# Outsourcing decryption algorithm of Verifiable transformed ciphertext for data sharing

**Guangwei Xu, Chen Wang, Shan Li, Xiujin Shi, Xin Luo, and Yanglan Gan**
School of Computer Science and Technology, Donghua University
Shanghai,201620, China
[e-mail: gwxu@dhu.edu.cn]
*Corresponding author: Xin Luo

*Received January 26, 2023; revised November 10, 2023; revised March 16, 2024; accepted April 2, 2024; published April 30, 2024*

## Abstract

Mobile cloud computing is a very attractive service paradigm that outsources users' data computing and storage from mobile devices to cloud data centers. To protect data privacy, users often encrypt their data to ensure data sharing securely before data outsourcing. However, the bilinear and power operations involved in the encryption and decryption computation make it impossible for mobile devices with weak computational power and network transmission capability to correctly obtain decryption results. To this end, this paper proposes an outsourcing decryption algorithm of verifiable transformed ciphertext. First, the algorithm uses the key blinding technique to divide the user's private key into two parts, i.e., the authorization key and the decryption secret key. Then, the cloud data center performs the outsourcing decryption operation of the encrypted data to achieve partial decryption of the encrypted data after obtaining the authorization key and the user's outsourced decryption request. The verifiable random function is used to prevent the semi-trusted cloud data center from not performing the outsourcing decryption operation as required so that the verifiability of the outsourcing decryption is satisfied. Finally, the algorithm uses the authorization period to control the final decryption of the authorized user. Theoretical and experimental analyses show that the proposed algorithm reduces the computational overhead of ciphertext decryption while ensuring the verifiability of outsourcing decryption.

***Keywords:*** Mobile Cloud Computing; Cloud Data Sharing; Outsourcing Decryption; Encrypted Data; Verifiable Transformed Ciphertext

The work was sponsored by the Natural Science Foundation of Shanghai (Nos.19ZR1402000 and 21ZR1400400), Education and Scientific Research Project of Shanghai (C160076), the National Natural Science Foundation of China (Nos.61772018 and 62172088), and the National Key Research and Development Program of China (No. 2020YFB1707700).

**http://doi.org/10.3837/tiis.2024.04.010** **ISSN : 1976-7277**

## 1. Introduction

Cloud computing has gained widespread support and adoption as an emerging service model that provides users with powerful computing and storage, among other capabilities. It brings many benefits for geographically dispersed users to share data and significantly reduces the cost of local storage management and maintenance. For mobile devices with limited computing resources (e.g., cell phones), enterprises or individuals can outsource intensively and complexly computing operations to a cloud service provider, thereby increasing local computing efficiency. However, although cloud service providers (CSP) bring convenience to users, they also bring a series of security issues and challenges [1-3].

When an enterprise or individual stores its data (especially sensitive data) directly in the cloud in plaintext, it loses direct control over his data. To ensure the privacy and security of outsourcing sensitive data, the most common method to ensure data confidentiality is currently to encrypt the outsourced data and then upload the ciphertext to the cloud service provider so that the cloud service provider is unable to access and obtain any valuable information from the outsourced data [4]. However, the sharing of encrypted data also poses some difficulties among users. When an enterprise or an individual wants to share the encrypted data or files with other users, the users cannot decrypt these encrypted data directly after downloading the ciphertext. Shamir et al. [5] first proposed identity-based encryption (IBE) scheme in 1984 which can securely implement the function of access control, where IBE is a cryptosystem which allows a user to generate its public key using some unique information about the identity of the user (e.g. a user's name). In this case, a sender can access to the public parameters of the system to encrypt a message using the text-value of the receiver's name as a key, and the receiver can obtain its decryption key from a central authority. Therefore, users can use this scheme to encrypt data, which can not only ensure the security of data but also realize the data sharing with other users.

However, most of the identity-based encryption schemes are constructed based on elliptic curve bilinear groups, which involve more bilinear pairing and power operations, resulting in less efficient decryption algorithms. When users execute the decryption algorithm, the computation of decryption grows linearly with the number of users. To this end, this paper proposes an outsourcing decryption scheme of verifiable transformed ciphertext (VTC-OD). The main contributions of this paper are listed below.

1) Considering that users of mobile devices with insufficient computing power need to consume huge computing overhead in performing the decryption, and that cloud service providers have strong computing power, users can perform partial decryption operations at the cloud service provider side with the powerful computing power and obtain the partial decrypted ciphertext, who only need to do a simple decryption operation to get the plaintext.

2) Since cloud service providers are not fully trustworthy, they may not perform the outsourced decryption request required by the users due to "laziness" to save computing resources or for some benefit, and try to falsify the outsourced decryption result resulting in the users not getting the correct and valid plaintext. Therefore, it is especially important to verify the correctness of the outsourced decryption operation and decrypted ciphertext.

3) Theoretical and experimental analyses are performed. We implement a system prototype and analyze the experimental evaluation, which shows the efficiency of our algorithm.

The rest of this paper is organized as follows. Section 2 summarizes the related work in this paper. Section 3 describes the cloud data sharing system model and presents the current problems. Section 4 proposes an outsourcing decryption scheme of verifiable transformed ciphertext. Section 5 analyzes the security and rationality of this algorithm. Section 6 illustrates the performance of the proposed algorithm in theoretical analysis and experimental simulation. Finally, Section 7 concludes the work and suggests future directions.

## 2. Related Work

To ensure the security of data and solve the problem that users of mobile devices with insufficient computing power need to consume huge computational overhead and long decryption time to perform decryption, this paper focuses on outsourcing decryption.

Many research scholars have proposed large amounts of outsourcing decryption schemes [6-27]. In 2011, Green et al. [6] introduced the first outsourced decryption in attribute-based encryption systems by outsourcing the complex decryption operations to CSP, and the basic idea of this scheme is to blind the user's private key, and then outsource the large number of power and bilinear pairing operations involved in the decryption process to CSP for performing, thus effectively reducing the computational overhead of decryption. Subsequently, attribute-based encryption (ABE) schemes supporting outsourced encryption were proposed in the literature [14-15], where ABE is a cryptosystem which enables fine grained access control of encrypted data using authorization policies. In this case, the secret key of a user and the ciphertext is dependent upon attributes (e.g. its email address, or the kind of subscription it has), and the decryption of a ciphertext is possible only if the set of attributes of the user key matches the attributes of the ciphertext. Since CSP are usually semi-trustworthy, to verify the correctness of the outsourced decryption results returned by CSP, Lai et al. [7] formally introduced the concept of verifiability and proposed an ABE scheme for verifiable outsourced decryption in 2013. After this, verifiable outsourced decryption algorithms were further investigated in the literature [8-10], where Ma and Zhang et al. [8] proposed verifiable and exempt attribute encryption, Lin and Zhang et al. [9] revisited the relationship between attribute-based encryption and verifiable outsourced decryption, and Zhang and Chen et al. [10] proposed a fog computing with outsourcing capability and access control scheme for attribute updating.

Moreover, the identity-based encryption has also been extended to the cryptosystem of proxy re-encryption (PRE) for allowing third parties (proxies) to alter a ciphertext which has been encrypted for one party and then it may be decrypted by another. Green et al. [28] proposed identity-based proxy re-encryption (IB-PRE), which uses the user's unique identity information as the public key, and is characterized by unidirectionality, non-transmissibility, and non-interactivity. Subsequently, many identity-based proxy re-encryption schemes have been proposed [29-31], which combine proxy re-encryption and identity-based encryption, and are more convenient in practical application scenarios. Ge et al. [29] proposed the revocable identity-based proxy re-encryption (RIB-BPRE) scheme, which dynamically shares the encrypted data by updating re-encryption keys and generating re-encrypted ciphertexts. Xu et al. [30] proposed the conditional identity-based broadcast proxy re-encryption (CIBPRE) scheme, which combines the ideas of conditional control and broadcasting, the re-encryption keys and ciphertexts have constant size. Kim et al. [31] proposed a broadcast proxy re-encryption (BPRE) scheme, which combines broadcast encryption and proxy re-encryption, is used for the redistribution of data uploaded on the cloud to multiple users.

However, these works do not address the authorization period of outsourced decryption, which can result in outsourced decryption ciphertexts that have passed the authorization period remaining valid. Therefore, it needs to develop a secure and verifiable outsourced decryption scheme based on the authorization period.

# 3. System Model and Problem Description

## 3.1 System Model

In the cloud storage, encrypted data sharing system model is shown in **Fig. 1** and consists of four main entities: content provider (CP), users, key generation center (KGC), and cloud service provider (CSP).

1) Cloud service provider: CSP is to provide CP with a storage service for storing outsourced data or files and a computing service for transforming stored data. That is, the CSP can transform the data stored by the CP from IBE ciphertext to identity-based broadcast encryption (IBBE) ciphertext and then send it to the user for accessing the ciphertext.

2) Content provider: to protect data privacy, the CP can process the data using the IBE encryption mechanism and then outsource the encrypted data to the CSP. If the CP wishes to share the data with the users, an authorization token is generated and sent to the CSP.

3) Users: users access the data through CSP, get the encrypted data, and decrypt the ciphertext with their private keys.

4) Key generation center: KGC is a trusted participant responsible for responding to registration requests, and generates system parameters and private keys.

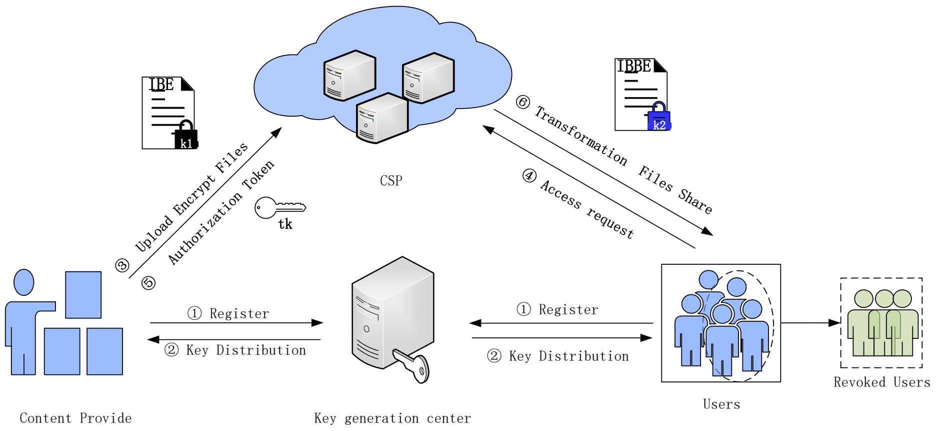


**Fig. 1.** System model

The encrypted data sharing model in the cloud storage consists of six main polynomial time algorithms, i.e., $Setup$, $Keygen$, $Encrypt$, $Authorize$, $Transform$, $Decrypt$, which are composed of the following process.

1) $Setup\ (1^{\lambda}, m) \rightarrow (msk, pp)$. This algorithm is executed by KGC. It inputs the security parameters $1^{\lambda}$ and $m$, outputs system parameters $pp$ and the master key $msk$, where $m$ denotes the maximum number of authorized users. The system parameters are public and the master key is kept secret by KGC.

2) $Register\ (pp, msk, ID) \rightarrow sk_{ID}$. This algorithm is executed by KGC. It inputs the system parameters $pp$, the master key $msk$, user identity $ID \in \{0,1\}^{*}$, and outputs the user's private key $sk_{ID}$ which is sent to the user over a secure channel.

3) $Encrypt\ (pp, M,\ ID) \rightarrow CT_{ID}$. This algorithm is executed by CP. It inputs the system

parameters $pp$, the plaintext $M$, CP's identity $ID$, and outputs a ciphertext $CT_{ID}$ in IBE format which is uploaded to CSP.

4) $Authorize(pp, sk_{ID}, S) \rightarrow Tk_{ID\rightarrow S}$. This algorithm is executed by CP. It inputs the system parameters $pp$, the private key of CP $sk_{ID_{cp}}$, the set of authorized users $S = \{ID_i\}_{\text{i}=1}^{n}$, and outputs authorization token $Tk_{ID\rightarrow S}$ that is sent to CSP over a secure channel.

5) $Transform\ (pp,\ Tk_{ID\rightarrow S}, CT_{ID}) \rightarrow CT_S$. This algorithm is executed by CSP. It inputs the system parameters $pp$, the authorization token $Tk_{ID\rightarrow S}$, and a ciphertext $CT_{ID}$ in IBE format, and outputs a transformed ciphertext $CT_S$ in IBBE format.

6) Decrypt $(pp, sk_{ID'}, CT_{ID}/CT_S) \rightarrow (M/\perp)$. This algorithm is executed by the user. It inputs the system parameters $pp$, the private key $sk_{ID'}$ and a ciphertext $CT_{ID}$ or $CT_S$. For the ciphertext $CT_{ID}$, if $ID' = ID$, CP executes the algorithm and outputs a plaintext message $M$, otherwise outputs error flag ⊥. For ciphertext $CT_S$, if $ID \in S$, users execute the algorithm and output a plaintext message $M$, otherwise output error flag ⊥.

## 3.2 Threat Model

**Definition 1.** Authorization period. The authorization period is the period between the start time and the end time when the users are authorized. Users are granted access to data in the shared system within the limits of the authorization period (e.g., the data service period purchased by the user). During the authorization period assigned by the system, users can access and decrypt the shared data in the system. Once the authorization period expires, users will no longer be able to access and decrypt the shared data, especially the data that was accessed and decrypted during the authorization period.

This paper assumes that CP and KGC are reliable, while users and CSP are semi-trustworthy, i.e., they will follow data sharing protocols "in good faith", but there may be some malicious behavior. Therefore， the following attacks are considered:

1) Unauthorized ciphertext decryption attack. Unauthorized users and semi-trusted CSP may try to decrypt part of the decrypted ciphertext transformed by the CSP without the private key by means of spoofing, eavesdropping, brute force cracking, etc.

2) Forgery attack. To charge users more and save computational resources, a semi-trusted CSP may directly return randomly decrypted packets that look similar to the real ones (e.g., strings of the same length) without performing the packet decryption.

3) Repudiation attack. Even if the CSP returns the correct result, the user may make a false statement to accuse the CSP of returning an incorrect outsourced decrypted ciphertext in order to avoid paying for resource consumption.

## 3.3 Problem Description and Design Objective

As analyzed in the above threat model, the new problems facing outsourced data sharing in this paper are as follows: 1) When decryption is performed by mobile devices, the decryption time is too long due to its limited computing power; 2) A semi-trusted CSP may not perform the outsourced decryption as requested by the user, but directly return a forged decryption result to the user in order to make more profit; 3) The user falsely claims that the CSP has returned the wrong decryption result for its benefit when the CSP returns the correct decryption result.

Based on the above problems, the design objectives of the proposed algorithm are as follows: 1) Shortening the decryption time of ciphertext; 2) Privacy-protected outsourced decryption; 3) Verifiability of outsourcing decryption.

### 3.4 Security Model

The security model guarantees that unauthorized attackers and malicious CSP cannot obtain any valuable information from the ciphertext. The traditional security definition for resisting adaptable selective ciphertext attack (CCA) [32] is to disallow changing any bits of the ciphertext, which is contrary to the ciphertext compression property of outsourced decryption. Therefore, it adopts the replayable chosen-ciphertext attack (RCCA) security, which allows the modification of a given legitimate ciphertext but not the underlying plaintext information.

**Definition 2.** RCCA security. A scheme is RCCA safe if the attacker $\mathcal{A}$ can win the following game in probabilistic polynomial time by a negligible advantage $Adv_{Our,\ \mathcal{A}}^{RCCA}$.

In this paper, a game between the challenger $\mathcal{C}$ and the attacker $\mathcal{A}$ is designed, and the game proceeds as follows.

1) Setup phase. The challenger $\mathcal{C}$ runs the algorithm $Setup\left(1^{\lambda}, m\right)$ to generate the system parameters $pp$ and the master key $msk$, and sends $pp$ to the attacker $\mathcal{A}$.

2) Query phase 1. The challenger $\mathcal{C}$ first initializes an empty table $T$ and an empty data set $D$. The attacker $\mathcal{A}$ can repeat the following queries to the challenger $\mathcal{C}$.

①Private key query $Q_{SK}(ID_i)$: the attacker $\mathcal{A}$ submits the identity $ID_i$ to the challenger $\mathcal{C}$ for private key query, if $ID_i \neq ID_{cp}{}^*$ , the challenger $\mathcal{C}$ runs the algorithm $KeyGen\left(pp, msk, ID_i, [t_{ID_i}, t'_{ID_i}]\right)$ to generate the private key $SK_{ID_i}$ and sends it to $\mathcal{A}$.

②Authorization key query $Q_{\widetilde{SK}}(ID_i)$: $\mathcal{A}$ makes a mobile user's authorization key query to the challenger $\mathcal{C}$. The challenger $\mathcal{C}$ first searches the information corresponding to the identity $ID_i$ in table $T$. If it finds the tuple $(ID_i, SK_{ID_i}, \widetilde{SK_{ID_\imath}}, \widetilde{pk})$, it returns the authorization key $\widetilde{SK_{ID_\imath}}$ to the attacker $\mathcal{A}$. Otherwise, the challenger $\mathcal{C}$ runs $KeyGen()$ to obtain the private key $SK_{ID_i}$ corresponding to the identity $ID_i$, runs $AuthKeyGen(SK_{ID_i}, [t_{ID_i}, t'_{ID_i}]) \rightarrow \widetilde{SK_{ID_\imath}}$ for the mobile user, adds the table $T$ into the tuple $(ID_i, SK_{ID_i}, \widetilde{SK_{ID_\imath}}, \widetilde{pk})$, and finally the challenger $\mathcal{C}$ sends $(\widetilde{SK_{ID_\imath}}, \widetilde{pk})$ to $\mathcal{A}$. It assumes that an attacker will not issue an authorization key query for the same identity $ID_i$ if it has already issued a private key query for the same identity. Since anyone can obtain a license key $\widetilde{SK_{ID_\imath}}$ by running the mobile user's license key generation algorithm using that user's private key, the assumptions are reasonable.

③User's final decryption query $Q_{Dec}(ID_i)$: $\mathcal{A}$ submits $(ID_i, CT_{S|\omega}, \widetilde{CT_{S|\omega}})$ to $\mathcal{C}$ for the user's final decryption query. $\mathcal{C}$ first searches for the tuple $(ID_i, SK_{ID_i}, \widetilde{pk}, \widetilde{\tau}, \widetilde{CT_{S|\omega}}, CT_F)$ in table $T$. If it finds the corresponding tuple about the identity $ID_i$ tuple, the decryption algorithm $UsersDec\left(pp, SK_{ID_i}, \widetilde{pk}, \widetilde{\tau}, \widetilde{CT_{S|\omega}}, CT_F\right)$ is executed and the decryption result is returned to $\mathcal{A}$. If the corresponding tuple is not found, the challenger $\mathcal{C}$ outputs $\perp$.

3) Challenge phase. The attacker $\mathcal{A}$ submits two challenge plaintexts $(M_0, M_1)$ to the challenger $\mathcal{C}$. The challenger $\mathcal{C}$ runs $KeyEncrypt\left(pp,\ ID_{cp}, M_b, \omega\right)$ to generate the challenge ciphertext $CT_{cp|\omega}^*$ and sends $CT_{cp|\omega}^*$ to the attacker $\mathcal{A}$, where $b$ is chosen randomly at $\{0,1\}$.

4) Query phase 2. After receiving the challenge ciphertext, $\mathcal{A}$ continues to initiate the private key query, the authorization key query, and the user's final decryption query as in query phase 1. The restriction of this phase query is that $\mathcal{A}$ has never queried the private key of $ID_{cp}{}^*$ and cannot be any $S$ at the same time querying $Q_{Dec}(ID_i)$ and $Q_{sk}(ID_i)$. That is if the decryption result of $\mathcal{A}$ in the final decryption query is $M_0$ or $M_1$, then $\mathcal{C}$ outputs $\perp$ to $\mathcal{A}$.

5) Guessing phase. The attacker $\mathcal{A}$ outputs a guess value $b' \in \{0,1\}$. If $b' = b$, then the attacker $\mathcal{A}$ wins. The advantage of the attacker $\mathcal{A}$ win in the game is defined as

$$Adv_{Our,\ \mathcal{A}}^{RCCA} = |\, Pr[b' = b] - \frac{1}{2}\, |.$$

# 4. Design of VTC-OD Scheme

## 4.1 Scheme Overview

Mobile cloud computing is a very attractive service paradigm in which data computation and storage are moved from mobile devices to the cloud. Mobile users with insufficient computing power can hardly afford the huge overhead of decryption computation due to the processor performance and battery capacity so that they outsource all decryption computation to CSP. However CSP may return incorrect decryption results to save computing resources. Since most current IBE encryption schemes are constructed based on elliptic curve or bilinear groups, more bilinear pairing and power operations are involved in the algorithm. Among them, in the phase of data decryption, the computation volume of authorized users performing decryption operations grows linearly with the number of authorized users $|S|$. Since the computational outsourcing can effectively reduce the computational load of the mobile device, it is applied to improve the real-time computing efficiency. In addition, this paper mainly studies from the user's perspective since the user is only involved in the decryption phase. Therefore, the decryption outsourcing solution for cloud data sharing becomes an urgent need. Thus, an outsourcing decryption of verifiable transformed ciphertext scheme is proposed to alleviate the time-consuming problem of decryption in mobile devices, which mainly consists of four algorithms, i.e., $AuthKeyGen$, $ProxyDec$, $Verify$, $UsersDec$, as shown in **Fig. 2**.

1) $AuthKeyGen(SK_{ID_i},[t_{ID_i},t'_{ID_i}]) \to \widetilde{SK_{ID_\iota}}$. User inputs the private key $SK_{ID_i}$, the start time $t_{ID_i}$ of the authorized private key , and the end time $t'_{ID_i}$ of the authorized private key, and it outputs the authorized key $\widetilde{SK_{ID_\iota}} = (\widetilde{sk_1},\widetilde{sk_2},\sigma_{ID_i},T_i)$.
2) $ProxyDec(CT_{S|\omega},\widetilde{SK_{ID_\iota}}) \to \widetilde{CT_{S|\omega}}$. CSP inputs the transformed ciphertext $CT_{S|\omega} = (C'_1,C'_2,C'_3,C'_4,C'_5,C'_6)$ and the user's authorized private key $\widetilde{SK_{ID_\iota}} = (\widetilde{sk_1},\widetilde{sk_2},\sigma_{ID_i},T_i)$ , and it outputs the partially decrypted ciphertext of the transformed ciphertext $\widetilde{CT_{S|\omega}} = (\widetilde{sk_1},\widetilde{sk_2},\sigma_{ID_i},T_i)$.
3) $Verify(pp,pk_\sigma,\widetilde{CT_{S|\omega}}) \to (1/0)$. User inputs the system public parameter $pp$, the signed public key $pk_\sigma$ of the CSP and the partially decrypted ciphertext returned by the CSP $\widetilde{CT_{S|\omega}}$, and outputs 1 correctly and 0 otherwise.
4) $UsersDec\left(pp,SK_{ID_i},\widetilde{pk},\tilde{\tau},\widetilde{CT_{S|\omega}},CT_F\right) \to (F/\perp)$. The authorized users inputs the system parameters $pp$, the user's private key $SK_{ID_i}$, the decryption parameters $\widetilde{pk}$ and $\tilde{\tau}$ returned by the CSP, the partially decrypted ciphertext $\widetilde{CT_{S|\omega}}$ , the data ciphertext $CT_F$, and outputs the plaintext message $F$ or the error flag $\perp$.

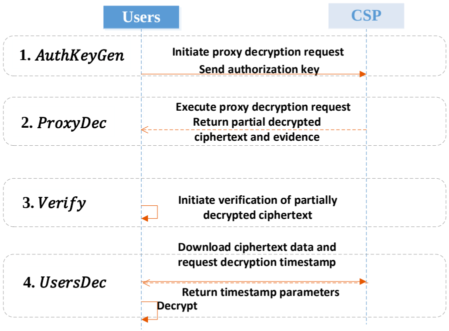

Users
CSP
1. AuthKeyGen
Initiate proxy decryption request
Send authorization key
2. ProxyDec
Execute proxy decryption request
Return partial decrypted
ciphertext and evidence
3. Verify
Initiate verification of partially
decrypted ciphertext
4. UsersDec
Download ciphertext data and
request decryption timestamp
Return timestamp parameters
Decrypt

**Fig. 2.** Outsourcing decryption of verifiable transformed ciphertext

To facilitate the description of the proposed algorithm, the definition of the symbols used in this paper is given in **Table 1**.

**Table 1.** Symbol definition

| Symbols | Description |
|---|---|
| $F$ | Files uploaded by the CP |
| $pp$ | System common parameters generated by KGC |
| $SK_{ID_i}$ | Users' authorized private keys generated by KGC |
| $\widetilde{SK_{ID_i}}$ | Authorized key generated by the users |
| $\widetilde{pk}$ | User's decryption parameters |
| $CT_F$ | Ciphertext generated by CP |
| $M$ | Symmetric key |
| $CT_{cp\|\omega}$ | IBE ciphertext generated by CP |
| $CT_{S\|\omega}$ | IBBE ciphertext generated by CSP |
| $\widetilde{CT_{S\|\omega}}$ | Partially decrypted ciphertext generated by CSP |
| $\sigma_{ID_i}$ | Authorized user signature |
| $t_{ID_i}$ | Start time of user authorized private key |
| $t'_{ID_i}$ | End time of user authorized private key |
| $t_c$ | Timestamp of the authorized user access request |

## 4.2 Generation of authorized keys

Since the existing IBE schemes whose decryption requires complex power and bilinear pairing operations, it takes a lot of time for mobile users with limited computing power to decrypt, and users who have expired the authorization period can still decrypt outsourced decrypted ciphertexts on CSP that have been accessed during the authorization period. The basic idea is that the mobile user outsources most of his complex decryption computation operations to the CSP without revealing his private key and data privacy. Therefore, the user sends the generated authorization key to the CSP by running the authorized key generation algorithm $AuthKeyGen$ to blind the private key in order not to reveal the private key.

$AuthKeyGen(SK_{ID_i},[t_{ID_i},t'_{ID_i}]) \rightarrow \widetilde{SK_{ID_i}}$. The process is executed by the user. It is based on the private key $SK_{ID_i}$ entered by the user, the start time of the authorized private key $t_{ID_i}$ and the end time of the authorized private key $t'_{ID_i}$, and outputs authorized key $\widetilde{SK_{ID_i}} = (\widetilde{sk_1},\widetilde{sk_2},\sigma_{ID_i},T_i)$. The specific calculation procedure is as follows:

First, the user generates the authorized key using his private key. The algorithm randomly selects a random number $\theta \in \mathbb{Z}_q^*$ and makes $\widetilde{pk}=\theta$, and performs a mask operation on the user's private key $SK_{ID_i} = (sk_1,sk_2,\sigma_{ID_i})$ to generate the authorized key. The authorized key is calculated by

$$\widetilde{SK_{ID_\iota}} = \begin{cases} \widetilde{sk}_1 = sk_1 = g^{\frac{1}{\alpha+H_1(ID_i)}}; \\ \widetilde{sk}_2 = {sk_2}^{\widetilde{pk}} = (y_{\{t_{ID_i}, t'_{ID_i}\}})^{r_i \cdot \widetilde{pk}} = h^{r_i \cdot \widetilde{pk} \cdot a^{t_{ID_i}} \cdot b^{z-t'_{ID_i}}}; \\ \sigma_{ID_i} = g^{\frac{1}{\beta+H_1\left(t'_{ID_i}\right)}}; \\ T_i = \left[t_{ID_i}, t'_{ID_i}\right]. \end{cases} \tag{1}$$

Second, the user initiates an authorization decryption operation request to the CSP. The user sends the authorized key $\widetilde{SK_{ID_\iota}}$ output by the algorithm to the CSP.

Finally, after the CSP accepts the request, the user sends the authorized key to the CSP and stores the decryption parameters $\widetilde{pk}$ locally.

### 4.3 Partial decryption of transformed ciphertext

CSP can provide outsourcing decryption service for the user because of its powerful computing power. When CSP receives the authorized key $\widetilde{SK_{ID_\iota}}$ of the user's decryption request, he gets the current time $t_x$ and then runs the partial decryption algorithm $ProxyDec$ to perform partial decryption of transformed ciphertext for the user and sends the partially decrypted ciphertext to the user.

$ProxyDec(CT_{S|\omega}, \widetilde{SK_{ID_\iota}}) \rightarrow \widetilde{CT_{S|\omega}}$. The process is performed by CSP. It inputs the transformed ciphertext $CT_{S|\omega} = (C'_1, C'_2, C'_3, C'_4, C'_5, C'_6)$, the user's authorized private key $\widetilde{SK_{ID_\iota}} = (\widetilde{sk}_1, \widetilde{sk}_2, \sigma_{ID_i}, T_i)$, and outputs partial decryption of the transformed ciphertext $\widetilde{CT_{S|\omega}} = (\widetilde{DK_{ID_\iota}}, \varphi)$. The specific calculation procedure is as follows:

First, the CSP determines whether the end time of the authorized user $T_i = \left[t_{ID_i}, t'_{ID_i}\right]$ has been forged or tampered with based on $t'_{ID_i}$ in the user's authorized private key. The verification formula is

$$e\left(g^{H_1\left(t'_{ID_i}\right)} \cdot g^{\beta}, g^{\frac{1}{\beta+H_1\left(t'_{ID_i}\right)}}\right) \overset{?}{=} e(g, g). \tag{2}$$

If Formula (2) does not hold, CSP returns the error flag ⊥ and marks the identity of this authorized user as a revoked user. Otherwise, the CSP marks the authorized user as revoked based on $\widetilde{sk}_2 = (y_{\{t_{ID_i}, t'_{ID_i}\}})^{r_i \cdot \widetilde{pk}} = h^{r_i \cdot \widetilde{pk} \cdot a^{t_{ID_i}} \cdot b^{z-t'_{ID_i}}}$, which further determines whether the authorization period of this user has expired or not. Applying Formula (1), if the multidimensional range derivative function is not computable to output the error flag ⊥, it indicates that $t_x \leq t_{ID_i}$ or $t_x \geq t'_{ID_i}$, i.e., the authorization period of the user's private key has expired. In this case, the CSP terminates the partial decryption operation. If the multidimensional range derivation function is computable, it indicates that $t_{ID_i} \leq t_x$ and $t'_{ID_i} \geq t_x$, i.e., the authorization period of the user's private key has not expired. The multidimensional range derivation function is specifically computed by

$$F_x = F_{t_{ID_i} \leq t_x, t'_{ID_i} \geq t_x}(y_{\{t_{ID_i}, t'_{ID_i}\}})^{\ r_i \cdot \widetilde{pk}} = (y_{\{t_x, t_x\}})^{r_i \cdot \widetilde{pk}}. \tag{3}$$

After calculating the $F_x$, it is combined with formulas (4)-(6) to calculate

$$\varphi = \frac{C'_5}{F_x} = \frac{h^{r_j \cdot a^{t_c} \cdot b^{z-t_c}}}{h^{r_i \cdot \widetilde{pk} \cdot a^{t_x} \cdot b^{z-t_x}}}. \tag{4}$$

From formula (4), it shows that the calculation of $\varphi$ is different from the calculation of $\varepsilon$ in the decryption step. The decryption operation was previously considered to be performed by the user on his local machine. However, considering the limited computing power of the users of mobile devices, some decryption operations on the user side need to be handed over to the CSP. The decryption operations are performed on the user's local machine.

Second, CSP calculates $sk_1 = g^{\frac{1}{\alpha+H_1(ID_i)}}$ based on the user's authorization key by

$$\tilde{B} = \left(e\left(C_1', h^{\rho_{i,S}(\alpha)}\right) \cdot e(sk_1, C_2')\right)^{\frac{1}{\Pi_{j=1,j\neq i}^{S} H_1(ID_j)}}, \tag{5}$$

where $\rho_{i,S}(\alpha) = \frac{1}{\alpha} \cdot \left(\prod_{j=1,j\neq i}^{S} (\alpha + H_1(ID_j)) - \prod_{j=1,j\neq i}^{S} H_1(ID_j)\right)$. Substituting $C_1' = {g_1}^{-r'}$ and $C_2' = h^{r' \cdot \boldsymbol{\Pi}_{j=1}^{n}\left(\alpha + H_1(ID_j)\right)}$ into formula (5), it uses polynomial interpolation to calculate

$$\tilde{B} = \left(e(g^{\alpha \cdot (-r')}, h^{\rho_{i,S}(\alpha)}) \cdot e(g^{\frac{1}{\alpha+H_1(ID_i)}}, h^{r' \cdot \boldsymbol{\Pi}_{j=1}^{n}\left(\alpha + H_1(ID_j)\right)})\right)^{\frac{1}{\Pi_{j=1,j\neq i}^{S} H_1(ID_j)}} = e(g,h)^{r'}. \tag{6}$$

After obtaining $\tilde{B}$, it substitutes $\tilde{B}$ into $C_3' = H_2(e(g,h)^{r'}) \cdot h^s$, and yields $C_3' = H_2(e(g,h)^{r'}) \cdot h^s$. From $h^s$ it can be calculated by

$$\widetilde{DK_{ID_\iota}} = \frac{C_6'}{e(h^s, C_4')}. \tag{7}$$

Substituting $C_4' = (u \cdot \mu^{\omega})^{r \cdot \left(\alpha + H_1(ID_{cp})\right)}$ and $C_6' = M \cdot e\left(g, h^{a^{tc} \cdot b^{z-tc}}\right)^{r_j} \cdot e\left(h^{r \cdot s}, (u \cdot \mu^{\omega})^{\left(\alpha + H_1(ID_{cp})\right)}\right)$ into formula (7), it has

$$\widetilde{DK_{ID_\iota}} = \frac{C_6'}{e(h^s, C_4')} = M \cdot e\left(g, h^{a^{tc} \cdot b^{z-tc}}\right)^{r_j}. \tag{8}$$

After computing $\widetilde{DK_{ID_\iota}}$, CSP applies the evidence generation algorithm $Prove(pp, sk_\sigma, x)$ for the verifiable random function, and randomly selects a random number $\delta \in \mathbb{Z}_q^*$ such that $x = \delta || \widetilde{DK_{ID_\iota}}$. Let the signature key pair of CSP be $(pk_\sigma, sk_\sigma) = (\gamma, {g_2}^{\gamma})$. Applying verifiable random functions (VRF) [33-34], the VRF evidence generation algorithm outputs $P = (x, y, \pi)$, where $y = \mathrm{e}(g_2, g_2)^{\frac{1}{H_1(x)+\gamma}}$ is the VRF function value, $\pi = {g_2}^{\frac{1}{H_1(x)+\gamma}}$ is the evidence.

Applying formulas (4) and (8) and $P$, CSP calculates the partially decrypted ciphertext by

$$\widetilde{CT_{S|\omega}} = \left(\widetilde{DK_{ID_\iota}}, \varphi, x, y, \pi\right). \tag{9}$$

Finally, the CSP sends the partial decryption result of the computed transformed ciphertext $\widetilde{CT_{S|\omega}}$ to the user.

### 4.4 Verification of partial decryption of transformed ciphertext

Although CSP can provide users with powerful decryption outsourcing computing services, CSP may not execute the outsourced decryption requests initiated by users, and try to falsify the outsourced decryption results to return to users. Therefore, the user needs to verify the correctness of the decrypted ciphertext sent by the CSP before performing the decryption operation. When the user obtains the partially decrypted ciphertext returned by the CSP $\widetilde{CT_{S|\omega}}$, he uses the signed public key $pk_\sigma$ of the CSP to verify the correctness of the partially decrypted ciphertext.

$Verify(pp, pk_\sigma, \widetilde{CT_{S|\omega}}) \rightarrow (1/0)$. The process is performed by the user. It inputs the system public parameter $pp$, the signed public key $pk_\sigma$ of the CSP and the partially decrypted ciphertext returned by the CSP $\widetilde{CT_{S|\omega}} = (\widetilde{DK_{ID_i}}, \varphi, x, y, \pi)$, and outputs 1 which means the partially decrypted ciphertext is correct, and outputs 0 which means CSP returns the wrong outsourced decryption result. The specific calculation process is as follows:

First, when the user obtains the partially decrypted ciphertext returned by the CSP, he applies the CSP's public key $pk_\sigma$ to initiate a verification request on the partially decrypted ciphertext returned by the CSP.

Second, the user runs the decryption verification algorithm $Verify(pp, pk_\sigma, \widetilde{CT_{S|\omega}})$ of the transformed ciphertext to judge the result returned by CSP, and the verification formula is

$$\begin{cases} y \overset{?}{=} e(g_2, \pi) \\ e\left({g_2}^{H_1(x)} \cdot pk_\sigma, \pi\right) \overset{?}{=} e(g_2, g_2). \end{cases} \tag{10}$$

If formula (10) does not hold, the algorithm outputs 0 and the user terminates the decryption operation. If the algorithm outputs 1, it indicates that the partially decrypted ciphertext returned by the CSP is correct, i.e., the verifiability of the outsourced decryption is achieved.

Finally, if the user gets an output value of 1 from the verification algorithm, the final decryption algorithm can be executed.

## 4.5 Final decryption of transformed ciphertext

After the user receives the partially decrypted ciphertext $\widetilde{CT_{S|\omega}}$ from the CSP, he downloads $\widetilde{CT_{S|\omega}}$ and the data ciphertext $CT_F$ to the local area, and decrypts the data. The user does it with the assistance of the CSP to connect to the network and get the timestamp $t_y$ of the decryption request from the CSP, then runs the final decryption algorithm $UsersDec\left(pp, SK_{ID_i}, \widetilde{pk}, \tilde{\tau}, \widetilde{CT_{S|\omega}}, CT_F\right)$ for the transformation ciphertext. Eventually, the transformed ciphertext is decrypted to obtain the plaintext.

1) Getting timestamp

First, the CSP makes a judgment according to formula (1). If the multidimensional range derivative function is not computable to output the error flag ⊥, it indicates that the time $t_y$ of the user's decryption request is not within the user's authorized time range $[t_x, t'_{ID_i}]$, and the algorithm terminates the decryption operation. If the multidimensional range derivation function is computable, it indicates that the time $t_y$ of the user's decryption request is within the user's authorized time range $[t_x, t'_{ID_i}]$, i.e., the authorization period of the user's private key has not expired. Its calculation formula is

$$F_y = F_{t_x \le t_y, t'_{ID_i} \ge t_y} F_x = h^{\widetilde{pk} \cdot r_i \cdot a^{t_y} \cdot b^{z-t_y}}. \tag{11}$$

Next, after deriving $F_y$ according to formulas (5)-(11), it is combined with formula (3) to calculate

$$\tilde{\varphi} = \frac{F_x}{F_y} = \frac{h^{r_i \cdot \widetilde{pk} \cdot a^{t_x} \cdot b^{z-t_x}}}{h^{\widetilde{pk} \cdot r_i \cdot a^{t_y} \cdot b^{z-t_y}}}. \tag{12}$$

Finally, the CSP returns the time-dependent parameter $\tilde{\tau} = (t_y, \tilde{\varphi})$ to this user.

2) Final decryption of the transformed ciphertext

The final decryption algorithm$UsersDec$ is run to obtain the plaintext when the user gets the time-related parameter $\tilde{\tau}$ of the decryption request from the CSP.

$UsersDec\left(pp, SK_{ID_i}, \widetilde{pk}, \tilde{\tau}, \widetilde{CT_{S|\omega}}, CT_F\right) \rightarrow (F/\perp)$. The authorized users execute the algorithm. It inputs the system parameters $pp$, the user's private key $SK_{ID_i}$, the decryption parameters $\widetilde{pk}$ and $\tilde{\tau}$ returned by the CSP, the partially decrypted ciphertext $\widetilde{CT_{S|\omega}}$ and the data ciphertext $CT_F$, and outputs the plaintext $F$ or the error flag $\perp$. The specific computation process is as follows:

First, the algorithm calculates the timestamps $t_y$ and $sk_2 = h^{r_i \cdot a^{t_{ID_i}} \cdot b^{z-t'_{ID_i}}}$ for the user's decryption request by

$$\widetilde{F_y} = F_{t_{ID_i} \leq t_y, t'_{ID_i} \geq t_y} (y_{\{t_{ID_i}, t'_{ID_i}\}})^{r_i} = h^{r_i \cdot a^{t_y} \cdot b^{z-t_y}} = (y_{\{t_y, t_y\}})^{r_i}. \tag{13}$$

Second, from formulas (5)-(13) and $\tilde{\tau}$, $\varphi$ returned by CSP, the user's $\widetilde{pk} = \theta$ can be calculated by

$$\widetilde{A} = \varphi \cdot \widetilde{F_y}^{\widetilde{pk}} \cdot \tilde{\varphi} = h^{r_j \cdot a^{t_c} \cdot b^{z-t_c}}. \tag{14}$$

From formula (14), the construction of $\tilde{A}$ ensures that only legitimate users within the authorization period can eventually decrypt the transformed ciphertext, and malicious CSP or illegal users cannot access and decrypt it.

Finally, applying $\widetilde{pk} = \theta$, $CT_F$, $\tilde{A}$ and $\widetilde{DK_{ID_\iota}}$, it can obtain the plaintext by

$$\frac{CT_F}{\widetilde{DK_{ID_\iota}} \cdot \mathrm{e}(g, \tilde{A})^{-1}} = \frac{F \cdot M}{M \cdot e\left(g, h^{a^{t_c} \cdot b^{z-t_c}}\right)^{r_j} \cdot e(g, h^{r_j \cdot a^{t_c} \cdot b^{z-t_c}})^{-1}} = F.$$

The algorithm finally returns the output plaintext $F$ to this user.

# 5. Security analysis

## 5.1 Correctness of ciphertext

Ciphertext correctness means that when a user sends the authorization key $\widetilde{SK_{ID_\iota}}$ to the CSP, CSP correctly executes the partial decryption algorithm of the transformed ciphertext and correctly generates the partially decrypted ciphertext $\widetilde{CT_{S|\omega}}$ so that the user who has not been exceeded the authorization period can correctly decrypt the ciphertext and obtain the original plaintext with his private key.

**Lemma 2.** Given the partially decrypted ciphertext $\widetilde{CT_{S|\omega}}$ correctly executed by the CSP, any authorized user within the authorization period, i.e., $ID_i \in S$ and $\omega = \omega'$, can use the private key $SK_{ID_i}$ provided by KGC and $\tilde{\tau}$ returned by CSP to perform the final decryption of the transformed ciphertext and obtain the originally shared plaintext $F$.

**Proof:** for the partially decrypted ciphertext $\widetilde{CT_{S|\omega}}$ which is correctly transformed by the CSP, any user under the condition that $ID_i \in S$ and $\omega = \omega'$ can perform the decryption algorithm by entering his private key $SK_{ID_i}$ and $\tilde{\tau}$ returned by the CSP.

First, the algorithm uses $sk_2$ in the private key to compute

$$\widetilde{F_y} = F_{t_{ID_i} \leq t_y, t'_{ID_i} \geq t_y} \left(y_{\left\{t_{ID_i}, t'_{ID_i}\right\}}\right)^{r_i} = (y_{\{t_{ID_i}, t'_{ID_i}\}})^{r_i \cdot a^{t_d - t_{ID_i}} \cdot b^{t'_{ID_i} - t_d} = (y_{\{t_y, t_y\}})^{r_i}}.$$

Second, applying $\widetilde{F_y}$, $\tilde{\varphi} = \frac{h^{r_i \cdot \widetilde{pk} \cdot a^{t_x} \cdot b^{z-t_x}}}{h^{\widetilde{pk} \cdot r_i \cdot a^{t_y} \cdot b^{z-t_y}}}$, and $\varphi = \frac{h^{r_j \cdot a^{t_c} \cdot b^{z-t_c}}}{h^{r_i \cdot \theta \cdot a^{t_x} \cdot b^{z-t_x}}}$, it has

$$\tilde{A} = \varphi \cdot \widetilde{F_y}^{\widetilde{pk}} \cdot \tilde{\varphi} = \frac{h^{r_j \cdot a^{t_c} \cdot b^{z-t_c}}}{h^{r_i \cdot \widetilde{pk} \cdot a^{t_x} \cdot b^{z-t_x}}} \cdot h^{\widetilde{pk} \cdot r_i \cdot a^{t_y} \cdot b^{z-t_y}} \cdot \frac{h^{r_i \cdot \widetilde{pk} \cdot a^{t_x} \cdot b^{z-t_x}}}{h^{\widetilde{pk} \cdot r_i \cdot a^{t_y} \cdot b^{z-t_y}}} = h^{r_j \cdot a^{t_c} \cdot b^{z-t_c}}.$$

Finally, the plaintext $F$ can be obtained applying $\widetilde{pk}=\theta, CT_F$, $\tilde{A}$ and $\widetilde{DK_{ID_\imath}}$ by

$$Left = \frac{CT_F}{\widetilde{DK_{ID_\imath}} \cdot \mathrm{e}\left(g, \tilde{A}\right)^{-1}} = \frac{F \cdot M}{M \cdot e\left(g, h^{a^{t_c} \cdot b^{z-t_c}}\right)^{r_j} \cdot e(g, h^{r_j \cdot a^{t_c} \cdot b^{z-t_c}})^{-1}} = Right.$$

So when $ID_i \in S$, it has $UsersDec\left(pp, SK_{ID_i}, \widetilde{pk}, \widetilde{\tau}, \widetilde{CT_{S|\omega}}, CT_F\right) = F$.

From the above proof process, it is obvious that the partially decrypted ciphertext which is correctly transformed by the CSP can be decrypted by the authorized unexpired user with the private key to obtain the original plaintext.

## 5.2 Proof of security

### 5.2.1 Resisting unauthorized ciphertext decryption attacks

Unauthorized ciphertext decryption attack refers to the possibility that unauthorized users and semi-trusted CSP may try to decrypt partially decrypted ciphertexts without private keys by means of spoofing, eavesdropping, brute force cracking, etc.

Assume that the attacker $\mathcal{A}$ can breach the proposed scheme in this paper with a negligible advantage $Adv_{Our,\ \mathcal{A}}^{RCCA}$ in probabilistic polynomial time. It can construct an algorithm $\mathcal{B}$ simulating the interaction between the challenger $\mathcal{C}$ and the attacker $\mathcal{A}$. The specific RCCA secure instance is described as follows.

1) Setup phase: given the system security parameter $\lambda$ and the maximum number of receivers $m$ allowed in encryption. It first runs the algorithm $Setup(1^\lambda, m)$ to obtain the group-related parameters $(q, \mathbb{G}, \mathbb{G}_T, e)$, where $\mathbb{G}$ and $\mathbb{G}_T$ are multiplicative cyclic groups of order prime $q$, $e: \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a bilinear map, and $g$ and $h$ are two generating elements of the group $\mathbb{G}$. Then it chooses the two hash functions $H_1: \{0,1\}^* \to \mathbb{Z}_q^*$and $H_2: \mathbb{G}_T \to \mathbb{G}$, several random numbers $(\alpha, \beta, \gamma, v) \in \mathbb{Z}_q^*$, computes $u \cdot \mu = h^\gamma, (u \cdot \mu)^\alpha = h^{\alpha\gamma}, pk_{sign} = g^\beta$, and sets the maximum number $m = n$ of data users allowed to access the same data. Finally, the algorithm outputs the master key $msk = \alpha$ and the system public parameters $pp = \left(q, \mathbb{G}, \mathbb{G}_T, e(g,h), g_1, pk_{sign}, u, u^\alpha, h, h^\alpha, h^{\alpha^2}, \ldots, h^{\alpha^n}, H_1, H_2\right)$.

2) Query phase 1: The attacker $\mathcal{A}$ can repeat the following query to the challenger $\mathcal{C}$.

①Private key query $Q_{SK}(ID_i)$: the attacker $\mathcal{A}$ initiates a private key query request. First,$\mathcal{A}$ sends $\mathcal{B}$ the identity $ID_i$ that asks for the. If $ID_i = ID^*$, $\mathcal{B}$ returns the invalid flag $\perp$; otherwise, $\mathcal{B}$ queries the identity $ID_i$ in table $T$ to process as follows. If $\left(ID_i, SK_{ID_i}\right)$ exists, then $\mathcal{B}$ returns the private key $SK_{ID_i}$ and its corresponding authorization period $[t_{ID_i}, t'_{ID_i}]$ to the attacker $\mathcal{A}$. Otherwise, $\mathcal{B}$ applies the identity and authorization period $[t_{ID_i}, t'_{ID_i}]$ of the attacker $\mathcal{A}$ to compute $SK_{ID_i} = (sk_1 = g^{\frac{1}{\alpha + H_1(ID_i)}}, sk_2 = h^{r_i \cdot a^{t_{ID_i}} \cdot b^{z - t'_{ID_i}}}, \sigma_{ID_i} = g^{\frac{1}{\beta + H_1\left(t'_{ID_i}\right)}}$ and return it to the attacker $\mathcal{A}$, while recording $(ID_i, w_i, SK_{ID_i})$ in Table $T_1$.

②Authorization key query: The attacker $\mathcal{A}$ makes a user authorization key query to the challenger $\mathcal{C}$. The challenger $\mathcal{C}$ first searches the information corresponding to the identity $ID_i$ in table $T$. If it finds the tuple $\left(ID_i, \widetilde{SK_{ID_\imath}}, \widetilde{pk}\right)$, then it directly returns the authorization key $\widetilde{SK_{ID_\imath}}$ to the attacker$\mathcal{A}$. Otherwise, the challenger $\mathcal{C}$ runs the private key generation algorithm to obtain the private key $SK_{ID_i}$ corresponding to the identity $ID_i$, and then runs the user authorization key generation algorithm $AuthKeyGen\left(SK_{ID_i}, \left[t_{ID_i}, t'_{ID_i}\right]\right)$ to compute $\widetilde{SK_{ID_\imath}} = (sk_1 = g^{\frac{1}{\alpha + H_1(ID_i)}}, \widetilde{sk_2} = {sk_2}^{\widetilde{pk}} = h^{r_i \cdot \widetilde{pk} \cdot a^{t_{ID_i}} \cdot b^{z - t'_{ID_i}}},\ \sigma_{ID_i} = g^{\frac{1}{\beta + H_1\left(t'_{ID_i}\right)}}, T_i = [t_{ID_i}, t'_{ID_i}])$

and add the tuple $(ID_i, \widetilde{SK_{ID_\iota}}, \widetilde{pk})$ to the table $T$. Finally, $\mathcal{C}$ sends $(\widetilde{SK_{ID_\iota}}, \widetilde{pk})$ to $\mathcal{A}$.

③Partial decryption query $\mathcal{Q}_{CSP}(ID_i)$: the attacker $\mathcal{A}$ submits the authorization key and ciphertext $CT_{S|\omega}$ corresponding to the identity $ID_i$ to the challenger $\mathcal{C}$ for decryption query. The challenger $\mathcal{C}$ first runs the private key generation algorithm to generate the private key $\widetilde{SK_{ID_\iota}}$ corresponding to the identity $ID_i$ . Then it executes the decryption algorithm $ProxyDec(CT_{S|\omega}, \widetilde{SK_{ID_\iota}})$ to compute the partially decrypted ciphertext $\widetilde{CT_{S|\omega}} = (\widetilde{DK_{ID_\iota}} = M \cdot e\left(g, h^{a^{t_c} \cdot b^{z-t_c}}\right)^{r_j}, \varphi = \frac{h^{r_j \cdot a^{t_c} \cdot b^{z-t_c}}}{h^{r_i \cdot \widetilde{pk} \cdot a^{t_x} \cdot b^{z-t_x}}})$, and finally returns $\widetilde{CT_{S|\omega}}$ to the attacker $\mathcal{A}$.

④Final decryption query $\mathcal{Q}_{Dec}(ID_i)$: the attacker $\mathcal{A}$ submits $(ID_i, CT_{S|\omega}, \widetilde{CT_{S|\omega}})$ to the challenger $\mathcal{C}$ for the final decryption query of the authorized user. The challenger $\mathcal{C}$ first searches for the tuple $(ID_i, SK_{ID_i}, \widetilde{pk}, \widetilde{\tau}, \widetilde{CT_{S|\omega}}, CT_F)$in table $T$. If it finds the corresponding tuple about identity $ID_i$ , it executes the authorized user decryption algorithm $UsersDec\left(pp, SK_{ID_i}, \widetilde{pk}, \widetilde{\tau}, \widetilde{CT_{S|\omega}}, CT_F\right)$ to compute $\widetilde{F_y} = (y_{\{t_y,t_y\}})^{r_i}$ and $\tilde{A} = h^{r_j \cdot a^{t_c} \cdot b^{z-t_c}}$ and returns the decryption result to the attacker $\mathcal{A}$. If the corresponding tuple is not found, the challenger $\mathcal{C}$ outputs $\perp$.

3) Challenge phase. The attacker $\mathcal{A}$ submits two challenge plaintexts $(M_0, M_1)$ to the challenger $\mathcal{C}$ .The challenger $\mathcal{C}$ first runs $KeyEncrypt\left(pp,\ ID_{cp}, M_b, \omega\right)$ to generate the original ciphertext $CT^*_{cp|\omega}$ , where $b$ is chosen randomly at $\{0,1\}$, then runs $Trans\left(pp, Tk_{cp\to S|\omega}, CT_{cp|\omega}, t_c\right)$ to convert the ciphertext $CT^*_{cp|\omega}$ to $CT^*_{S|\omega}$ using the authorization token $Tk_{cp\to S|\omega}$ , and finally runs $ProxyDec(CT_{S|\omega}, \widetilde{SK_{ID_\iota}})$ to compute a partially transformed decrypted ciphertext $\widetilde{CT_{S|\omega}} = (\widetilde{DK_{ID_\iota}} = M \cdot e(g, h^{a^{t_c} \cdot b^{z-t_c}})^{r_j}, \varphi = \frac{h^{r_j \cdot a^{t_c} \cdot b^{z-t_c}}}{h^{r_i \cdot \widetilde{pk} \cdot a^{t_x} \cdot b^{z-t_x}}})$ and send it to the attacker$\mathcal{A}$ as a challenge.

4) Query phase 2. In addition to the restrictions described in the RCCA game, the attacker $\mathcal{A}$ continues with the queries in query phase 1.

5) Guessing phase. The attacker $\mathcal{A}$ outputs a guess $b' \in \{0,1\}$, and if $b' = b$, then the attacker $\mathcal{A}$ wins.

Probabilistic analysis: if the attacker $\mathcal{A}$ can break the security of RCCA in the above RCCA security game with negligible probability $|\Pr[b' = b] - \frac{1}{2}|$ of the VTC-OD scheme, the security of RCCA can be breached. But the above probability depends on the random number $b'$ used by the attacker $\mathcal{A}$ and the challenger $\mathcal{C}$, and the probability of the random number $b'$ is $\frac{1}{2}$ and non-negligible, so the security of RCCA cannot be breached, i.e., it can resist unauthorized access attacks.

### 5.2.2 Resisting forgery and repudiation attacks

The forgery attack means that a semi-trusted CSP may return a random outsourced decryption message without actually performing the outsourced decryption. The repudiation attack means that even if the CSP returns the correct result, the user makes a false statement to accuse the CSP of returning an incorrect outsourced decryption result to evade outsourcing decryption fee.

Referring to the verifiable random function VRF has the binding property, the value $y$ of the VRF function in the verification auxiliary message $p$ returned by the CSP can be used to achieve verifiability of the outsourced decryption. In the process of CSP executing the partial decryption, the CSP uses the signed private key $sk_\sigma$ to compute the VRF evidence $\pi$ on the combination $\delta||\widetilde{DK_{ID_\iota}}$. If the CSP incorrectly performs the outsourced decryption, it sends an incorrectly transformed outsourced decryption ciphertext to the authorized user, i.e., $\widetilde{CT_{S|\omega}}^* \neq \widetilde{CT_{S|\omega}}$. In this case, the authentication algorithm of the VRF results in that formula (9) does not hold, i.e., $y \neq \mathrm{e}(g_2, \pi)$ and $\mathrm{e}(g_2{}^{H_1(x)} \cdot pk_\sigma, \pi) \neq \mathrm{e}(g_2, g_2)$, and the authorized user will recover a different combination from the outsourced decrypted ciphertext, i.e.$\delta^*||\widetilde{DK_{ID_\iota}}^* \neq \delta||\widetilde{DK_{ID_\iota}}$. Because, the VRF binding ensures that for different inputs $x$, the VRF function value $y$ is different. Therefore, $y$ does not match $x$ and the outsourced decryption is incorrect.

If the authorized user accuses the CSP of returning an incorrect outsourced decryption ciphertext, he needs to forge an evidence $\pi$. Similarly, the CSP executes the verification algorithm of the VRF function. If it outputs 0, formula (9) does not hold, i.e., $y \neq \mathrm{e}(g_2, \pi)$ and $\mathrm{e}(g_2{}^{H_1(x)} \cdot pk_\sigma, \pi) \neq \mathrm{e}(g_2, g_2)$, and the authorized user lies. Therefore, the algorithm can resist the forgery attack of CSP and the repudiation attack of authorized users.

## 6. Performance analysis

### 6.1 Theoretical analysis

In this section, **Table 2** gives a comparison of the computational overhead of decryption between the scheme VTC-OD in this paper and the attribute-based outsourced decryption scheme FA-ABE [3] and the revocable identity-based broadcast proxy re-encryption scheme RIB-BPRE [29]. Both the proposed scheme and the compared schemes dynamically share encrypted data by updating the re-encrypted key and generating a re-encrypted ciphertext, and both of the above compared schemes are representative of their respective fields. The exponential operations and bilinear pairing calculations are mainly considered in the computational overhead. The hash function is omitted here since it is much smaller than the operation of bilinear pairing and exponential in the group. In the table, $|\boldsymbol{S}|$ denotes the total number of authorized user identity sets $\boldsymbol{S}$, $|\boldsymbol{I}|$ denotes the number of attributes that satisfy the access policy, $\boldsymbol{t_e}$ denotes the time spent to perform one exponential operation, and $\boldsymbol{t_p}$ denotes the time spent to perform one bilinear pairing operation.

As can be seen from **Table 2**, the proposed scheme needs to perform the exponential operation in group $\mathbb{G}$ twice in the user authorization key generation phase. In the partial decryption phase of CSP, the computation overhead of scheme FA-ABE grows linearly with the number of attributes of the access policy, and the computation overhead of RIB-BPRE is 0 because it does not implement outsourced decryption, and the computation overhead of VTC-OD grows linearly with the authorized users |S|. In the verification phase, VTC-OD performs one more bilinear pairing operation than FA-ABE, and the overhead of RIB-BPRE is 0 because the verifiability is not implemented. In the final user decryption phase, the cost of RIB-BPRE is linearly related to the authorized users $|\boldsymbol{S}|$, and VTC-OD performs one less exponential operation than FA-ABE, and its computational overhead is slightly smaller.

**Table 2.** Comparison of the computational overhead

| Algorithms | $\boldsymbol{AuthKeyGen}$ | $\boldsymbol{ProxyDec}$ | $\boldsymbol{Verify}$ | $\boldsymbol{UsersDec}$ |
|---|---|---|---|---|
| FA-ABE | —— | $(\lvert I\rvert + 1)t_e + (3\lvert I\rvert + 3)t_p$ | $t_e + 2t_p$ | $3t_e + 2t_p$ |
| RIB-BPRE | —— | —— | —— | $O(\lvert S\rvert)t_e + 4t_p$ |
| VTC-OD | $2t_e$ | $O(\lvert S\rvert)t_e + 4t_p$ | $t_e + 3t_p$ | $2t_e + 2t_p$ |

## 6.2 Experimental analysis

To further compare the performance of the proposed algorithm, we choose the experimental environment of CSP with AliCloud shared standard (1 core 2G) ECS and universal (4 core 16G) ECS, operating system Ubuntu 20.04 64-bit, and the experimental environment of user with Xiaomi Mi MIX2 cell phone (Snapdragon 835). The elliptic curve chosen for the experiment is Type A(y^2=x^3+x), so $q$ is a 160-bit prime, $\mathbb{G}$ element has a size of 256 bits. JPBC library is used in the experiments to implement the proposed and compared schemes. Run 20 times for each experimental to obtain the average execution time as the final experimental data, set the maximum of the authorized users $m$ = 100, and vary |S| from 20 to 100. Considering that this paper focuses on outsourcing decryption, only the decryption algorithms are compared here to analyze the performance experimentally.

1) Computational overhead on the user side

The computational overhead at the user side refers to the execution time of the three-step operations of the mobile user to perform the authorized key generation, the partial transformed ciphertext decryption verification, and the final transformed ciphertext decryption.

The execution time of authorization key generation $\boldsymbol{AuthKeyGen}(.)$ is shown in **Fig. 3**. The overhead of both RIB-BPRE and FA-ABE is 0 since they do not execute this operation. The overhead of VTC-OD is greater and independent of the number of authorized users |S| since the mobile user blinds his private key in this stage, and he is independent of the other users. The execution time of the partial decryption verification $\boldsymbol{Verify}(.)$ is shown in **Fig. 4**. The computational overhead of RIB-BPRE is 0 since it does not perform the verification process on the transformed ciphertext of CSP. The computational overhead of VTC-OD is slightly larger than that of FA-ABE since VTC-OD performs one more bilinear pairing operation when verifying the partially decrypted ciphertext returned by CSP.

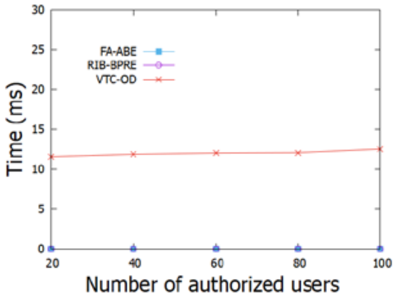


**Fig. 3.** Time of the authorization key generation

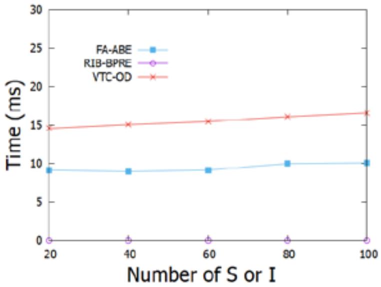


**Fig. 4.** Time of partial decryption verification

The execution time of the final decryption $\boldsymbol{UsersDec}(.)$ is shown in **Fig. 5**. The cost of VTC-OD is independent of the number of authorized users and is less than that of FA-ABE, since it performs one less exponential operation when users perform the final decryption of the transformed ciphertext. Moreover, the overhead of VTC-OD is also lower than that of RIB-BPRE, since VTC-OD outsources the mobile user's complex bilinear pairing operations and exponential operations to the CSP so that the mobile user only performs a small part of the decryption computation, which greatly reduces the computation overhead of the mobile user.

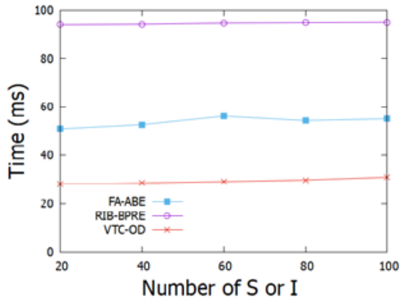


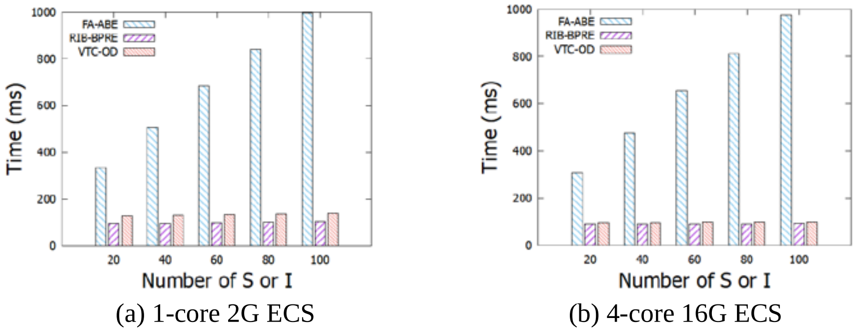
**Fig. 5.** Time of the final decryption

2) Computational overhead on the CSP side

The computation overhead of the CSP is the time that the CSP calculates the decryption timestamp parameter or performs the partial decryption of the transformed ciphertext based on the decryption request from the mobile user. For the transformation ciphertext partial decryption $\boldsymbol{ProxyDec}(.)$, CSPs with different performance execute the experiments, and the results are shown in **Fig. 6** (a) and (b), respectively.

When using a 1-core 2G ECS, it can be seen in **Fig. 6** (a) that the computation overhead of FA-ABE is much larger than that of RIB-BPRE and VTC-OD, since the computation overhead of CSP for FA-ABE is positively related to the number of attributes of access policy when computing both exponential and bilinear pairing operations. However, the computation overhead of CSP for VTC-OD is slightly larger than that of RIB-BPRE, since VTC-OD outsources part of the user’s decryption to CSP for calculation, and performs more operations than RIB-BPRE. In addition, the computation overhead of FA-ABE grows rapidly with the number of attributes of the access policy. Also, the overhead of RIB-BPRE and VTC-OD grows linearly with the number of authorized users, but its growth is slow.

By comparing **Fig. 6** (a) and (b), it can be seen that when using a 4-core 16G ECS, the execution time of CSP for FA-ABE, RIB-BPRE and VTC-OD is approximately 30ms, 10ms, and 41ms lower than that of using a 1-core 2G ECS, respectively. The reason is that CSP equipped with better performance has faster computation speed and shorter outsourced decryption time.



(a) 1-core 2G ECS (b) 4-core 16G ECS

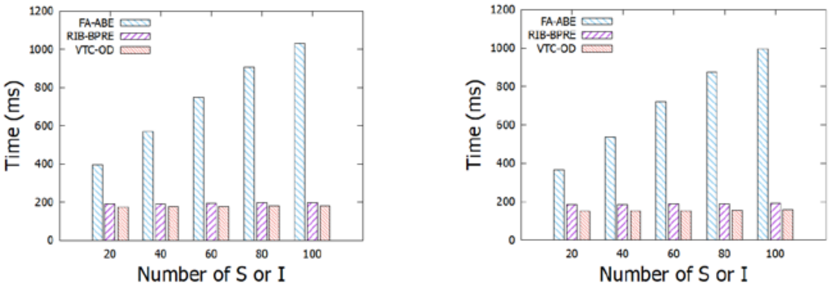
**Fig. 6.** Partial decryption time of transformed ciphertext

3) Total computational overhead of decryption

The total computation overhead of decryption refers to the sum of the time consumed by the user and the CSP in the decryption, and contains four algorithms: the generation algorithm of the authorized key, the partial decryption algorithm of the transformed ciphertext, the verification algorithm of the partial decryption of the transformed ciphertext, and the final decryption of the transformed ciphertext. Thus, CSPs with different performance execute the experiments, and the results are shown in **Fig. 7** (a) and (b), respectively.

When using a 1-core 2G ECS, it can be seen in **Fig. 7** (a) that the total computation time of VTC-OD is smaller than that of RIB-BPRE since the user of VTC-OD outsources the complex bilinear pairing and power operations to the computationally powerful CSP, which reduces the user's computation overhead and makes the total decryption overhead slightly smaller. In addition, the total computation time of FA-ABE grows linearly and rapidly with the number of attributes, but that of RIB-BPRE and VTC-OD grows slowly with the number of authorized users and remains the same. The reason is that although both FA-ABE and VTC-OD implement outsourced decryption, the computation overhead at the CSP side for FA-ABE increases significantly with the number of attributes.

By comparing **Fig. 7** (a) and (b), it can be seen that when using a 4-core 16G ECS, the execution time of CSP for FA-ABE, RIB-BPRE and VTC-OD is about 30ms, 8ms and 28ms less than that of using a 1-core 2G ECS, and the total decryption time of VTC-OD is the smallest. Thus, it can be concluded that VTC-OD reduces the decryption time of mobile devices with limited computing power, since the CSP with better performance has more computational power and performs outsourced decryption faster.



(a) 1-core 2G ECS

(b) 4-core 16G ECS

**Fig. 7.** Total computational time of decryption

In summary, the proposed algorithm achieves both the outsourced decryption function and the verifiable function of the mobile user for the partially decrypted ciphertext returned by the CSP. Moreover, the proposed algorithm can decrypt the ciphertext without revealing it to the CSP. Also, it can be seen that the decryption computation overhead at the mobile user side is reduced by outsourcing the complex decryption computation to the CSP.

## 7. Conclusion

To address the problem that mobile devices with limited computational power take too long to decrypt the shared ciphertext, an outsourcing decryption scheme of verifiable transformed ciphertext is proposed to reduce the computational overhead of the user's decryption. In this scheme, users with limited computing power can generate the authorized key by executing the authorized key generation algorithm, and then initiate the outsourced decryption request of the transformed ciphertext to the CSP and outsource the large number of bilinear and exponential operations involved in the decryption process to the CSP. CSP performs the partial decryption of the transformed ciphertext and sends the partially decrypted ciphertext to the user for decryption, and at the same time, the verifiable random function is used to prevent the semi-trusted CSP from returning the random outsourced decrypted ciphertext directly without performing the outsourced decryption to achieve the verifiability of the outsourced decryption. In the future, we will add a comparison of the amount of computation reduced by partial decryption.

## References

[1] Jingwei Li, Chunfu Jia, Jin Li, and Xiaofeng Chen, "Outsourcing encryption of attribute-based encryption with mapreduce," in *Proc. of International Conference on Information and Communications Security*, pp. 191-201, October 29-31, 2012. Article (CrossRef Link)

[2] Zhibin Zhou, and Dijiang Huang, "Efficient and secure data storage operations for mobile cloud computing," in *Proc. of 2012 8th international conference on network and service management (cnsm) and 2012 workshop on systems virtualiztion management (svm)*, pp. 37-45, October 22-26 2012. Article (CrossRef Link)

[3] Ti Wang, Hui Ma, Yongbin Zhou, Rui Zhang, and Zishuai Song, "Fully accountable data sharing for pay-as-you-go cloud scenes," *IEEE Transactions on Dependable and Secure Computing*, vol.18, no.4, pp. 2005-2016, 1 July-Aug. 2021. Article (CrossRef Link)

[4] Chunpeng Ge, Willy Susilo, Joonsang Baek, Zhe Liu, Jinyue Xia, and Liming Fang, "A Verifiable and Fair Attribute-Based Proxy Re-Encryption Scheme for Data Sharing in Clouds," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 5, pp. 2907-2919, 1 Sept.-Oct. 2022. Article (CrossRef Link)

[5] Shamir Adi, "Identity-based cryptosystems and signature schemes," *Advances in Cryptology: Proceedings of CRYPTO 84 4*, pp. 47-53, 1985. Article (CrossRef Link)

[6] Green Matthew, Susan Hohenberger, and Brent Waters, "Outsourcing the Decryption of ABE Ciphertexts," in *Proc. of 20th USENIX Security Symposium*, August 10-12, 2011. Article (CrossRef Link)

[7] Junzuo Lai, Robert H. Deng, Chaowen Guan, and Jian Weng, "Attribute-Based Encryption with Verifiable Outsourced Decryption," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 8, pp. 1343-1354, Aug. 2013. Article (CrossRef Link)

[8] Hui Ma, Rui Zhang, Zhiguo Wan, Yao Lu, and Suqing Lin, "Verifiable and Exculpable Outsourced Attribute-Based Encryption for Access Control in Cloud Computing," *IEEE Transactions on Dependable and Secure Computing*, vol. 14, no. 6, pp. 679-692, Nov.-Dec. 2017. Article (CrossRef Link)

[9] Suqing Lin, Rui Zhang, Hui Ma, and Mingsheng Wang, "Revisiting Attribute-Based Encryption with Verifiable Outsourced Decryption," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 10, pp. 2119-2130, Oct. 2015. Article (CrossRef Link)

[10] Peng Zhang, Zehong Chen, Joseph K. Liu, Kaitai Liang, and Hongwei Liu, “An efficient access control scheme with outsourcing capability and attribute update for fog computing,” *Future Generation Computer Systems*, vol.78, no. 2, pp. 753-762, January, 2018. Article (CrossRef Link)

[11] Zechao Liu, Zoe L. Jiang, Xuan Wang, and S.M. Yiu, "Practical attribute-based encryption: Outsourcing decryption, attribute revocation and policy updating," *Journal of Network and Computer Applications*, vol.108, pp. 112-123, 15 April 2018. Article (CrossRef Link)

[12] Chaosheng Feng, Keping Yu, Moayad Aloqaily, Mamoun Alazab, Zhihan Lv, and Shahid Mumtaz, "Attribute-Based Encryption with Parallel Outsourced Decryption for Edge Intelligent IoV," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 11, pp. 13784-13795, Nov. 2020. Article (CrossRef Link)

[13] Jiguo Li, Yao Wang, Yichen Zhang, and Jinguang Han, "Full Verifiability for Outsourced Decryption in Attribute Based Encryption," *IEEE Transactions on Services Computing*, vol. 13, no. 3, pp. 478-487, 1 May-June 2020. Article (CrossRef Link)

[14] Zhiyuan Zhao, Jianhua Wang, Kaiyong Xu, and Songhui Guo, “Fully Outsourced Attribute-Based Encryption with Verifiability for Cloud Storage,” *Journal of Computer Research and Development*, vol. 56, no. 2, pp. 442-452, February, 2019. Article (CrossRef Link)

[15] Qi Li, Hongbo Zhu, Zuobin Ying, and Tao Zhang, "Traceable Ciphertext-Policy Attribute-Based Encryption with Verifiable Outsourced Decryption in eHealth Cloud," *Wireless Communications and Mobile Computing*, Jun, 2018. Article (CrossRef Link)

[16] Yongjian Liao, Ganglin Zhang, and Hongjie Chen, "Cost-Efficient Outsourced Decryption of Attribute-Based Encryption Schemes for Both Users and Cloud Server in Green Cloud Computing," *IEEE Access*, vol. 8, pp. 20862-20869, 2020. Article (CrossRef Link)

[17] Huaqun Wang, Debiao He, and Jinguang Han, "VOD-ADAC: Anonymous Distributed Fine-Grained Access Control Protocol with Verifiable Outsourced Decryption in Public Cloud," *IEEE Transactions on Services Computing*, vol. 13, no. 3, pp. 572-583, 1 May-June 2020. Article (CrossRef Link)

[18] Jiguo Li, Fengjie Sha, Yichen Zhang, Xinyi Huang, and Jian Shen, "Verifiable Outsourced Decryption of Attribute-Based Encryption with Constant Ciphertext Length," *Security and Communication Networks*, Jan, 2017. Article (CrossRef Link)

[19] Hong Zhong, Jie Cui, Wenlong Zhu, and Yan Xu, "Efficient and Verifiable Muti-authority Attribute-Based Encryption Scheme," *Journal of Software*, vol.29, no.7, pp.2006-2017, 2018. Article (CrossRef Link)

[20] Zheng, Hui, Jun Shao, Guiyi Wei, Li Hu, Bianjing Pan, Kai Liu, and Xiaohang Mao, "Attribute-based encryption with publicly verifiable outsourced decryption," in *Proc. of International Conference on Network and System Security*, pp. 552-566, December, 2019. Article (CrossRef Link)

[21] Cui Hui, Zhiguo Wan, Xinlei Wei, Surya Nepal, and Xun Yi, "Pay as You Decrypt: Decryption Outsourcing for Functional Encryption Using Blockchain," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3227-3238, 2020. Article (CrossRef Link)

[22] Zhishuo Zhang, Wei Zhang, and Zhiguang Qin, "Fully constant-size CP-ABE with privacy-preserving outsourced decryption for lightweight devices in cloud-assisted IoT," *Security and Communication Networks*, 2021. Article (CrossRef Link)

[23] Zechao Liu, Zoe L. Jiang, Xuan Wang, Xinyi Huang, Siu-Ming Yiu, and Kunihiko Sadakane, "Offline/online attribute-based encryption with verifiable outsourced decryption," *Concurrency and Computation: Practice and Experience*, vol.29, no.7, 2017. Article (CrossRef Link)

[24] Jing Li, Zhitao Guan, Xiaojiang Du, Zijian Zhang, and Jun Wu, "An efficient encryption scheme with verifiable outsourced decryption in mobile cloud computing," in *Proc. of 2017 IEEE International Conference on Communications (ICC)*, pp. 1-6, May, 2017. Article (CrossRef Link)

[25] Yaqian Kang, and Zhenhua Liu, "A Fully Secure Verifiable and Outsourced Decryption Ranked Searchable Encryption Scheme Supporting Synonym Query," in *Proc. of 2017 IEEE Second International Conference on Data Science in Cyberspace (DSC)*, pp. 223-231, Jun, 2017. Article (CrossRef Link)

[26] Zhidan Li, Wenmin Li, Zhengping Jin, Hua Zhang, and Qiaoyan Wen, "An Efficient ABE Scheme with Verifiable Outsourced Encryption and Decryption," *IEEE Access*, vol. 7, pp. 29023-29037, 2019. Article (CrossRef Link)

[27] Zhijie Wang, Dijiang Huang, Yan Zhu, Bing Li, and Chun-Jen Chung, "Efficient Attribute-Based Comparable Data Access Control," *IEEE Transactions on Computers*, vol. 64, no. 12, pp. 3430-3443, 1 Dec. 2015. Article (CrossRef Link)

[28] Matthew Green and Giuseppe Ateniese, "Identity-based proxy re-encryption," in *Proc. of International conference on Applied Cryptography and Network Security*, pp. 288-306, June 5-8, 2007. Article (CrossRef Link).

[29] Chunpeng Ge, Zhe Liu, Jinyue Xia, Liming Fang, "Revocable identity-based broadcast proxy re-encryption for data sharing in clouds," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 3, pp. 1214-1226, 2021. Article (CrossRef Link).

[30] Peng Xu, Jiao Tengfei, Qianhong Wu, Wei Wang, Hai Jin, "Conditional identity-based broadcast proxy re-encryption and its application to cloud email, " *IEEE Transactions on Computers*, vol. 65, no.1, pp. 66-79, 2016. Article (CrossRef Link).

[31] Won-Bin Kim, Su-Hyun Kim, Daehee Seo, and Im-Yeong Lee, "Broadcast proxy reencryption based on certificateless public key cryptography for secure data sharing," *Wireless Communications and Mobile Computing*, vol. 2021, pp. 1-16, 2021. Article (CrossRef Link)

[32] Boneh Dan, and Xavier Boyen, "Efficient selective-ID secure identity-based encryption without random oracles," in *Proc. of International conference on the theory and applications of cryptographic techniques*, pp. 223-238, 2004. Article (CrossRef Link)

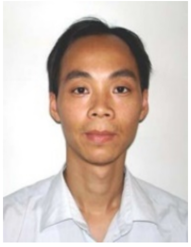
[33] Micali Silvio, Michael Rabin, and Salil Vadhan, "Verifiable random functions," in *Proc. of 40th Annual Symposium on Foundations of Computer Science*, pp. 120-130, October, 1999. Article (CrossRef Link)

[34] Bitansky Nir, "Verifiable random functions from non-interactive witness-indistinguishable proofs," *Journal of Cryptology*, vol. 33, no. 2, pp. 459-493, 2020. Article (CrossRef Link)

**Guangwei Xu** is a professor in the School of Computer Science and Technology at Donghua University, Shanghai, China. His research interests include remote data storage, data integrity verification, privacy protection in data sharing, searchable encryption, QoS and routing of the sensor network and internet of vehicles.

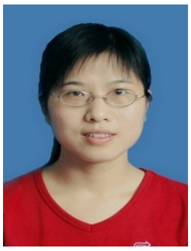
**Chen Wang** is a master candidate in the School of Computer Science and Technology at Donghua University, Shanghai, China. His main research interests include the verification of data integrity and model inference.

**Shan Li** is a master in the School of Computer Science and Technology at Donghua University, Shanghai, China. Her main research interests include the verification of data integrity, privacy protection in data sharing, and outsourced computing.



**Xiujin Shi** is an associate professor in the School of Computer Science and Technology at Donghua University, Shanghai, China. His research interests include the data service, and data security.

**Xin Luo** is an associate professor in the School of Computer Science and Technology at Donghua University, Shanghai, China. His research interests include image processing, and data security.



**Yanglan Gan** is a professor in the School of Computer Science and Technology at Donghua University, Shanghai, China. Her main research interests include the big data processing and data mining in bioinformatics.