IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received June 11, 2017, accepted July 4, 2017, date of publication July 13, 2017, date of current version July 31, 2017.

*Digital Object Identifier 10.1109/ACCESS.2017.2726584*

# Dynamic Encrypted Data Sharing Scheme Based on Conditional Proxy Broadcast Re-Encryption for Cloud Storage

**LINMEI JIANG AND DONGHUI GUO, (Senior Member, IEEE)**
School of Information Science and Engineering, Xiamen University, Xiamen 361005, China

Corresponding author: Donghui Guo (dhguo@xmu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61274133 and in part by the Science and Technology Project of Fujian Education Department of China under Grant JAT160038.

**ABSTRACT** Since Cloud Service Provider is a semi-trusted party in cloud storage, to protect data from being disclosed, users' data are encrypted before being uploaded to a cloud server. Undoubtedly, flexible encrypted data sharing is a very important demand required by cloud storage users, whereas few schemes have being designed to satisfy this demand. In this paper, based on conditional proxy broadcast re-encryption technology, an encrypted data sharing scheme for secure cloud storage is proposed. The scheme not only achieves broadcast data sharing by taking advantage of broadcast encryption, but also achieves dynamic sharing that enables adding a user to and removing a user from sharing groups dynamically without the need to change encryption public keys. Moreover, by using proxy re-encryption technology, our scheme enables the proxy (cloud server) to directly share encrypted data to the target users without the intervention of data owner while keeping data privacy, so that greatly improves the sharing performance. Meanwhile, the correctness and the security are proved; the performance is analyzed, and the experimental results are shown to verify the feasibility and the efficiency of the proposed scheme.

**INDEX TERMS** Data sharing, broadcast encryption, proxy re-encryption, pairing, access control, cloud storage.

## I. INTRODUCTION

Nowadays, the popularity of cloud storage has increased rapidly, and ordinary users as well as many large firms tend to outsource their data to CSP (Cloud Service Provider) [1]. In cloud storage environment, the CSP should provide users with controllable, cross-domain and flexible data sharing service. Whereas, since CSP is widely considered a semi-trusted party, a user always tends to upload encrypted data (ciphertexts) instead of original data (plaintexts) to cloud server for fear of data being disclosed. Thus, cloud data sharing means to share encrypted data stored in cloud servers, and it essentially involves a cryptographic access control problem. Nevertheless, traditional access control technologies which adopt access policies and privileges to control a group of users are plaintext oriented, and have weaknesses such as non-dead URL, unauthorized re-sharing, non-HTTPS shortened URL and sharing of trash files. These weaknesses widely exist in most popular cloud storage services like Dropbox, Google Drive and Microsoft SkyDrive [2]. Obviously, such technologies are not suitable for cloud storage that aimed to share data with ungrouped individuals, because the CSP can easily get the plaintexts bypass the access policies and privileges limit. Therefore, researchers are seeking novel cryptographic access control technologies to support cloud data sharing to satisfy users requirements.

Broadcast encryption which was firstly put forward by Berkovits [3] is a cryptographic access control technology being widely studied and widely used in the scenario where the data are required to be transferred from one to many, such as copyright protection of digital media, distance education, video conference and pay-TV. Broadcast encryption can be divided into two categories, one is symmetric broadcast encryption, and the other one is asymmetric broadcast encryption. The former encrypts broadcast data with symmetric encryption algorithm, e.g. Berkovits' scheme [3], Naor et al.'s scheme [4] and Halevy et al.'s scheme [5].

All these symmetric broadcast schemes are subjected to the difficulty in managing secret keys. On the contrary, the latter is also known as public key broadcast encryption, which encrypts broadcast data with the public key of an asymmetric encryption algorithm. A significant advantage of public key broadcast encryption is that encryption and decryption can be detached so that anyone knows the public encryption key can encrypt the broadcast data. The first public key broadcast encryption scheme [6] was proposed by Dodis and Fazio in 2002. However, Dodis and Fazio's scheme has the weakness of too big size of encryption keys. In addition, from perspective of cloud storage users, in order to share data to unforeseeable individuals dispersing on the Internet, the broadcast encryption technology must support dynamically adding a user to the sharing group without changing the encryption public key. For this purpose, Delerablee et al. proposed the first dynamic broadcast encryption scheme [7] that allows users join the broadcast system at any point. Thereafter, many broadcast schemes [8]–[11] are proposed in succession. On the other hand, although broadcast encryption which naturally has the "broadcast distribution" feature can easily support broadcast sharing, it is inefficient when used in a secure cloud storage platform, because data are not stored in its owner's devices but in the cloud devices held by semi-trusted CSP. If directly using broadcast encryption technology to share cloud data, the data owner must first download his encrypted data, then decrypt it, and then encrypt it with new key, and subsequently upload to cloud server for target users to download. These procedures inevitably increase the network load and lower the efficiency. For this, combining broadcast encryption and proxy re-encryption (PRE) becomes a good choice to realize broadcast sharing in secure cloud storage. Proxy re-encryption [12]–[19] enables a semi-trusted proxy to transform a ciphertext encrypted with A's public key to a ciphertext encrypted with B's public key without disclosing plaintext to the proxy. Nowadays, with the popularity of cloud computing, conditional PRE [20]–[22] that enables clients to limit the proxy by only diverting the ciphertext meeting a specified condition is put forward to improve practicability. Based on the idea of broadcast encryption and conditional proxy re-encryption, Chu *et al.* [23] first proposed the idea of CPBRE (Conditional Proxy Broadcast Re-Encryption), and put forward a CPBRE scheme. Recently, Sun et al. put forward a similar scheme [24] which attempts to deal with cloud data sharing. However, neither Chu et al.'s scheme nor Sun et al. scheme supports users taking part in or leaving the sharing group dynamically. That is to say, they are not dynamic broadcast encryption scheme. Therefore, it is not suitable for cloud storage data sharing.

In this paper, we have the following main contributions:

1) We propose an efficient encrypted data sharing scheme for secure cloud storage based on conditional proxy broadcast re-encryption. The proposed scheme (named as CPBRE-DS) not only inherits the support of user dynamics from Delerablee et al. scheme [7], but also enables the proxy directly re-encrypts sharing data in the cloud without disclosing the data to any party including the proxy.
2) We give a security analysis of the proposed scheme, which shows that it is secure against the semi-trusted CSP.
3) We analyze theoretically and test experimentally the performance of the proposed scheme, and the results show that our scheme is efficient.

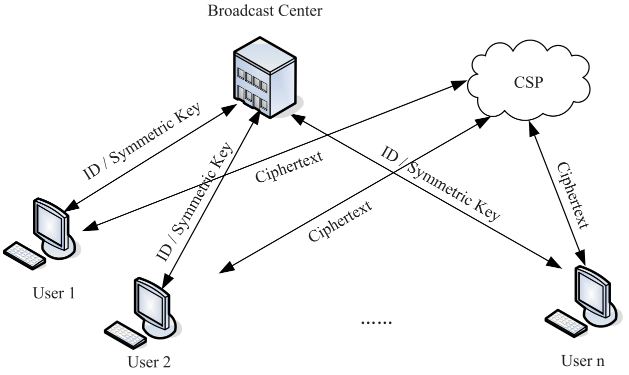

**FIGURE 1.** Network model.

The remainder of this paper is organized as follows. In section II, the network model of CPBRE-DS is described firstly. Then, in section III, the related security assumption and formal definitions of CPBRE-DS are introduced. Afterwards, in section IV, our encrypted data sharing scheme for secure cloud storage based on conditional proxy broadcast re-encryption is illustrated in detail. Thereafter, the correctness proof, security analysis and performance analysis are made in section V. Subsequently, according to the experimental results, the computation performance comparison between our scheme and Chu et al.'s scheme [23] is made in section VI. Finally, a conclusion of the paper is presented in section VII.

## II. SYSTEM MODEL

The network model of an encrypted data broadcast sharing scheme for secure cloud storage is shown in Fig. 1, which comprises 3 entities, including client, broadcast center and CSP as explained below.

1) Client: The data owner who has lots of data to store in cloud server and shares with other clients.
2) Broadcast Center: The broadcast center is not only responsible for initializing security parameters for the whole system, but also responsible for allocating secret keys and re-encryption keys for clients according to their identifiers. The secret keys and re-encryption keys generated by the broadcast center must be sent to clients through a secure communication channel.
3) CSP: The CSP provides professional data storage service and data sharing service for clients.

The interaction among the 3 entities in the network model is as follows. Firstly, a user selects and sends his ID to the

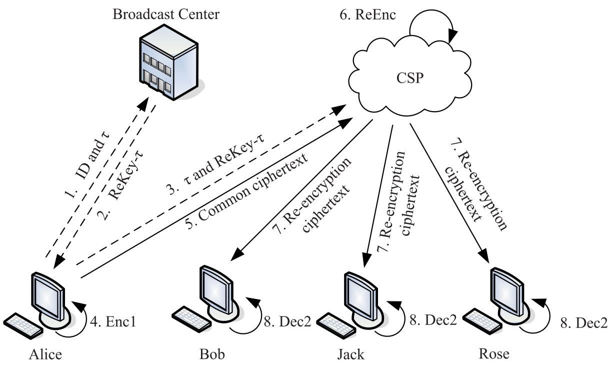

**FIGURE 2.** Alice shares data to Bob, Jack and Rose.

broadcast center where a secret key is generated for him. Afterwards, the user can encrypt his data with the secret key and a sharing condition $\tau$, which can be transported in open channel. Then, the user sends the encrypted data to the cloud servers of the CSP. Thereafter, anytime when the user wants to share the encrypted data, he can require a re-encryption key from the broadcast center according to the sharing condition $\tau$, and then sends the re-encryption key to the cloud server through a secure channel. Then, the cloud server can re-encrypt the user's data with the re-encryption key for target users to download. Lastly, the target users download the re-encrypted data from the cloud server and decrypt it with their own secret keys. In addition, to share the other data encrypted under the same sharing condition $\tau$, the user doesn't need to send a re-encryption key to the cloud server any longer.

To demonstrate the model more clearly, on the premise that all users have got their secret key from broadcast center, Fig. 2 shows the procedure that Alice shares her data under condition $\tau$ to Bob, Rose and Tom.

There are 8 steps in this procedure. Firstly, in step 1 to 3, Alice applies a re-encryption key under condition $\tau$ (ReKey-$\tau$) from broadcast center and forwards $\tau$ and ReKey-$\tau$ to the proxy server of CSP. Then, Alice encrypts her data (Enc1) with her secret key and sends the common ciphertext to the proxy server in step 4 and 5. After that, the proxy server re-encrypts the common ciphertext to re-encryption ciphertext in step 6. At last, in step 7 and 8, the target users, i.e., Bob, Jack and Rose, can download the re-encryption ciphertext and decrypt it with their own secret keys at any time. What more, Alice can even perform step 4 and 5 before step 1, 2 and 3. That is to say, when to share her data depends entirely upon her own willing. Thereafter, if Alice wants to share other data under the same condition $\tau$, she can directly encrypt the data and uploads them to the proxy server without performing step 1~3.

## III. DEFINITION

In this section, we briefly introduce bilinear paring and the security assumption $(t, n) - GDDHE$ which are used in our scheme, and then give the definition of CPBRE-DS.

### A. BILINEAR PAIRING AND $(t, n) - GDDHE$

Let $G_1$, $G_2$ and $G_T$ be cyclic groups of prime order $q$, $g_0$ and $h_0$ be respective generators of $G_1$ and $G_2$. Meanwhile, let $0_{G_1}$, $0_{G_2}$ and $1_{G_T}$ be respective identity elements of $G_1$, $G_2$ and $G_T$. A bilinear pairing is the mapping $\hat{e} : G_1 \times G_2 \rightarrow G_T$ that satisfies the following properties [25].

- Bilinearity: For any $a, b \in Z_q^*$ and $(S, T) \in G_1 \times G_2$, the formulae $\hat{e}(aS, bT) = \hat{e}(S, T)^{ab}$ holds.
- Non-degenerate: There exists $(S, T) \in G_1 \times G_2$ that makes $\hat{e}(S, T) \neq 1_{G_T}$ holds.
- Computability: For any $(S, T) \in G_1 \times G_2$, there is an efficient algorithm to calculate $\hat{e}(S, T)$.

There exists an efficient and publicly computable isomorphic mapping $\phi : G_2 \rightarrow G_1$ that makes $\phi(h_0) \rightarrow g_0$.

Technically, the pairing $\hat{e}$ can be calculated from a modified Weil pairing or a Tate pairing [25], [26].

The security of our scheme is based on $(t, n) - GDDHE$. Let $\mathcal{S} = (q, G_1, G_2, G_T, \hat{e})$ be a bilinear map group system, $g_0$ be a generator of $G_1$ and $h_0$ be a generator of $G_2$. Meanwhile, let $\mathcal{U}$ and $\mathcal{V}$ be two random univariate polynomials defined as follows.

$$\mathcal{U}(X) = \prod_{i=1}^{t} (X + x_i) = \sum_{i=0}^{t} \mu_i X^i$$

$$\mathcal{V}(X) = \prod_{i=t+1}^{n} (X + x_i) = \sum_{i=0}^{n-t} \nu_i X^i$$

Where, $x_i \in \mathcal{Z}_q^*$ and all $x_i$ are random and pairwise distinct. The $(t, n) - GDDHE$ (General Decisional Diffie-Hellman Exponent) [7] problem is defined as follows.

*Definition 1:* $(t, n) - GDDHE$. Given $g_0$, $[\gamma]g_0, \cdots, [\gamma^{t-1}]g_0, [\gamma \cdot \mathcal{U}(\gamma)]g_0, [k \cdot \gamma \cdot \mathcal{U}(\gamma)]g_0, h_0, [\gamma]h_0, \cdots, [\gamma^n]h_0$, $[k \cdot \mathcal{V}(\gamma)]h_0$, $\hat{e}(g_0, h_0)^{\mathcal{U}^2(\gamma) \cdot \mathcal{V}(\gamma)}$ and $z \in G_T$, judge whether $z = \hat{e}(g_0, h_0)^{k \cdot \mathcal{U}(\gamma) \cdot \mathcal{V}(\gamma)}$ holds or not.

### B. CONDITIONAL PROXY BROADCAST RE-ENCRYPTION FOR DATA SHARING

The new notion of our conditional proxy broadcast re-encryption for data sharing is defined as follows.

① $Setup(\kappa) \rightarrow PubParams$: For setting up the system parameters. Input security parameter $\kappa$, output system master key $msk$ and public parameters $PubParams$. This algorithm is executed when system is initialized, and the $PubParams$ will be published when it is finished.

② $KeyGen(PubParams, msk, ID_i) \rightarrow sk_i$: For generating the secret key of target user $ID_i$. Input public parameters $PubParams$, master key $msk$ and target user's identifier $ID_i$, output the secret key $sk_i$ of the target user $ID_i$. The broadcast center is responsible for executing this algorithm and sending the secret key $sk_i$ to the target user $ID_i$ through secure channel after it is finished.

③ $Enc(PubParams, \tau, M) \rightarrow Ct$: For encrypting plaintext $M$ to common ciphertext $Ct$ under condition $\tau$. Input public parameters $PubParams$, condition $\tau$ and plaintext $M$, output ciphertext $Ct$ of $M$. Everyone gets the public parameters

*PubParams* may execute this algorithm and send $\tau$ and the ciphertext *Ct* to CSP through an open channel.

④ $RkGen(PubParams, msk, \tau) \rightarrow rk_\tau$: For generating the re-encryption key under condition $\tau$. Input public parameters *PubParams*, master key *msk* and condition $\tau$, output re-encryption key $rk_\tau$. The broadcast center is responsible for executing this algorithm and guarantee that different users have different condition $\tau$. At last, the broadcast center sends the re-encryption key $rk_\tau$ to the applier through secure channel, and the applier may forward condition $\tau$ and $rk_\tau$ to the proxy via secure channel at any time.

⑤ $ReEnc(PubParams, rk_\tau, \tau, Ct) \rightarrow Cr$: For converting the common ciphertext *Ct* to re-encrypted ciphertext *Cr*. Input public parameters *PubParams*, re-encryption key $rk_\tau$, condition $\tau$ and common ciphertext *Ct*, output re-encrypted ciphertext *Cr* when it succeeds and $m = \perp$ when it fails. The proxy is responsible for executing this algorithm, and sending $\tau$ and the ciphertext *Cr* to target users through open channel.

⑥ $Dec1(PubParams, msk, \tau, Ct) \rightarrow m$: For decrypting common ciphertext *Ct*. Input public parameters *PubParams*, master key *msk*, condition $\tau$ and common ciphertext *Ct*, output the plaintext $m = M$ when it succeeds and $m = \perp$ when it fails.

⑦ $Dec2(PubParams, sk_i, \tau, Cr) \rightarrow m$: For decrypting re-encrypted ciphertext *Cr*. Input public parameters *PubParams*, the secret key $sk_i$ of target user $ID_i$, condition $\tau$ and re-encrypted ciphertext *Cr*, output the plaintext $m = M$ when it succeeds and $m = \perp$ when it fails.

In addition, for any given target user $ID_i$, any condition $\tau$, and any plaintext *M*, the algorithms above must meet the following correctness constraints.

$$Dec1(PubParams, msk, \tau, Enc(PubParams, \tau, M)) = M \tag{1}$$

$$Dec2(PubParams, sk_i, \tau, ReEnc(PubParams, rk_\tau, \tau, Enc(PubParams, \tau, M))) = M \tag{2}$$

*Definition 2:* A scheme which is composed of the above 7 algorithms and the corresponding constraints formula (1) and formula (2) is called a CPBRE-DS scheme.

The resisting adaptive-chosen-plaintext attack security of a CPBRE-DS scheme can be defined by a game between challenger *C* and adversary *A* as below:

① *Init*: Adversary *A* freely selects target users set $S^*$ and condition $\tau^*$ to attack.

② *Setup*: Challenger *C* executes algorithm $Setup(\kappa)$, then output and publish system public parameters *PubParams*.

③ *Phase*1: Adversary *A* may adaptively repeat the following queries:

$Extract(ID_i)$: Adversary *A* sends an identifier $ID_i$ to challenger *C*, then *C* executes algorithm $KeyGen(PubParams, msk, ID_i)$ to generate the secret key $sk_i$ of $ID_i$ and sends $sk_i$ back to *A*.

$RkExtract(\tau)$: Adversary *A* sends a condition $\tau$ to challenger *C*, then *C* executes algorithm $RkGen(PubParams, msk, \tau)$ to generate re-encryption key $rk_\tau$ and sends $rk_\tau$ back to *A*.

④ *Challenge*: Adversary *A* output two plaintexts $M_0$ and $M_1$, which have the same length. Then, challenger *C* flips a coin $\beta \in \{0, 1\}$, then executes algorithm $Enc(PubParams, \tau^*, M)$ to compute the ciphertext $Ct^*$ and sends $Ct^*$ to adversary *A*.

⑤ *Phase*2: Adversary *A* may make the same queries as *Phase*1 except $Extract(ID^* \in S^*)$ and $RkExtract(\tau^*)$.

⑥ *Guess*: Adversary *A* returns a guess $\beta' \in \{0, 1\}$ of $\beta$.

The game defined above is called an IND-sCond-CPA game. The adversary *A* in IND-sCond-CPA game is called an IND-sCond-CPA adversary, and the advantage of which in winning the game in a CPBRE-DS scheme is defined as:

$$Adv_A^{IND-sCond-CPA} = |Pr[\beta' = \beta] - \tfrac{1}{2}| \tag{3}$$

*Definition 3:* A CPBRE-DS scheme is said IND-sCond-CPA secure, if the advantages $Adv_A^{IND-sCond-CPA}$ for all polynomial time adversaries *A* in the IND-sCond-CPA game are negligible.

## IV. PROPOSED SCHEME

According to the definition described in *subsection B* of section III, we construct our bilinear pairing and $(t, n) - GDDHE$ based CPBRE-DS scheme as below:

① $Setup(\kappa) \rightarrow PubParams$: The system manager selects a positive integer security parameter $\kappa$ as the input and builds a bilinear pairing system $\mathcal{U} = (q, G_1, G_2, G_T, \hat{e})$, where $|q| = \kappa$ and $\hat{e} : G_1 \times G_2 \rightarrow G_T$. Let $\mathcal{Z}_q^*$ be a multiplicative group modulo *q*, that is the set $\{1, 2, \cdots, q-1\}$. Select a generator $g \in G_1$ and a generator $h \in G_2$; meanwhile, select appropriate cryptographic hash function $H : \{0, 1\}^* \rightarrow Z_q^*$, permutation function $\varphi : G_T \rightarrow \{0, 1\}^{2\kappa}$, $\pi : G_1 \rightarrow \{0, 1\}^{2\kappa}$, and $\psi : \{0, 1\}^* \rightarrow G_1$, where $\forall r \in G_1$, there is $\psi(\pi(r)) = r$. Thereafter, system manager selects a random big integer $\lambda \in \mathcal{Z}_q^*$, and make $msk = \lambda$ the system master secret key, then calculates $P_{pub} = \lambda \cdot g$. The public parameters are:

$$PubParams = \{q, G_1, G_2, G_T, \hat{e}, g, h, P_{pub}, \hat{e}(g, h), H, \varphi, \pi, \psi\}$$

② $KeyGen(PubParams, msk, ID_i) \rightarrow sk_i$: The broadcast center computes:

$$A_i = \frac{H(ID_i)}{\lambda + H(ID_i)} \cdot g, B_i = \frac{1}{\lambda + H(ID_i)} \cdot h$$

The secret key of target user $ID_i$ is $sk_i = (A_i, B_i)$. The broadcast center sends $sk_i$ to the target user $ID_i$ through secure channel.

③ $Enc(PubParams, \tau, M) \rightarrow Ct$: Suppose the identifier set of target users is $R = \{ID_1, \cdots, ID_r\}$, to encrypt plaintext *M* to common ciphertext *Ct* under condition $\tau$, one must first apply $P_i$ from broadcast center, where $i \in \{1, 2, \cdots, r\}$. Here, $P_i$ is also considered a public key of broadcast center because all legitimate clients may get it, and it can be computed in two steps as below:

$$\theta_i = \frac{1}{\prod_{j=1}^{i}(\lambda + H(ID_j))},$$
$$P_i = \theta_i \cdot h.$$

Then, select two random big integers $s, \alpha \in Z_q^*$ and calculate the following formulas.

$$\begin{aligned} c_1 &= \tau, \\ c_2 &= s \cdot P_{pub} + (-s\tau) \cdot g, \\ c_3 &= M \oplus \varphi(\hat{e}(g, P_r)^s), \\ c_4 &= \pi(s \cdot P_{pub}) \oplus \varphi(\hat{e}(g, h)^\alpha), \\ c_5 &= \alpha \cdot P_{pub} + (\alpha\tau) \cdot g, \\ c_6 &= s \cdot P_r. \end{aligned}$$

The common ciphertext is $Ct = (c_1, c_2, c_3, c_4, c_5, c_6)$. Lastly, $\tau$ and $Ct$ are sent to the proxy through open channel.

④ $RkGen(PubParams, msk, \tau) \rightarrow rk_\tau$: The broadcast center computes re-encryption key under condition $\tau$ as follows:

$$rk_\tau = \frac{1}{\lambda + \tau} \cdot h$$

Then, the broadcast center sends $\tau$ and $rk_\tau$ to the applier and the applier may forward them to the proxy through secure channel at any time.

⑤ $ReEnc(PubParams, rk_\tau, \tau, Ct) \rightarrow Cr$: To convert common ciphertext $Ct$ to re-encrypted ciphertext $Cr$, after getting common ciphertext $Ct$ and re-encryption key $rk_\tau$, the proxy first check whether $c_1$ is equal to $\tau$, if not, it returns error symbol $\perp$ and exit; otherwise it continues to compute the following formulas.

$$\begin{aligned} c_4' &= \psi(c_4 \oplus \varphi(\hat{e}(c_5, rk_\tau))) \\ &= \psi(c_4 \oplus \varphi(\hat{e}(\alpha \cdot P_{pub} + \alpha\tau \cdot g, \frac{1}{\lambda + \tau} \cdot h))) \\ &= \psi(c_4 \oplus \varphi(\hat{e}(\alpha(\lambda + \tau) \cdot g, \frac{1}{\lambda + \tau} \cdot h))) \\ &= \psi(c_4 \oplus \varphi(\hat{e}(g, h)^{\alpha(\lambda+\tau)\cdot\frac{1}{\lambda+\tau}})) \\ &= \psi(\pi(s \cdot P_{pub}) \oplus \varphi(\hat{e}(g, h)^\alpha) \oplus \varphi(\hat{e}(g, h)^\alpha)) \\ &= \psi(\pi(s \cdot P_{pub})) \\ &= s \cdot P_{pub} \end{aligned}$$

The re-encrypted ciphertext is $Cr = (c_1, c_2, c_3, c_4', c_6)$, and the proxy will send $\tau$ and $Cr$ to target users through open channel.

⑥ $Dec1(PubParams, msk, \tau, Ct) \rightarrow m$: On obtaining the common ciphertext $Ct$, one possesses the master key $msk$ and knows the condition $\tau$, i.e. the broadcast center, can decrypt the common ciphertext by calculating the following formula:

$$M = c_3 \oplus \varphi(\hat{e}(c_2, P_r)^{1/(\lambda - \tau)})$$

Where, $P_r$ is calculated from $P_i = \theta_i \cdot h$ (see ③). If decryption succeeds the plaintext $m = M$ is returned, else an error symbol $\perp$ is returned.

⑦ $Dec2(PubParams, sk_i, \tau, Cr) \rightarrow m$: On obtaining the re-encrypted ciphertext $Cr$, one owns the secret key $sk_i$ (the target user) and knows the condition $\tau$ can decrypt the re-encrypted ciphertext by calculating the following formula:

$$\begin{aligned} B_{i,R} &= \frac{1}{\prod_{j=1}^{r}(\lambda + H(ID_j))} \cdot B_i \\ &= \frac{1}{(\lambda + H(ID_i))\prod_{j=1}^{r}(\lambda + H(ID_j))} \cdot h \\ M &= c_3 \oplus \varphi(\hat{e}(c_4', B_{i,R}) \cdot \hat{e}(A_i, c_6)) \end{aligned}$$

If decryption succeeds, the plaintext $m = M$ is returned, else an error symbol $\perp$ is returned. Here, $ID_i \neq ID_j$, and a key point is that $B_{i,R}$ could be figured out indirectly through $B_i$ and $\{ID_i, P_i\}$ without system master key $\lambda$, where $P_i$ can be applied from broadcast center. The computational complex to calculate $B_{i,R}$ is only O($r$). For detailed computation process, please refer to the Aggregate' algorithm in Delerablee et al.'s paper [7].

## V. ANALYSIS

In this section, we first prove the correctness of our CPBRE-DS scheme according to the definition 2. Then, we prove its CPA security based on $(t, n) - GDDHE$ assumption. Lastly, we analyze the security and performance of the proposed scheme theoretically through comparing with Chu et al.'s CPBRE scheme [23].

### A. CORRECTNESS PROOF

According to definition 2, a CPBRE-DS scheme must meet two correctness constraint formulas (1) and (2). So, in this section, we'll prove that the two formulas hold for the proposed CPBRE-DS scheme.

*Theorem 1:* In the proposed CPBRE-DS scheme, if all the algorithms are executed strictly and correctly, then formula (1) and (2) must hold.

*Proof:* Firstly, we prove the correctness of formula (1). Apparently, after executing $Enc(PubParams, \tau, M)$, the common ciphertext is $Ct = (c_1, c_2, c_3, c_4, c_5, c_6)$. Then, we can derive formula (1) as follows.

$$\begin{aligned} &Dec1(PubParams, msk, \tau, Enc(PubParams, \tau, M)) \\ &\quad = c_3 \oplus \varphi(\hat{e}(c_2, P_r)^{1/(\lambda-\tau)}) \\ &\quad = c_3 \oplus \varphi(\hat{e}(c_2, h)^{\theta_r/(\lambda-\tau)}) \\ &\quad = c_3 \oplus \varphi(\hat{e}(s \cdot P_{pub} + (-s\tau) \cdot g, h)^{\theta_r/(\lambda-\tau)}) \\ &\quad = c_3 \oplus \varphi(\hat{e}(s\lambda \cdot g + (-s\tau) \cdot g, h)^{\theta_r/(\lambda-\tau)}) \\ &\quad = c_3 \oplus \varphi(\hat{e}(s(\lambda - \tau) \cdot g, h)^{\theta_r/(\lambda-\tau)}) \\ &\quad = c_3 \oplus \varphi(\hat{e}(g, h)^{s\theta_r}) \\ &\quad = M \oplus \varphi(\hat{e}(g, P_r)^s) \oplus \varphi(\hat{e}(g, h)^{s\theta_r}) \\ &\quad = M \oplus \varphi(\hat{e}(g, h)^{s\theta_r}) \oplus \varphi(\hat{e}(g, h)^{s\theta_r}) \\ &\quad = M \end{aligned}$$

Therefore, formula (1) holds. Then, we prove the correctness of formula (2). Apparently, after executing $ReEnc(PubParams, rk_\tau, \tau, Ct)$, the re-encrypted ciphertext is $Cr = (c_1, c_2, c_3, c_4', c_6)$, that is, we have:

$$\begin{aligned} &ReEnc(PubParams, rk_\tau, Enc(PubParams, \tau, M)) \\ &\qquad = (c_1, c_2, c_3, c_4', c_6) \end{aligned}$$

So, we can derive formula (2) as follows.

$$\begin{aligned} &Dec2(PubParams, sk_i, \tau, \\ &\qquad ReEnc(PubParams, rk_\tau, \tau, \\ &\qquad\qquad Enc(PubParams, \tau, M))) \\ &= Dec2(PubParams, sk_i, \tau, (c_1, c_2, c_3, c_4', c_6)) \\ &= c_3 \oplus \varphi(\hat{e}(c_4', B_{i,R}) \cdot \hat{e}(A_i, c_6)) \\ &= c_3 \oplus \varphi(\hat{e}(s \cdot P_{pub}, \frac{1}{(\lambda + H(ID_i)) \prod\limits_{j=1}^{r} (\lambda + H(ID_j))} \cdot h) \\ &\qquad \cdot \hat{e}(\frac{H(ID_j)}{\lambda + H(ID_j)} \cdot g, s \cdot P_r)) \\ &= c_3 \oplus \varphi(\hat{e}(s \cdot P_{pub}, \frac{1}{(\lambda + H(ID_i)) \prod\limits_{j=1}^{r} (\lambda + H(ID_j))} \cdot h) \\ &\qquad \cdot \hat{e}(\frac{H(ID_j)}{\lambda + H(ID_j)} \cdot g, \frac{s}{\prod\limits_{j=1}^{r} (\lambda + H(ID_j))} \cdot h)) \\ &= c_3 \oplus \varphi(\hat{e}(g, h)^{\frac{s\lambda}{(\lambda+H(ID_i)) \prod_{j=1}^{r} (\lambda+H(ID_j))}} \\ &\qquad \cdot \hat{e}(g, h)^{\frac{s \cdot H(ID_j)}{(\lambda+H(ID_j)) \prod_{j=1}^{r} (\lambda+H(ID_j))}}) \\ &= c_3 \oplus \varphi(\hat{e}(g, h)^{\frac{s\lambda}{(\lambda+H(ID_i)) \prod_{j=1}^{r} (\lambda+H(ID_j))} + \frac{s \cdot H(ID_j)}{(\lambda+H(ID_j)) \prod_{j=1}^{r} (\lambda+H(ID_j))}}) \\ &= c_3 \oplus \varphi(\hat{e}(g, h)^{\frac{s}{\prod_{j=1}^{r} (\lambda+H(ID_j))}}) \\ &= c_3 \oplus \varphi(\hat{e}(g, h)^{s\theta_r}) \\ &= M \oplus \varphi(\hat{e}(g, P_r)^s) \oplus \varphi(\hat{e}(g, h)^{s\theta_r}) \\ &= M \oplus \varphi(\hat{e}(g, h)^{s\theta_r}) \oplus \varphi(\hat{e}(g, h)^{s\theta_r}) \\ &= M \end{aligned}$$

Therefore, formula (2) holds.

In short, since both formula (1) and formula (2) hold, theorem 1 is proved.

### B. SECURITY ANALYSIS

*Theorem 2:* If the $(t, n) - GDDHE$ problem is difficult, then CPBRE-DS scheme is IND-sCond-CPA secure.

*Proof:* Suppose there exists an adversary $A$ breaking the proposed CPBRE-DS scheme to decrypt re-encrypted ciphertext with non-negligible advantage, then with $A$, we can construct an algorithm $B$ to break $(t, n) - GDDHE$ problem within polynomial time.

Firstly, we build the bilinear pairing system $\mathcal{S} = (q, G_1, G_2, G_T, \hat{e})$ for algorithm $B$, where $g_0 \in G_1$ and $h_0 \in G_2$ are generators. Let $\mathcal{U}(x) = \prod\limits_{i=1}^{t} (x + h(ID_i))$ and $\mathcal{V}(x) = \prod\limits_{i=1}^{n-t} (x + h(ID_i))$ be two random polynomials of respective degree $t$ and $n - t$ with non-zero pairwise distinct roots. Given any instance of $(t, n) - GDDHE$ problem: $g_0$, $\lambda \cdot g_0$, $\cdots, \lambda^{t-1} \cdot g_0, \lambda \cdot \mathcal{U}(\lambda) \cdot g_0, s\lambda \cdot \mathcal{U}(\lambda) \cdot g_0, h_0, \lambda \cdot h_0, \cdots, \lambda^n \cdot h_0$, $s \cdot \mathcal{V}(\lambda) \cdot h_0$, $\hat{e}(g_0, h_0)^{\mathcal{U}^2(\lambda) \cdot \mathcal{V}(\lambda)}$ and $z \in G_T$, the objective of algorithm $B$ is to correctly output 0 if $z = \hat{e}(g_0, h_0)^{s \cdot \mathcal{U}(\lambda) \cdot \mathcal{V}(\lambda)}$ and output 1 otherwise. Algorithm $B$ takes $A$ as its subroutine and simulates the IND-sCond-CPA game as follows.

① *Init*: Algorithm $B$ obtains identifier set $S^* = \{ID_1^*, ID_2^*, \cdots, ID_t^*\}$ and condition $\tau^*$ to be challenged from adversary $A$.

② *Setup*: Let $g = f(\lambda) \cdot g_0$, algorithm $B$ computes $h = \mathcal{U}(\lambda) \cdot \mathcal{V}(\lambda) \cdot h_0$, $P_{pub} = \lambda \cdot g = \lambda \cdot \mathcal{U}(\lambda) \cdot g_0$ and $\hat{e}(g, h) = \hat{e}(g_0, h_0)^{\mathcal{U}^2(\lambda) \cdot \mathcal{V}(\lambda)}$, then sends public parameters $PubParams = \{q, G_1, G_2, G_T, \hat{e}, g, h, P_{pub}, \hat{e}(g, h), H, \varphi, \pi, \psi\}$ to adversary $A$.

③ *Phase*1: Algorithm $B$ answers adversary $A$ the following queries.

*Extract*$(ID_i)$: Adversary $A$ sends an identifier $ID_i$ to algorithm $B$, there may be two situations:

1) $ID_i \notin S^*$: Algorithm $B$ executes key generating algorithm $KeyGen(PubParams, msk, ID_i)$ to generate user $ID_i$'s secret key $sk_i$ and returns $sk_i$ to adversary $A$.
2) $ID_i \in S^*$: Let $\mathcal{U}_i(x) = \mathcal{U}(x)/(x + H(ID_i))$, where $i \in \{1, 2, \cdots, t)$. Then, algorithm $B$ computes $\tilde{A}_i = H(ID_i) \cdot \mathcal{U}_i(\lambda) \cdot g_0 = \frac{H(ID_i)}{\lambda+H(ID_i)} \cdot g$ and $\tilde{B}_i = \mathcal{U}_i(\lambda) \cdot \mathcal{V}(\lambda) \cdot h_0 = \frac{1}{\lambda+H(ID_i)} \cdot h$, the secret key of $ID_i$ is $s\tilde{k}_i = (\tilde{A}_i, \tilde{B}_i)$, and return $s\tilde{k}_i$ to adversary $A$.

*RkExtract*$(\tau)$: Adversary $A$ sends condition $\tau$ to algorithm $B$, there are also two situations:

1) $\tau \neq \tau^*$: $B$ executes re-encryption key generating algorithm $RkGen(PubParams, msk, \tau)$ to generate $rk_\tau$ and returns $rk_\tau$ to adversary $A$.
2) $\tau = \tau^*$: $B$ output 0 or 1 randomly and terminates the game.

④ *Challenge*: Adversary $A$ output two plaintext $M_0$ and $M_1$ with the same length. Algorithm $B$ flip a coin to get a random bit $\beta \in \{0, 1\}$, then executes algorithm $ReEnc(PubParams, rk_{\tau^*}, Enc(PubParams, \tau^*, M_\beta))$ to get the legitimate ciphertext $Cr^* = (c_1^*, c_2^*, c_3^*, {c_4'}^*, c_6^*)$ which can be decrypted correctly and sends $Cr^*$ to adversary $A$. Thus, on the condition of the $(t, n) - GDDHE$ problem given above, each component of $Cr^*$ is as follows:

$$\begin{aligned} c_1^* &= \tau^*, \\ c_2^* &= s \cdot P_{pub} + (-s\tau^*) \cdot g, \\ c_3^* &= M \cdot \hat{e}(g, h)^{-s\theta_r}, \\ {c_4'}^* &= s \cdot P_{pub}, \\ c_6^* &= s \cdot P_r. \end{aligned}$$

Thereafter, by feeding $g = \mathcal{U}(\lambda) \cdot g_0$ and $h = \mathcal{U}(\lambda) \cdot \mathcal{V}(\lambda) \cdot h_0$ to the formulas above and invoking the decryption algorithm:

$$\begin{aligned} &Dec2(PubParams, sk_i, \tau^*, \\ &\qquad ReEnc(PubParams, rk_{\tau^*}, \tau^*, \\ &\qquad\qquad Enc(PubParams, \tau^*, M))) \\ &= c_3^* \cdot \hat{e}(c_4'^*, \tilde{B}_{i,R}) \cdot \hat{e}(\tilde{A}_i, c_6^*) \end{aligned}$$

Then, the algorithm $B$ gets $M_\beta = c_3 \cdot \hat{e}(g,h)^{\frac{s}{\prod_{j=1}^{r}(\lambda+H(ID_j))}} = c_3 \cdot \hat{e}(g_0,h_0)^{\frac{s \cdot \mathcal{U}^2(\lambda)\cdot\mathcal{V}(\lambda)}{\mathcal{U}(\lambda)}} = c_3 \cdot \hat{e}(g_0,h_0)^{s\cdot\mathcal{U}(\lambda)\cdot\mathcal{V}(\lambda)}$

⑤ *Phase*2: Adversary $A$ may make the same key extraction queries as *Phase 1*.

⑥ *Guess*: Adversary $A$ outputs a guess $\beta'$ of $\beta$. If $\beta' = \beta$, adversary $A$ wins the game.

If adversary $A$ can win the game above, then it indicates that algorithm $B$ can solve $(t, n) - GDDHE$ problem within polynomial time which is in conflict with the premise that $(t, n) - GDDHE$ problem is cryptographically difficult. Therefore, theorem 2 is proved.

### C. PERFORMANCE ANALYSIS

The space costs and computational costs of the proposed CPBRE-DS scheme will be analyzed via comparing with Chu et al.'s CPBRE scheme [23] and Sun et al.'s PBRE scheme [24] in this section.

**TABLE 1. Space costs comparison.**

| Key or Ciphertext | CPBRE-DS | CPBRE [23] | PBRE [24] |
|---|---|---|---|
| Key | 2 | 1 | 1 |
| Common ciphertext | 5 | 4 | 6 |
| Re-enc key | 1 | 5 | 7 |
| Re-enc ciphertext | 4 | 6 | 9 |
| Sum | 12 | 16 | 23 |

We first discuss the space costs. The group elements numbers, which determine the space occupied, of the proposed CPBRE-DS scheme, Chu et al.'s CPBRE scheme and Sun et al.'s PBRE scheme are listed in Table I. It can be seen from the table that the key size and common ciphertext of CPBRE-DS scheme are one element more than that of CPBRE scheme respectively. However, the re-encryption key size of CPBRE-DS scheme is four elements less than that of CPBRE scheme, and the re-encrypted ciphertext length of CPBRE-DS scheme is 2 elements less than that of CPBRE scheme. Overall, the space costs of the proposed CPBRE-DS scheme are less than that of Chu et al.'s CPBRE scheme, and much less than that of Sun et al. PBRE scheme.

Now, we discuss the computational costs. For convenience, the following symbols are defined: $t_a$ and $t_m$ are ECC point addition and point multiplication respectively, $t_b$ is bilinear pairing computation. In ECC algorithms, point multiplication is more complex than point addition, bilinear pairing computation is more complex than point multiplication, that is to say, $t_a < t_m < t_b$. In addition, $n$ denotes the size of complete user set, and $r$ denotes the size of target user set that contains the users to share a file at a time.

**TABLE 2. Computational costs comparison.**

| Algorithm | CPBRE-DS | CPBRE [23] | PBRE [24] |
|---|---|---|---|
| Setup | $t_m + t_b$ | $2n \cdot t_m$ | $2n \cdot t_m$ |
| KeyGen | $2t_m$ | $t_m$ | $t_m$ |
| Enc/ Encrypt | $t_a$<br>$+(8+r)t_m$ | $(r+4)t_a$<br>$+5t_m$<br>$+t_b$ | $(r+4)t_a$<br>$+5t_m$<br>$+t_b$ |
| RkGen | $t_m$ | $(r+6)t_a$<br>$+7t_m$<br>$+t_b$ | $(r+6)t_a$<br>$+7t_m$<br>$+t_b$ |
| ReEnc | $t_b$ | $(r+1)t_a$<br>$+t_b$ | $(2r+3)t_a$<br>$+t_m$<br>$+8t_b$ |
| Dec1/<br>Decrypt-I | $t_m$<br>$+t_b$ | $(r+2)t_a$<br>$+2t_b$ | $(3r+4)t_a$<br>$+8t_m$<br>$+8t_b$ |
| Dec2/<br>Decrypt-II | $(r+1)t_a$<br>$+(r+1)t_m$<br>$+2t_b$ | $(r+3)t_a$<br>$+3t_b$ | $(3r+3)t_a$<br>$+6t_m$<br>$+8t_b$ |
| Sum | $(r+2)t_a$<br>$+(r+13)t_m$<br>$+5t_b$ | $(5r+16)t_a$<br>$+(2n+13)t_m$<br>$+8t_b$ | $(10r+20)t_a$<br>$+(2n+28)t_m$<br>$+26t_b$ |

Table II lists the computational costs comparison of the proposed CPBRE-DS scheme, Chu et al.'s CPBRE [23] scheme and Sun et al.'s PBRE scheme [24]. It can be seen from the table that, for CPBRE-DS scheme, the computational costs of all algorithm are irrelevant to the size of complete users set, and only "Enc" and "Dec2" are linearly related to the size of target users set; while, for CPBRE scheme, the computational costs of the "setup" algorithm is linearly associated with the size of complete users set, and all other algorithms except "KeyGen" are linearly correlated with the size of target users set. On the other hand, except the efficiency of "KeyGen", "Enc" and "Dec2 / Decrypt-II" of CPBRE-DS are a little lower than that of CPBRE scheme, all the other algorithms of CPBRE-DS are more efficient than those of CPBRE apparently, while all algorithms of CPBRE are more efficient than those of PBRE. As for the overall computational costs, the proposed CPBRE-DS scheme is undoubtedly better than Chu et al.'s CPBRE scheme, and much better than Sun et al.'s PBRE scheme.

## VI. EXPERIMENT AND ANALYSIS

To evaluate the feasibility and actual computational efficiency of our scheme, experiments are conducted according to the design of the proposed CPBRE-DS scheme and Chu et al.'s CPBRE scheme [23] respectively. In this section, we show the experimental results to illustrate the efficiency of the proposed scheme by comparing with Chu et al.'s CPBRE scheme.

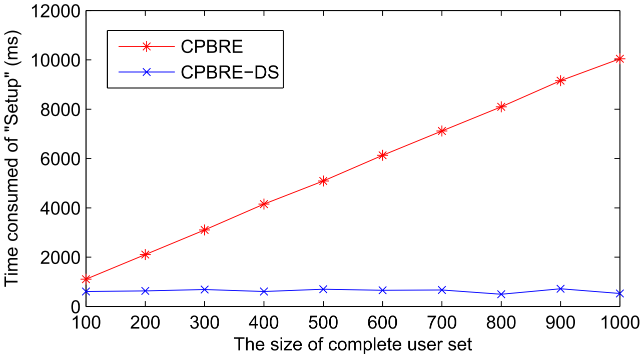

**FIGURE 3.** Time consumed of "setup".

### A. EXPERIMENTAL ENVIRONMENT

The experimental environment involves Visual Studio C++ 2012 IDE, open source library gmp 5.1.0 for big number computation, openssl 0.9.8e for cryptography algorithm and pbc 0.5.14 for bilinear pairing computation. The spec of the experimental computers is as follows: the host is a computer with Inter (R) Core (TM) i5-4590 on 3.30GHz and 16.0GB memory, and operating system installed is windows 7 ultimate (x64). To ensure fairness of the comparison, the algorithms to compute big number, to make symmetric encryption and bilinear pairing computation and the related parameters are the same for the two schemes, where 192 bits CFB AES is used for symmetric encryption and sha1 is used for hashing. In addition, what needs to be stated is, because Chu et al.'s CPBRE scheme uses symmetric bilinear pairing and the proposed scheme uses asymmetric bilinear pairing, so the bilinear pairing parameters they used are difference. That is, the former uses the "pbc" standard A parameters, and the latter uses the "pbc" standard D159 parameters instead.

### B. RESULT AND ANALYSIS

In the experimental environment described above, the following items are mainly tested in the experiments.

1) The impact of the size of complete user set on computational cost of "setup" algorithm. 2) When the complete user set is determined, the impact of the size of target user set on respective computational cost of encryption, re-encryption, common ciphertext decryption and re-encrypted ciphertext decryption. Firstly, for test item 1, four complete user sets whose sizes are nearly 10, 100, 1000 and 10000 are selected respectively, and the test result is shown in Fig. 3. Apparently, we can see from Fig. 3 that time consumed in "setup" of CPBRE-DS scheme is irrelevant to the size, whereas that of Chu et al.'s scheme is proportional to the size. Moreover, the time consumed in "setup" of our CPBRE-DS scheme is much less than that of Chu et al.'s scheme.

As for test item 2, the size of the complete user set is fixed to 1000, but the sizes of 9 target user sets are 100, 200, $\cdots$, 900 respectively, that is, the coverage percentage of target user set in complete user set changes from 10% to 90% evenly. For both schemes, time consumed of encryption, re-encryption, common ciphertext decryption and re-encrypted ciphertext decryption are tested, and the results are shown in Fig. 4 to Fig. 7.

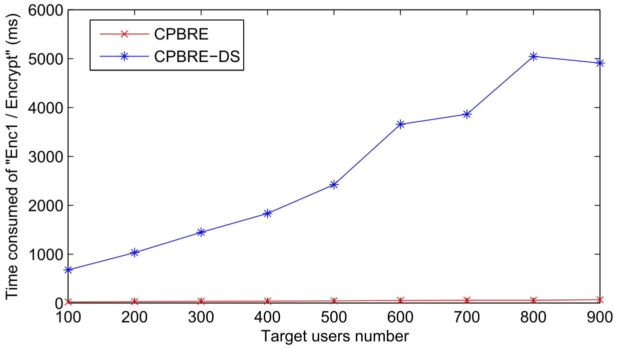

**FIGURE 4.** Time consumed of "Enc / Encryption".

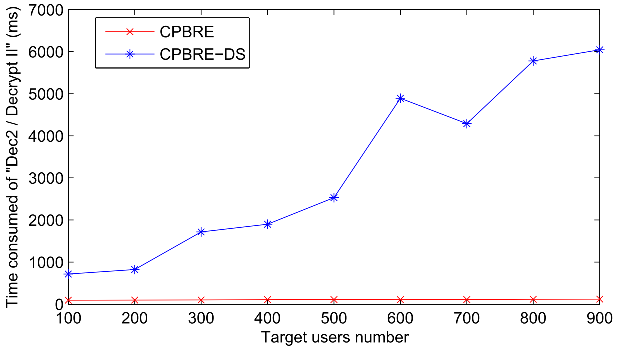

**FIGURE 5.** Time consumed of "Dec2 / Decryption II".

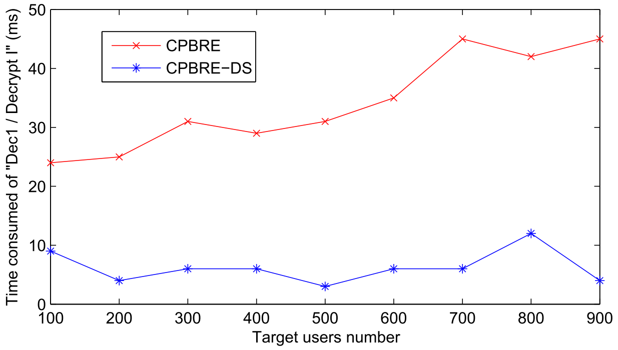

**FIGURE 6.** Time consumed of "Dec1 / Decryption I".

It can be seen from Fig. 4 and Fig. 5 that the time consumed of "Encrypt" and "Decrypt II" of Chu et al.'s CPBRE scheme are less than the counterparts of the proposed CPBRE-DS scheme. For time consumed of "Encrypt" and "Decrypt II" algorithms, considering along with the computational costs comparison listed in table 2, we can see that those of Chu et al.'s scheme are linearly related to target users number in ECC addition, but those of the proposed CPBRE-DS are linearly related to target users number in ECC multiplication. On the other hand, it can be seen from Fig. 6 and Fig. 7 that the time consumed of "Dec1" and "ReEnc" of the proposed CPBRE-DS scheme are less than the counterparts of Chu et al.'s CPBRE scheme. For time consumed of "Dec1" and "ReEnc II" algorithms, considering along with the computational costs comparison listed

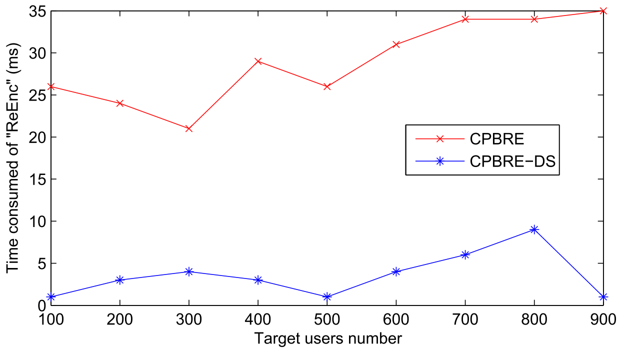

**FIGURE 7.** Relations between time consumed of "ReEnc" and target users number.

in table 2, we can see that those of Chu et al.'s scheme are linearly related to target users number in ECC addition, but those of the proposed CPBRE-DS are irrelevant to target users number. Therefore, in Fig. 6 and Fig. 7, time consumed of Chu et al.'s CPBRE scheme is up with the increasing of target users number, while time consumed of the proposed scheme is always less than 10ms. Apparently, the proposed CPBRE-DS scheme enjoys absolute advantages in the performance of "Dec1" and "ReEnc" algorithm.

In the synthesis of the above analysis, comparing the proposed CPBRE-DS scheme and Chu et al.'s CPBRE scheme, we get the following conclusion. For the "Setup" efficiency, the proposed scheme wins absolutely. For "Enc / Encrypt" and "Dec2 / Decrypt II" algorithm, Chu et al.'s scheme is more efficient; while for "ReEnc" and "Dec1 / Decrypt I", the proposed scheme is more efficient. In the end, because both "Enc / Encrypt" and "Dec2 / Decrypt II" happen in client side, their efficiencies have little effect on the entire system. Whereas, "Dec1 / Decrypt I" happens on broadcast center and "ReEnc" happens on cloud server, their low efficiencies can easily become the bottleneck of the entire system. In addition, considering that the proposed scheme supports the good feature of dynamically adding a user into or removing a user from the sharing group, we say the proposed scheme is more suitable for data sharing in cloud storage.

## VII. CONCLUSION

A secure and convenient way for users to share their encrypted data is a very important functionality requirement that cloud storage providers should considered to provide. In addition, the size of complete user set should not be limited beforehand, because the users in public cloud are organized loosely. Instead, it should admit users taking part in and quitting the sharing group dynamically and freely. Thus, we put forward a dynamic encrypted data sharing scheme based on conditional proxy broadcast re-encryption and illustrate its detailed design and implementation. Meanwhile, we prove the correctness and security of the proposed scheme, and analyze the space costs and computational costs of each algorithm involving in the proposed scheme. At last, we illustrate the feasibility of the proposed scheme through comparing the experimental results with Chu et al.'s scheme either.

## REFERENCES

[1] B. Feng, X. Ma, C. Guo, H. Shi, Z. Fu, and T. Qiu, "An efficient protocol with bidirectional verification for storage security in cloud computing," *IEEE Access*, vol. 4, pp. 7899–7911, 2016.

[2] C.-K. Chu, W.-T. Zhu, J. Han, J. K. Liu, J. Xu, and J. Zhou, "Security concerns in popular cloud storage services," *IEEE Pervasive Comput.*, vol. 12, no. 4, pp. 50–57, Oct./Dec. 2013.

[3] S. Berkovits, "How to broadcast a secret," in *Proc. Workshop Theory Appl. Cryptogr. Techn.*, 1991, pp. 535–541.

[4] D. Naor, M. Naor, and J. Lotspiech, "Revocation and tracing schemes for stateless receivers," in *Proc. Annu. Int. Cryptol. Conf.*, 2001, pp. 41–62.

[5] D. Halevy and A. Shamir, "The LSD broadcast encryption scheme," in *Proc. Annu. Int. Cryptol. Conf.*, 2002, pp. 47–60.

[6] Y. Dodis and N. Fazio, "Public key broadcast encryption for stateless receivers," in *Proc. ACM Workshop Digit. Rights Manage.*, 2002, pp. 61–80.

[7] C. Delerablée, P. Paillier, and D. Pointcheval, "Fully collusion secure dynamic broadcast encryption with constant-size ciphertexts or decryption keys," in *Proc. Int. Conf. Pairing-Based Cryptogr.*, 2007, pp. 39–59.

[8] C. Delerablée, "Identity-based broadcast encryption with constant size ciphertexts and private keys," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.*, 2007, pp. 200–215.

[9] C. Gentry and B. Waters, "Adaptive security in broadcast encryption systems (with short ciphertexts)," in *Proc. Annu. Int. Conf. Theory Appl. Cryptogr. Techn.*, 2009, pp. 171–188.

[10] J. Hur, C. Park, and S. O. Hwang, "Privacy-preserving identity-based broadcast encryption," *Inf. Fusion*, vol. 13, no. 4, pp. 296–303, 2012.

[11] Z. Zhou, D. Huang, and Z. Wang, "Efficient privacy-preserving ciphertext-policy attribute based-encryption and broadcast encryption," *IEEE Trans. Comput.*, vol. 64, no. 1, pp. 126–138, Jan. 2015.

[12] 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.

[13] 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.

[14] R. Canetti and S. Hohenberger, "Chosen-ciphertext secure proxy re-encryption," in *Proc. 14th ACM Conf. Comput. Commun. Secur.*, 2007, pp. 185–194.

[15] M. Green and G. Ateniese, "Identity-based proxy re-encryption," in *Proc. Int. Conf. Appl. Cryptogr. Netw. Secur.*, 2007, pp. 288–306.

[16] B. Libert and D. Vergnaud, "Unidirectional chosen-ciphertext secure proxy re-encryption," in *Proc. Int. Workshop Public Key Cryptogr.*, 2008, pp. 360–379.

[17] H. Wang and Z. Cao, "More efficient CCA-secure unidirectional proxy re-encryption schemes without random oracles," *Secur. Commun. Netw.*, vol. 6, no. 2, pp. 173–181, 2013.

[18] S. S. M. Chow, J. Weng, Y. Yang, and R. H. Deng, "Efficient unidirectional proxy re-encryption," in *Proc. Int. Conf. Cryptol. Africa*, 2010, pp. 316–332.

[19] G. Hanaoka *et al.*, "Generic construction of chosen ciphertext secure proxy re-encryption," in *Proc. Topics Cryptol. Cryptogr. Track RSA Conf.*, San Francisco, CA, USA, Feb 2012, pp. 349–364.

[20] J. W. Seo, D. H. Yum, and P. J. Lee, "Proxy-invisible CCA-secure type-based proxy re-encryption without random oracles," *Theor. Comput. Sci.*, vol. 491, pp. 83–93, Jun. 2013.

[21] J. Weng, R. H. Deng, X. Ding, C. K. Chu, and J. Lai, "Conditional proxy re-encryption secure against chosen-ciphertext attack," in *Proc. ACM ASIACCS*, 2009, pp. 322–332.

[22] C.-I. Fan, C.-N. Wu, C.-H. Chen, Y.-F. Tseng, and C.-C. Feng, "Attribute-based proxy re-encryption with dynamic membership," in *Proc. 10th Asia Joint Conf. Inf. Secur. (AsiaJCIS)*, May 2015, pp. 26–32.

[23] C. K. Chu, J. Weng, S. S. M. Chow, J. Zhou, and R. H. Deng, "Conditional proxy broadcast re-encryption," in *Proc. Austral. Conf. Inf. Secur. Privacy*, 2009, pp. 327–342.

[24] M. Sun, C. Ge, J. Wang, and L. Fang, "A proxy broadcast re-encryption for cloud data sharing," *Multimedia Tools Appl.*, vol. 76, no. 1, pp. 1–15, 2017.

[25] K. H. Rosen, *Elliptic Curves: Number Theory and Cryptography*. Boston, MA, USA: Chapman & Grimes, 2003.

[26] I. F. Blake, V. K. Murty, and G. Xu, "Refinements of Miller's algorithm for computing the Weil/Tate pairing," *J. Algorithms*, vol. 58, no. 2, pp. 134–149, 2006.

**LINMEI JIANG** received the B.S. degree in computer application from the Hangzhou Institute of Commerce in 1998 and the M.S. degree in computer application technology from Shanghai Normal University in 2007. He is currently pursuing the Ph.D. degree with Xiamen University. He is also a Lecturer with Huaqiao University. His research interests include network security, secure storage in cloud computing, and Web software engineering.

**DONGHUI GUO** (SM'17) received the Ph.D. degree in semiconductor physics and devices from Xiamen University in 1994. He is currently a Professor with Xiamen University and also the Director of the Integrated Circuit Engineering Technology Research Center. His research interests include cloud computing and information security, artificial intelligence, network communications, multi-core processor refactoring, nano-single electron devices, and biological sensor chip.