly chooses $s' \in Z_p$, and computes
$$K_2 = g^{s'} \prod_{j=1}^{n^*} (g^{a^{q+1-j}})^{\theta_j} \triangleq g^s.$$
This implicitly sets $s = s' + \sum_{j=1}^{n^*} \theta_j \cdot a^{q+1-j}$.
Then, we have
$$K_1 = g^{\alpha} g^{as} = g^{a^{q+1}+\beta} \cdot g^{as' + \sum_{j=1}^{n^*} \theta_j \cdot a^{q+2-j}}$$
$$= g^{\beta} (g^a)^{s'} \prod_{j=2}^{n^*} (g^{a^{q+2-j}})^{\theta_j},$$
which can be easily computed.
For those $x \in S$ and no j such that $\rho^*(j) = x$ exists, $\mathcal{B}$ sets $K_x = K_2^{t_x}$.
For those $x \in S$ and $X = \{j : \rho^*(j) = x\} \neq \emptyset$, as $\vec{\theta} \cdot \mathbb{A}^*_j = 0$, then
$$K_x = f_x^s$$
$$= \left(g^{t_x} \prod_{j \in X} \prod_{i=1}^{n^*} g^{a^i \mathbb{A}^*_{j,i}/b_j} \right)^{s' + \sum_{k=1}^{n^*} \theta_k \cdot a^{q+1-k}}$$
$$= K_2^{t_x} \prod_{j \in X} \prod_{i=1}^{n^*} \left(g^{(a^i/b_j)/s'} \cdot \prod_{\substack{k=1 \\ k \neq i}}^{n^*} (g^{a^{q+1+i-k}/b_j})^{\theta_k} \right)^{\mathbb{A}^*_{j,i}},$$
which can also be computed by the given parameters.
Thus, $\mathcal{B}$ simulates the private key query and adds (S, sk_S) to $List_{sk}$.
  2) $\mathcal{O}_{rk}(S, (\widetilde{\mathbb{A}}, \widetilde{\rho}))$: $\mathcal{B}$ first checks that
     - If $R(S, (\mathbb{A}^*, \rho^*) = 1)$ and there exists an entry $(\widetilde{S}, sk_{\widetilde{S}})$, where $R(\widetilde{S}, (\widetilde{\mathbb{A}}, \widetilde{\rho}) = 1$, in $List_{sk}$, outputs $\perp$.
     - Else if $R(S, (\mathbb{A}^*, \rho^*) = 1)$ and no such an entry $(\widetilde{S}, sk_{\widetilde{S}})$, where $R(\widetilde{S}, (\widetilde{\mathbb{A}}, \widetilde{\rho}) = 1$, exists in $List_{sk}$, $\mathcal{B}$ selects a random rk and adds $(S, (\widetilde{\mathbb{A}}, \widetilde{\rho}), rk, 0)$ to $List_{rk}$. Otherwise,
     - $\mathcal{B}$ first makes a query $\mathcal{O}_{sk}(S)$ to get sk_S, and then computes rk with sk_S as in the $ReKey Gen$ algorithm. Finally $\mathcal{B}$ adds $(S, (\widetilde{\mathbb{A}}, \widetilde{\rho}), rk, 1)$ to $List_{rk}$.
  3) $\mathcal{O}_{re}(CT, S, (\widetilde{\mathbb{A}}, \widetilde{\rho}))$: If $R(S, (\mathbb{A}^*, \rho^*)) = 1$ and exists an entry $(\widetilde{S}, sk_{\widetilde{S}})$, where $R(\widetilde{S}, (\widetilde{\mathbb{A}}, \widetilde{\rho}) = 1$, in $List_{sk}$, outputs $\perp$. Otherwise, if there exists an entry $(S, (\widetilde{\mathbb{A}}, \widetilde{\rho}), rk, sign)$ in $List_{rk}$, $\mathcal{B}$ re-encrypts CT with rk. Otherwise, $\mathcal{B}$ first makes a query $\mathcal{O}_{rk}(S, (\widetilde{\mathbb{A}}, \widetilde{\rho}))$ to get rk and then re-encrypts $CT>$ with rk.
- Challenge. Two plaintext (m_0, m_1) with equal length are selected by $\mathcal{A}$ and then $\mathcal{A}$ sends them to $\mathcal{B}$. $\mathcal{B}$ randomly chooses $\sigma \in \{0, 1\}$ and computes
$$C = m_{\sigma} \cdot T \cdot e(g^r, g^{\beta}), C_1 = g^r, C_2 = (g^r)^{\varpi} = Q^r.$$
$\mathcal{B}$ randomly choose $y'_2, \ldots, y'_{n^*} \in Z_p^{n^*-1}$ and $r_1, \ldots, r'_l \in Z_p^l$. For $j \in [1, l^*]$, denote W_j as all $k \neq j$ and $\rho^*(k) = \rho^*(j)$. $\mathcal{B}$ implicitly set $\vec{\mu} = (r, ra + y'_2, ra^2 + y'_3, \ldots, ra^{n-1} + y'_{n^*})$ and $\lambda_j = \vec{\mu} \cdot \mathbb{A}^*_j$, $j \in [1, l^*]$.
Then, $\mathcal{B}$ computes
$$C_{4,j} = g^{-rb_j} g^{-r'_j} \triangleq g^{r_j},$$
$$C_{3,j} = g^{a\lambda_j} f_{\rho^*(j)}^{-r_j}$$
$$= f_{\rho^*(j)}^{r'_j} \cdot (g^{r \cdot b_j})^{t_{\rho^*(j)}}$$
$$\cdot \left(\prod_{k \in W_j} \prod_{i=1}^{n^*} (g^{a^i r b_j / b_k})^{\mathbb{A}^*_{k,i}} \right) \cdot \left(\prod_{i=2}^{n^*} (g^a)^{y'_j \mathbb{A}^*_{j,i}} \right).$$
- Query phase II. $\mathcal{A}$ can query as in Query phase I under the restrictions defined in the Semantic-Or security model.
- Guess. $\mathcal{A}$ outputs a guess σ'. $\mathcal{B}$ outputs 1 that indicates $T = e(g,g)^{r \cdot \alpha^{q+1}}$ if $\sigma' = \sigma$. Otherwise, $\mathcal{B}$ outputs 0 indicates that T is a random value in G_T.

Analysis. When $T = e(g,g)^{r \cdot a^{q+1}}$, then $C = m_{\sigma} \cdot T \cdot e(g^r, g^{\beta}) = m \cdot e(g,g)^{r \cdot (a^{q+1} + \beta)} = m \cdot e(g,g)^{\alpha r}$, which is a perfect ciphertext. Then, $Pr[\mathcal{B}(\vec{v}, e(g,g)^{r \cdot a^{q+1}}) = 1] = 1/2 + \epsilon$. When T is a random value in G_T, then $Pr[\mathcal{B}(\vec{v}, T) = 1] = 1/2$. Thus, $|Pr[\mathcal{B}(\vec{v}, e(g,g)^{r \cdot a^{q+1}}) = 1] - Pr[\mathcal{B}(\vec{v}, T) = 1]| = |1/2 + \epsilon - 1/2| = \epsilon$, which means that $\mathcal{B}$ can solve the q-parallel BDHE assumption with a non-negligible advantage ϵ.

Lemma 2: VF-CP-ABPRE scheme is Semantic-Re secure under the q-parallel BDHE assumption.

Proof. Suppose a PPT adversary $\mathcal{A}$ can break the Semantic-Or security of the proposed scheme with a non-negligible probability ϵ, we are able to construct a simulator $\mathcal{B}$ that can solve the q-parallel BDHE assumption with an advantage ϵ. $\mathcal{B}$ is given a q-parallel BDHE instance $(\vec{v}, T)$ as

2. Note that, as illustrated in [6], if $R(S, (\mathbb{A}^*, \rho^*)) = 0$, such a vector $\vec{\theta}$ must exists and can be found in polynomial time.

illustrate in Section 3.4, its goal is to decide whether T equals to $e(g,g)^{r\cdot\alpha^{q+1}}$ or T is randomly chosen from G_T. □

The Initial, Setup and Query phase I are identical to that in the Lemma 3 proof.

- Challenge. Two plaintext (m_0, m_1) with equal length are selected by $\mathcal{A}$ and then $\mathcal{A}$ sends them to $\mathcal{B}$. $\mathcal{B}$ first choose an attribute set S where $R(S, (\mathbb{A}^*, \rho^*)) = 0$, and generates a private key sk_S and re-encryption key $rk = ReKeyGen(sk_S, (\mathbb{A}^*, \rho^*))$. Then $\mathcal{B}$ chooses $(\mathbb{A}, \rho)$ where $R(S, (\mathbb{A}, \rho)) = 1$ and $CT = Enc(m_\sigma, (\mathbb{A}, \rho))$ as in the Challenge phase in the proof of Lemma 1. Finally, $\mathcal{B}$ computes $CT'^* = ReEnc(CT, rk)$, and returns CT'^* to $\mathcal{A}$.
- Query phase II. $\mathcal{A}$ can query as in Query phase I under the restrictions defined in the Semantic-Re security model.
- Guess. $\mathcal{A}$ outputs a guess σ'. $\mathcal{B}$ outputs 1 that indicates $T = e(g,g)^{r\cdot\alpha^{q+1}}$ if $\sigma' = \sigma$. Otherwise, $\mathcal{B}$ outputs 0 indicates that T is a random value in G_T.

Analysis. When $T = e(g,g)^{r\cdot\alpha^{q+1}}$, CT'^* is a perfect ciphertext. Then, $Pr[\mathcal{B}(\vec{v}, e(g,g)^{r\cdot\alpha^{q+1}}) = 1] = 1/2 + \epsilon$. When T is a random value in G_T, then $Pr[\mathcal{B}(\vec{v}, T) = 1] = 1/2$. Thus, we have $|Pr[\mathcal{B}(\vec{v}, e(g,g)^{r\cdot\alpha^{q+1}}) = 1] - Pr[\mathcal{B}(\vec{v}, T) = 1]| = |1/2 + \epsilon - 1/2| = \epsilon$, which means that $\mathcal{B}$ can solve the q-parallel BDHE assumption with a non-negligible advantage ϵ.

Thus, the proof of Theorem 1 is completed.

Theorem 2. *VF-CP-ABPRE scheme is verifiable under the discrete logarithm assumption.*

Proof. Suppose there is an adversary $\mathcal{A}$ that can break the verifiability of the presented VF-CP-ABPRE scheme with an negligible advantage ϵ, we can construct a simulator $\mathcal{B}$, that can solve the discrete logarithm problem with an advantage ϵ. $\mathcal{B}$ is given a discrete logarithm instance $(e, G, G_T, p, g, g^\beta)$ as in Section 3.5, its goal is to output the value β.

- Setup. Simulator $\mathcal{B}$ sets a bilinear pairing e, G, G_T, g, p and chooses $\alpha, \delta, \varpi \in Z_p$, $f_1, \ldots, f_U \in G$ and a cryptographic hash function $H: \{0,1\}^* \to Z_p$. $\mathcal{B}$ sets $\phi = g^\beta$, $\varphi = g^\delta$, $Q = g^\varpi$. $\mathcal{B}$ also chooses two cryptographic hash functions H_1, H_2, and a message-lock encryption algorithm MLE. Finally, $\mathcal{B}$ outputs the public parameter $PP = (e, G, G_T, g, f_1, \ldots, f_U, \phi, \varphi, Q, g^a, e(g,g)^\alpha, H_1, H_2, MLE)$.
- Query phase I. When $\mathcal{A}$ issues $\mathcal{O}_{sk}(S)$, $\mathcal{O}_{rk}(S, (\widetilde{\mathbb{A}}, \widetilde{\rho}))$ and $\mathcal{O}_{claim}(CT, rk, CT', \pi)$, since $\mathcal{B}$ knows the master secret key α, $\mathcal{B}$ can generate the private key and re-encryption key via the $KeyGen$, $ReKeyGen$ and $Claim$ algorithms, and then returns the results to $\mathcal{A}$.
- Challenge. $\mathcal{A}$ outputs a message m^* and access policy $(\mathbb{A}^*, \rho^*)$ and sends them to $\mathcal{B}$. $\mathcal{B}$ computes the $CT^* = Enc(m^*, (\mathbb{A}^*, \rho^*))$, and sends CT^* to $\mathcal{A}$.
- Query phase II. The simulator $\mathcal{B}$ answers the queries as in query phase I.
- Output. $\mathcal{A}$ outputs a re-encrypted ciphertext $CT'^* = (C'^*, \overline{C}'^*, C_0'^*, C_1'^*, C_2'^*, \{C_{3,j}'^*, C_{4,j}'^*\}_{j\in[1,\widetilde{l}]}, C_5'^*, (\widetilde{\mathbb{A}}^*, \widetilde{\rho}^*))$. □

Simulator $\mathcal{B}$ chooses an attribute set $\widetilde{S}^*$ where $R(\widetilde{S}^*, (\widetilde{\mathbb{A}}^*, \widetilde{\rho}^*)) = 1$), and generates a secret key $sk_{\widetilde{S}^*}$ using the master secret key. Then, $\mathcal{B}$ decrypts CT'^* to get the plaintext m'^* and R'^*. If $\mathcal{A}$ wins the verifiability game, which means $m'^* \notin \{m^*, \perp\}$, then we have $\overline{C}'^* = C'^*$. $\mathcal{B}$ can ensure that

$$\begin{aligned}
&\overline{C}'^* = C'^* \\
&\Leftrightarrow \phi^{H_2(m'^*)}\varphi^{H_2(R'^*)} = \phi^{H_2(m^*)}\varphi^{H_2(R^*)} \\
&\Leftrightarrow g^{\beta\cdot H_2(m'^*)+\delta\cdot H_2(R'^*)} = g^{\beta\cdot H_2(m^*)+\delta\cdot H_2(R^*)} \\
&\Leftrightarrow \beta\cdot(H_2(m'^*) - H_2(m^*)) = \delta\cdot(H_2(R^*) - H_2(R'^*)).
\end{aligned}$$

Finally, $\mathcal{B}$ can compute $\beta = \frac{\delta\cdot(H_2(R^*)-H_2(R'^*))}{H_2(m'^*)-H_2(m^*)}$, and returns β as its answer.

Analysis. The above simulation is perfect, the advantage of $\mathcal{B}$ to solve the discrete logarithm assumption is identical to the advantage of an adversary $\mathcal{A}$ to win in the integrity security game.

Theorem 3. *VF-CP-ABPRE scheme is fair if H_1, H_2 are cryptographic hash functions, MLE is a message-lock encryption scheme and the discrete logarithm assumption holds.*

Proof. Assuming an adversary $\mathcal{A}$ can break the fairness game with a non-negligible advantage ϵ, then we can construct a simulator to break the collision resistant property of hash function H_2 with a probability ϵ', such that $\epsilon' \geqslant \epsilon - Adv_{MLE} - Adv_{DL}$, where Adv_{MLE} and Adv_{DL} denote the advantage of an adversary breaking the MLE encryption and discrete logarithm assumption respectively.

- Setup. The process is as the same as the Setup in the verifiability game.
- Query phase I. The process is as the same as in the verifiability game.
- Challenge. $\mathcal{A}$ outputs a message m^* and access policy $(\mathbb{A}^*, \rho^*)$. Simulator $\mathcal{B}$ computes $CT^* = Enc(m^*, (\mathbb{A}^*, \rho^*))$, $CT'^* = ReEnc(CT^*, rk^*)$, where rk^* is a re-encryption key from $(\mathbb{A}^*, \rho^*)$ to a new access policy $(\widetilde{\mathbb{A}}, \widetilde{\rho})$, and sends CT^*, CT'^* to $\mathcal{A}$.
- Query phase II. The same as in the verifiability game.
- Guess. $\mathcal{A}$ outputs a proof $\pi = (m, R)$.

Analysis. If $\mathcal{A}$ wins the game, it means $Claim(CT^*, CT'^*, \pi) = true$. Then we have $r = H_2(MLE(m||R))$, $\overline{C}'^* = \overline{C}^* = \phi^{H_2(m)}\varphi^{H_2(R)}$ and $C_0'^* \neq C_5'^{*r}$. Since $C_0'^* \neq C_5'^{*r}$, where $C_0'^* = e(g,g)^{\alpha H_2(\chi)H_2(MLE(m^*||R^*))}$, $C_5'^* = e(g,g)^{\alpha H_2(\chi)}$, we have $(m||R) \neq (m^*||R^*)$. Furthermore, as $\overline{C}'^* = \overline{C}^* = \phi^{H_2(m)}\varphi^{H_2(R)}$, we have $\phi^{H_2(m^*)}\varphi^{H_2(R^*)} = \phi^{H_2(m)}\varphi^{H_2(R)}$. Denote $\varphi = \phi^\eta$ for some unknown random value η. Then we have $H_2(m^*) = H_2(m)$ or $H_2(R^* = H_2(R)$. Otherwise, anyone can use these values to compute $\eta = (H_2(m) - H_2(m^*))/(H_2(R^*) - H_2(R))$ to solve the discrete logarithm assumption. Thus, the simulator $\mathcal{B}$ find a collision tuple of the cryptographic hash function H_2, where $m \neq m^*$,

TABLE 1
Security and Functionality Comparison With [26], [28], and [32]

Schemes	Access structure	Security	Verifiability	Fairness
[24]	Ciphertext policy	Semantic security	×	×
[26]	Key policy	CCA	×	×
[30]	Ciphertext policy	CCA	×	×
Ours	Ciphertext policy	Semantic security	✓	✓

$H_2(m^*) = H_2(m)$, or $R \neq R^*$, $H_2(R^*) = H_2(R)$. $\mathcal{B}$'s advantage to find such a collision tuple is $\epsilon - Adv_{MLE} - Adv_{DL}$. □

Discussion. The collusion resistant property which ensures that even the cloud server collude with a shared user cannot reveal the original recipient's private key is a basic security requirement for ABPRE schemes. In our proposed construction, the $rk_1 = K_1^{H_2(\chi)} \cdot Q^{\delta}$ part of the original recipient's private key is randomized by a random value Q^{δ}. When the cloud server collude with a shared user, they can get the value of χ. However, they cannot recover the value of Q^{δ} from elements (g, g^{δ}, Q) which essentially is a Diffie-Hellman assumption. Thus, the original recipient' private key part rk_1 cannot be revealed. In this manner, the proposed construction achieves the collusion resistant property.

5 Performance and Evaluation

5.1 Security and Functionality Comparison

We made a comparison of the proposed scheme and schemes [26], [28], [32] in terms of access structure, security, verifiability, and fairness. Table 1 summarizes that [28] works in the key policy setting and [26], [32] and our construction work in the ciphertext policy setting. Moreover, [28] and [32] achieve the CCA security, and [26] and our construction achieve the semantic security. However, only our construction achieves the verifiability, and fairness properties.

5.2 Performance Evaluation

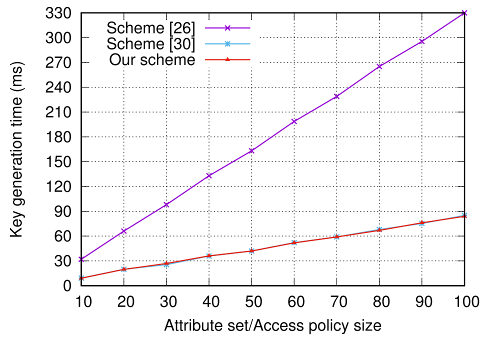
We conducted an implementation to evaluate the presented VF-CP-ABPRE scheme in the perspective of computation cost, and compared our proposed scheme with the key-policy attribute-based data sharing scheme [28] and the ciphertext-policy [32] attribute-based data sharing scheme.

In our implementation, we use a Java cryptography package [34] for pairing computation which is implemented based on a C library of Pairing-based cryptography [35]. We use the $Y^2 = X^3 + X$ elliptic curve (type A) and the group order is set as 160 bit. The hash function *SHA*-256 is adopted as the hash function H_1 and H_2 in our scheme. The hardware specification we used in this experiment is a laptop with CPU: Intel i5-8520U @1.60GHZ 1.80GHZ, RAM: 8G. And the laptop runs on Linux Mint 18.1.

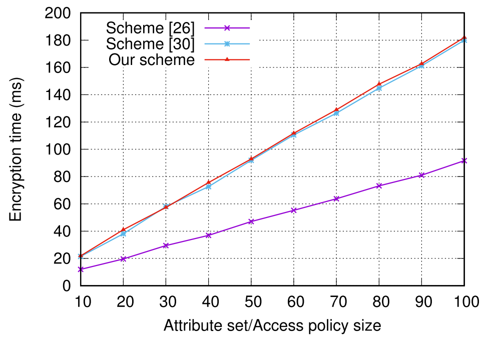
Implementation. The size of universal attribute set is set be 1000 in our experiments. The attribute set is set in the private key generation and the access policy during the generation of the original ciphertext are sized varied from 10 to 100 with a step 10. During the re-encryption key generation, the shared access policy $(\widetilde{\mathbb{A}}, \widetilde{\rho})$ is also sized varied from 10 to 100 with a step 10. All the access policies are set as the AND gate of the selected attributes. Each experiment was execute 100 times to get an accurate average execution time.

Fig. 2 shows the execution time comparison of each algorithm in our scheme with [28], [32]. Figs. 2a and 2b show that the our scheme's key generation time as well as the one in [32] performed better than the scheme in [28]. While the encryption time in our scheme and [32] is more than that in [28]. This is because the access policy is generated in the encryption phase in the ciphertext-policy setting. However the scheme [28] works in the key-policy setting and the access policy is generated at the key generation phase. Moreover, the execution time of the $KeyGen$ and Enc linearly increases as the size access policy grows. Figs. 2c and 2d show that the re-encryption key generation time and the actual encryption time in our scheme and in scheme [32] are very close and our scheme performs slightly better than scheme [28]. As shown in Figs. 2e and 2f, the original and re-encrypted ciphertext decryption time are almost the same in the three schemes. In order to present the efficiency of the verifiability and fairness in our scheme, since only our scheme achieves the verifiability and fairness property, we compare the $Claim$ verification time in our scheme with the time of repeating the re-encryption operation[3] in [28] and [32]. The time comparison, shown in Fig. 2g, demonstrates that the $Claim$ verification time in our scheme is much less than other schemes [28], [32] and doesn't increase even if the access policy size grows. While the execution time of repeating the re-encryption operation in [28], [32] are much higher than that in our scheme and increase linearly with the access policy. This is because that in our scheme, only three exponent computation in group G and three hash function computation are needed in the $Claim$ algorithm. The computation cost of the $Claim$ algorithm is constant and doesn't increase with the size of the access policy. However, in order to achieve verifiability and fairness for schemes [28] and [32], a verifier needs to repeat the $ReEnc$ algorithm to verify whether the output of the cloud server is correct. However, the $ReEnc$ algorithm is linear with the access policy.

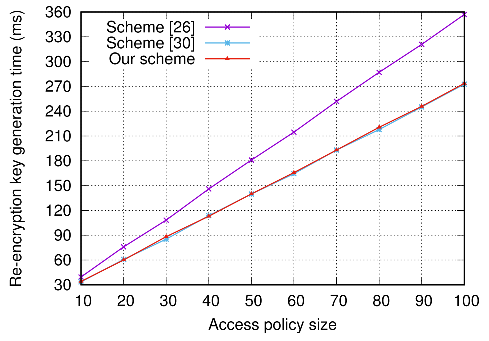
3. Since only our scheme achieves the verifiability and fairness property, in the previous schemes [28], [32] a verifier can check whether the returned re-encrypted ciphertext is correct only by repeating the re-encryption operation.



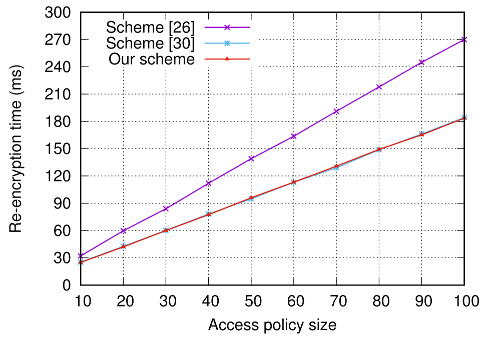
(a) $KeyGen$ algorithm computation time



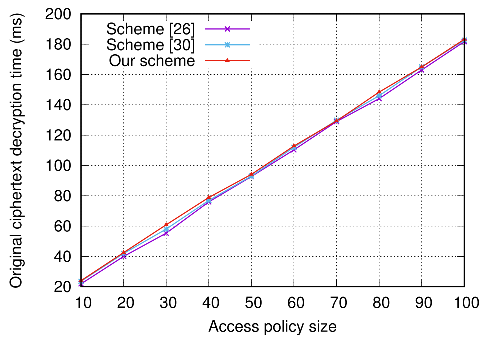
(b) Enc algorithm computation time



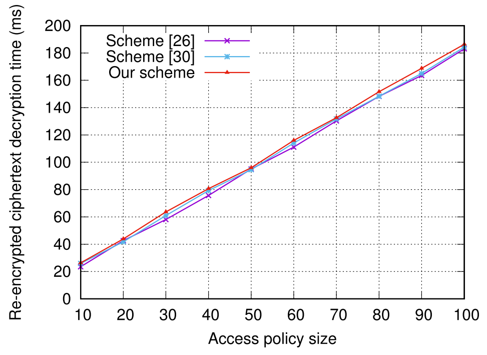
(c) $ReKeyGen$ algorithm computation time



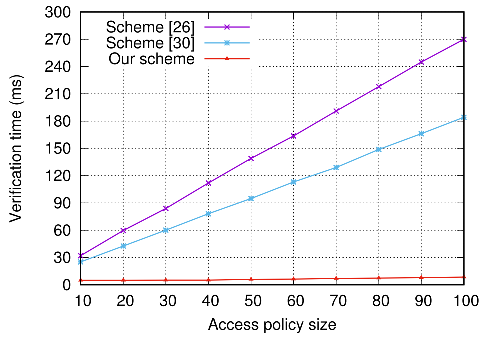
(d) $ReEnc$ algorithm computation time



(e) Original ciphertext decryption time



(f) Re-encrypted ciphertext decryption time



(g) $Claim$/Repeating re-encryption computation time

Fig. 2. Computation time of our proposed VF-CP-ABPRE scheme.

6 Conclusion

This paper introduces the verifiability and fairness security requirements for attribute-based data sharing in clouds and put forward a notion of verifiable and fair ciphertext-policy attribute-based proxy re-encryption(VF-CP-ABPRE). The scheme provides the ability for a shared user to verify the validity of the re-encrypted ciphertext. Moreover, a shared user cannot make a malicious accusation to the cloud provider if the cloud has indeed returned a correct re-encrypted ciphertext. We also have proven our VF-CP-ABPRE scheme's semantic security, verifiability and fairness in our security model. We also conducted an implementation to evaluate our proposed scheme, and compared the execution time between our scheme and previous schemes to demonstrate the efficiency of the proposed VF-CP-ABPRE scheme.

This work opens up many interesting research questions. One could be how to design a secure VF-CP-ABPRE scheme that is adaptively secure. Another interesting problem is designing a publicly verifiable CP-ABPRE scheme that supports the cloud server to generate a proof and everyone can verify the re-encrypted ciphertext efficiently without repeating the re-encryption operation.

Acknowledgments

This work was supported in part by the Australian Research Council Discovery Project DP180100665.

References

[1] X. Chen, X. Huang, J. Li, J. Ma, W. Lou, and D. S. Wong, "New algorithms for secure outsourcing of large-scale systems of linear equations," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 1, pp. 69–78, Jan. 2015.

[2] K. Ren, C. Wang, and Q. Wang, "Security challenges for the public cloud," *IEEE Internet Comput.*, vol. 16, no. 1, pp. 69–73, Jan./Feb. 2012.

[3] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Proc. Int. Conf. Theory Appl. Cryptogr. Techn.*, 2005, pp. 457–473.

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

[5] H. Ma, R. Zhang, Z. Wan, Y. Lu, and S. Lin, "Verifiable and exculpable outsourced attribute-based encryption for access control in cloud computing," *IEEE Trans. Dependable Secure Comput.*, vol. 14, no. 6, pp. 679–692, Nov./Dec. 2017.

[6] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proc. IEEE Symp. Secur. Privacy*, 2007, pp. 321–334.

[7] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proc. ACM Conf. Comput. Commun. Secur.*, 2006, pp. 89–98.

[8] K. Emura, A. Miyaji, A. Nomura, K. Omote, and M. Soshi, "A ciphertext-policy attribute-based encryption scheme with constant ciphertext length," in *Proc. Int. Conf. Inf. Secur. Pract. Experience*, 2009, pp. 13–23.

[9] S. Hohenberger and B. Waters, "Attribute-based encryption with fast decryption," in *Proc. Int. Workshop Public Key Cryptogr.*, 2013, pp. 162–179.

[10] N. Attrapadung, B. Libert, and E. De Panafieu , "Expressive key-policy attribute-based encryption with constant-size ciphertexts," in *Proc. Int. Workshop Public Key Cryptogr.*, 2011, pp. 90–108.

[11] J. Herranz, F. Laguillaumie, and C. Ràfols, "Constant size ciphertexts in threshold attribute-based encryption," in *Proc. Int. Workshop Public Key Cryptogr.*, 2010, pp. 19–34.

[12] N. Attrapadung, J. Herranz, F. Laguillaumie, B. Libert, E. De Panafieu , and C. Ràfols, "Attribute-based encryption schemes with constant-size ciphertexts," *Theor. Comput. Sci.*, vol. 422, pp. 15–38, 2012.

[13] C. Chen, Z. Zhang, and D. Feng, "Efficient ciphertext policy attribute-based encryption with constant-size ciphertext and constant computation-cost," in *Proc. Int. Conf. Provable Secur.*, 2011, pp. 84–101.

[14] A. Lewko and B. Waters, "New proof methods for attribute-based encryption: Achieving full security through selective techniques," in *Proc. Annu. Cryptol. Conf.*, 2012, pp. 180–198.

[15] B. Waters, "Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization," in *Proc. Int. Workshop Public Key Cryptogr.*, 2011, pp. 53–70.

[16] J. Chen and H. Wee, "Semi-adaptive attribute-based encryption and improved delegation for boolean formula," in *Proc. Int. Conf. Secur. Cryptogr. Netw.*, 2014, pp. 277–297.

[17] A. Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters, "Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption," in *Proc. Annu. Int. Conf. Theory Appl. Cryptogr. Techn.*, 2010, pp. 62–91.

[18] J. Wei, X. Chen, X. Huang, X. Hu, and W. Susilo, "RS-HABE: Revocable-storage and hierarchical attribute-based access scheme for secure sharing of E-health records in public cloud," *IEEE Trans. Dependable Secure Comput.*, vol. PP, no. 99, p. 1, 2019.

[19] T. Nishide, K. Yoneyama, and K. Ohta, "Attribute-based encryption with partially hidden encryptor-specified access structures," in *Proc. Int. Conf. Appl. Cryptography Netw. Secur.*, 2008, pp. 111–129.

[20] T. V. X. Phuong, G. Yang, and W. Susilo, "Hidden ciphertext policy attribute-based encryption under standard assumptions," *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 1, pp. 35–45, Jan. 2016.

[21] H. Cui, R. H. Deng, G. Wu, and J. Lai, "An efficient and expressive ciphertext-policy attribute-based encryption scheme with partially hidden access structures," in *Proc. Int. Conf. Provable Secur.*, 2016, pp. 19–38.

[22] A. Kapadia, P. P. Tsang, and S. W. Smith, "Attribute-based publishing with hidden credentials and hidden policies," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2007, pp. 179–192.

[23] M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography," in *Proc. Int. Conf. Theory Appl. Cryptogr. Techn.*, 1998, pp. 127–144.

[24] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, "Improved proxy re-encryption schemes with applications to secure distributed storage," *ACM Trans. Inf. Syst. Secur.*, vol. 9, no. 1, pp. 1–30, 2006.

[25] S. Ohata, Y. Kawai, T. Matsuda, G. Hanaoka, and K. Matsuura, "Re-encryption verifiability: How to detect malicious activities of a proxy in proxy re-encryption," in *Proc. Cryptogr. Track RSA Conf.*, 2015, pp. 410–428.

[26] X. Liang, Z. Cao, H. Lin, and J. Shao, "Attribute based proxy re-encryption with delegating capabilities," in *Proc. Int. Symp. Inf. Comput Commun. Secur.*, 2009, pp. 276–286.

[27] K. Liang *et al.*"A secure and efficient ciphertext-policy attribute-based proxy re-encryption for cloud data sharing," *Future Gener. Comput. Syst.*, vol. 52, pp. 95–108, 2015.

[28] K. Liang and W. Susilo, "Searchable attribute-based mechanism with efficient data sharing for secure cloud storage," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 9, pp. 1981–1992, Sep. 2015.

[29] C. Ge, W. Susilo, J. Wang, Z. Huang, L. Fang, and Y. Ren, "A key-policy attribute-based proxy re-encryption without random oracles," *Comput. J.*, vol. 59, no. 7, pp. 970–982, 2016.

[30] K. Liang, M. H. Au, W. Susilo, D. S. Wong, G. Yang, and Y. Yu, "An adaptively CCA-secure ciphertext-policy attribute-based proxy re-encryption for cloud data sharing," in *Proc. Int. Conf. Inf. Secur. Pract. Experience*, 2014, pp. 448–461.

[31] C. Ge, W. Susilo, L. Fang, J. Wang, and Y. Shi, "A CCA-secure key-policy attribute-based proxy re-encryption in the adaptive corruption model for dropbox data sharing system," *Des. Codes Cryptogr.*, vol. 86, no. 11, pp. 2587–2603, 2018.

[32] C. Ge, W. Susilo, Z. Liu, J. Xia, P. Szalachowski, and F. Liming, "Secure keyword search and data sharing mechanism for cloud computing," *IEEE Trans. Dependable Secure Comput.*, vol. PP, no. 99, p. 1, 2020.

[33] M. Bellare, S. Keelveedhi, and T. Ristenpart, "Message-locked encryption and secure deduplication," in *Proc. Annu. Int. Conf. Theory Appl. Cryptogr. Techn.*, 2013, pp. 296–312.

[34] Nik-U, "Pbc package," 2015. [Online]. Available: https://github.com/Nik-U/pbc

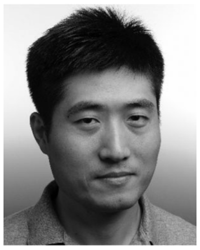
[35] B. Lynn *et al.*, "Pbc library," 2006. [Online]. Available: http://crypto.stanford.edu/pbc

Chunpeng Ge (Member, IEEE) received the PhD degree in computer science from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2016. He is currently a research fellow with the University of Wollongong and an associate professor with the Nanjing University of Aeronautics and Astronautics. His current research interests include cryptography, information security, and privacy preserving for blockchain. His recent work has focused on the topics of public key encryption with keyword search, proxy re-encryption, and identity-based encryption.

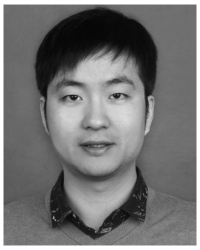
Willy Susilo (Fellow, IEEE) received the PhD degree in computer science from the University of Wollongong, Wollongong, Australia. He is currently a distinguished professor, the head of the School of Computing and Information Technology and the director of the Institute of Cybersecurity and Cryptology, University of Wollongong. He has published more than 400 research papers in the area of cybersecurity and cryptology. His main research interests include cybersecurity, cryptography, and information security. He was a recipient of the Australian Research Council (ARC) future fellow by the ARC and the researcher of the Year Award by the University of Wollongong in 2016. He is the editor-in-chief of the Elsevier's *Computer Standards and Interface* and the MDPI's *Information* journal. He has served as a program committee member in dozens of international conferences. He is currently serving as an associate editor in several international journals, including the *Computer Standards and Interface* (Elsevier) and the *International Journal of Information Security* (Springer). His work has been cited more than 16 000 times in Google Scholar.

Joonsang Baek (Member, IEEE) received the PhD degree from Monash University, Melbourne, Australia, in 2004. He is a senior lecturer with the School of Computer Science and Information Technology and a member of the Institute of Cybersecurity and Cryptology, University of Wollongong, Australia. He was a research scientist with the Institute for Infocomm Research, Singapore, and an assistant professor with the Khalifa University of Science and Technology, United Arab Emirates. His PhD thesis was on security analysis of signcryption, and has received great attention from the research community. He has published his work in numerous reputable journals and conference proceedings. His current research interests include the field of applied cryptography and cybersecurity. He has also served as a Program Committee member and the chair for a number of renowned conferences on information security and cryptography.

Zhe Liu (Senior Member, IEEE) received the BS and MS degrees from Shandong University, Jinan, China, in 2008 and 2011, respectively, and the PhD degree from the University of Luxembourg, Luxembourg, in 2015. He is a professor with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics,China. His research interests include security, privacy and cryptography solutions for the Internet of Things. He has co-authored more than 80 research peer-reviewed journal and conference papers. He was a recipient of the prestigious FNR Awards-Outstanding PhD Thesis Award in 2016, ACM CHINA SIGSAC Rising Star Award in 2017 as well as DAMO Academy Young Fellow in 2019. He serves as program committee member in more than 60 international conferences, including program chairs in INSCRYPT 2019, CRYPTOIC 2019, and ACM CHINA SIGSAC 2020.



Jinyue Xia received the PhD degree in computer science from the University of North Carolina at Charlotte, Charlotte, North Carolina, in 2017. His current research interests include data security, cryptography, and information security. His recent work has focused on the topics of public key encryption with proxy re-encryption and identity-based encryption.



Liming Fang (Member, IEEE) received the PhD degree in computer science from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2012, and has been a postdoctor in the information security from the City University of Hong Kong. He is the associate professor with the School of Computer Science, Nanjing University of Aeronautics and Astronautics. Now, he is a visiting scholar with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology. His current research interests include cryptography and information security. His recent work has focused on the topics of public key encryption with keyword search, proxy re-encryption, and identity-based encryption.