Multimed Tools Appl (2018) 77:18327–18354
https://doi.org/10.1007/s11042-017-5387-1

Efficient pairing-free PRE schemes for multimedia data sharing in IoT

Xing Hu[1,2] · Chunming Tang[3] · Duncan S. Wong[4] · Xianghan Zheng[5]

Received: 30 June 2017 / Revised: 1 October 2017 / Accepted: 2 November 2017 /
Published online: 20 November 2017

Abstract Nowadays, Internet of things (IoT) become more and more popular. At the same time, the requirements of security mechanism for multimedia in IoT received a huge concern. Multimedia data is easily shared by devises, applications and social networks set by IoT. Therefore, it is indispensable to guarantee the privacy and security of shared multimedia data. In this paper, we address the secure multimedia data sharing problem in cloud computing by designing proxy re-encryption (PRE) scheme. Our schemes cope with the issues of data validity, data confidentiality and authentication during encrypted multimedia data sharing. Unlike as usually done in the literature, we present a CCA-secure PRE scheme which removes pairings firstly. Then we design a refined CCA-secure PRE scheme called publicly verifiable PRE without parings. It is demonstrated that our schemes meet not only the security and high efficiency requirements of multimedia data sharing, but also the public verifiability. The validity of ciphertext, both the original and re-encrypted ciphertext, can be publicly verified which brings additional efficiency due to offloading the validity check of ciphertexts from the power-limited clients to any semi-honest public cloud.

Keywords Multimedia data security · Proxy re-encryption · CCA security · Without pairing · Public verifiability

✉ Chunming Tang
ctang@gzhu.edu.cn

[1] School of Mathematical and Computational Science, Hunan University of Science and Technology, Hunan, China

[2] Key Laboratory of Information Secirity, School of Mathematics and Information Science, Guangzhou University, Guangzhou, China

[3] School of Mathematics and Information Science, Guangzhou University, Guangzhou, China

[4] CryptoBLK Company, Hong Kong, China

[5] College of Mathematics and Computer Sciences, Fuzhou University, Fuzhou, China

1 Introduction

1.1 Motivation

Multimedia data includes all kinds of media types, such as audio, image, video etc. The proliferation of cloud computing make multimedia data sharing in networks much easier. Actually, cloud server plays an important role in multimedia data collecting. Multimedia data is now the main information source stored and transformed in the cloud. The truth, however, is that maximum endeavors are devoted to multimedia contents whereas less attention is paid to its security and privacy. In cloud settings, it is of particular importance to ensure a high privacy and security during multimedia data sharing.

1.2 Use case

Consider the following scenario:

1. A user, Alice, is planning to upload several multimedia files $F_1, F_2, \cdots, F_n$ to Dropbox. The general idea is for Alice to encrypt these files before sending them to Dropbox. The encryption is done using the conventional hybrid encryption paradigm. In particular, the files $F_1, F_2, \cdots, F_n$ are first encrypted to $C_1, C_2, \cdots, C_n$ under random symmetric content keys, $K_1, K_2, \cdots, K_n$, using a block cipher with an appropriate mode of operation (e.g. AES-CBC), then encrypt the keys $K_1, K_2, \cdots, K_n$ by using a public key encryption (PKE) scheme (e.g. ElGamal encryption scheme over an Elliptic Curve Group) under the user's public key. The ciphertexts uploaded to Dropbox consists of the encrypted multimedia files $C_1, C_2, \cdots, C_n$ and the encrypted content keys denoted by $CK_1, CK_2, \cdots, CK_n$. Note that in each encrypted multimedia file, say C_i, it also includes the initialization vector. For simplicity, we assume that the initialization vector is part of each encrypted private file, C_i.
2. At a certain time, the user Alice would like to share her multimedia files with a friend Bob. A conventional solution is that Alice downloads all the encrypted content keys $CK_1, CK_2, \cdots, CK_n$ from Dropbox, then decrypts them to obtain the content keysK_1, $K_2, \cdots, K_n$, and encrypts them again to $CK'_1, CK'_2, \cdots, CK'_n$, under Bob's public key, finally uploads these newly encrypted content keys to the Dropbox for Bob to download. The advantage of this solution is that Alice does not need to download any of the encrypted multimedia files, hence, can save the bandwidth during the communications between Alice and Dropbox. However, this solution still involves a lot of downloads and uploads of the encrypted content keys and the network overhead is linear to the number of files that Alice wants to share. Furthermore, this solution also incurs a lot of computation on Alice's side and may not be practical, especially for battery-powered computing devices.

The technical challenge in this use case is therefore on how to do this encrypted multimedia data sharing efficiently without triggering too much communications between the cloud server and the cloud user, and without incurring much computational burden to the user simultaneously.

There is another potential solution to this problem. The solution is to let Alice give out her private key to Dropbox, and let Dropbox do the decrypt-then-encrypt on behalf of Alice. However, this solution relies on the security of Dropbox and Alice has to trust Dropbox not to disclose the multimedia files to any third party without authorization. Hence this solution cannot provide much assurance to Alice that only she has the control on the accessibility of her own encrypted files.

By applying **PRE**, we target to minimize the communication between the cloud server and the cloud user for encrypted multimedia data sharing. We also target to reduce the user's computational burden, and at the same time, to ensure the privacy of Alice's encrypted multimedia files so that no adversary can obtain the files even after compromising the cloud storage service provider, i.e. Dropbox in the example above. We also applying a interesting property called public verifiability. With this property, the validity of ciphertexts can be publicly verified by anyone. So we can offload the validity check of ciphertexts from power-limited clients to any semi-honest public cloud to further improve the efficiency.

1.3 A practical and efficient encrypted multimedia data sharing solution using Velosti's USB device

The multimedia data owner, say Alice, has a Velosti USB device which contains her key-pair (i.e. public and private), and also a software which is called the encrypted cloud data client-side management software (in short, we call it a client software). In Alice's Dropbox folder, the client software creates a folder called Velosti. All the files stored in the Velosti folder will be encrypted in the hybrid encryption fashion as described above, but using the Velosti USB device. To access the encrypted files, Alice has to insert the Velosti USB device and execute the client software.

Furthermore, a copy of Alice's public key will be made available in the public folder of Alice's Dropbox so that all Dropbox users can get a copy of her public key once after learning the identity of Alice's Dropbox account. It is also the case for other users. For example, Bob, who will also have a copy of his public key in his Dropbox public folder.

Suppose Alice is about to share several encrypted files in the Velosti folder with Bob. Through the client software, Alice specifies the encrypted files that she wants to share with Bob. The client software will then notifies Dropbox using the Dropbox API on the files that Alice wants to share with Bob. Next, Dropbox will notify Bob about this sharing using Dropbox's existing data sharing notification protocol. However, since these files are encrypted, in particularly, the corresponding content keys are encrypted by using Alice's public key, Bob or anyone else cannot decipher these files. Hence, besides notifying Dropbox, Alice's client software also visits Bob's Dropbox public folder to get a copy of Bob's public key, and computes a transformation key ReKey using the private key of Alice and the public key of Bob. Notice that, the transformation key is used to complete a transformation from Alice's encrypted files to another form that Bob can decrypt. After generating the transformation key ReKey, Alice encrypts ReKey under Bob's public key and uploads a copy of the encrypted ReKey to Alice's public folder.

After receiving a sharing notification from Dropbox, Bob, via his own client software, visits Alice's public folder for getting the encrypted transformation key ReKey, then decrypts and recovers ReKey using his decryption key. By using the key ReKey and his private key, Bob can download the encrypted files from Dropbox that are shared by Alice, and decipher them.

In this PRE-based solution, no server is needed. The integration of security and the existing sharing mechanism of Dropbox is done seamlessly. The user experience is also enhanced by making use of the Velosti's USB device so that user passwords are not mandatory, instead, they are optional for providing the additional two-factor authentication.

1.4 Related work

Many researches has been done to provide the data privacy in IoT. Yang et al. [48] presented a fuzzy information retrieval scheme based on lattice assumption. Their contribution supports multiple user system without sharing secret key. This scheme is secure for multimedia cloud applications even in quantum-era. Wang et al. [47] proposed leakage resilient CP-ABE and KP-ABE schemes in a improved auxiliary input model. This scheme has been prove to be secure by constructing an improved strong extractor from the modified GoldreichCLevin theorem. Chang et al. [15] proposed a framework which is used in business clouds. Experiments have been designed in detail to show its robustness is secure in multilayered structure. Vijayakumar et al. [46] introduced an improved authentication for vehicular ad-hoc networks, and a method of anonymous authentication has been presented to preserve privacy. Amin et al. [1] presented an authentication protocol using smartcard, which is based on an architecture proposed in this paper for distributed cloud environment. This protocol allows the registered user to securely access all private information from all the private cloud servers.

Many approaches are available for protecting the shared multimedia data. Take encryption algorithm as an example, it transforms the multimedia data into encrypted form using a private key and the encrypted form can only be decrypted by the user hold the decryption key. Encryption is the primary tool which can guarantee the data privacy and against an unauthorized access [16, 24].

Commutative Encryption and Watermarking can provide extensive security for the multimedia data. Bouslimi et al. [11] presented a algorithm jointing watermarking and encryption. Its convergence primitives promotes the research of privacy and security [7]. Cancellaro et al. [12] combine the encryption and watermarking to protect image.

Besides, Bianchi [8] applied discrete Fourier transform on encrypted multimedia data. A combination of SVD and CA was proposed which can provide novel solutions to preserve multimedia data privacy [49]. Ye et al. [50] proposed the first JFE method to address the problem of multimedia data sharing.

PRE scheme can also be used for implementing secure multimedia data sharing, since any multimedia data can be transformed into binary.

A PRE scheme allows a private key holder (e.g. Alice) to produce a re-encryption key. By using it, a conversion from Alice's ciphertext C_A to Bob's ciphertext C_B can be made by a proxy (e.g. network server). Therefore, a PRE scheme can be applied into various applications, such as encrypted email forwarding [9], the DRM of Apple's iTunes [44], distributed file storage systems [4, 5], secure certified email mailing lists [35, 36] and access control [45]. In the above-mentioned cases, the core idea is the re-encryption.

The definition of PRE was first introduced by [9]. However, the presented PRE scheme is secure against chosen-plaintext attack (CPA).

Ivan et al. [33] presented a CCA security model for PRE, but Canetti et al. proved that the CCA security of their schemes is not hold [13]. Green and Ateniese [30] focus on CCA secure ID-based PRE and proposed a corresponding security model. Chu et al. [18] presented a ID-based PRE which removes random oracles. However, it is demonstrated that [18] was not CCA secure [43].

Homomorphic encryption (HE) scheme can also used to construct PRE schemes. Goldwasser et al [26] gave the first semantically secure additively HE scheme over Z_2. It is followed by other additively HE schemes, such as Paillier [40] and Damgard [21]. Besides, linear codes and lattices are also used to obtain additively HE schemes [2, 27, 34, 39, 41]. Another type of HE is multiplicative HE, and ElGamal [23] is the typical one.

A fully homomorphic encryption (FHE) scheme allows anyone to evaluate both additive or multiplicative functions over encrypted data without decrypting firstly. Gentry [28] proposed the first FHE scheme, and based on which a CPA-secure PRE was directly constructed. Subsequently, some progress were made in FHE [10, 19, 25]. However, these existing constructions of FHE are not suitable for practical uses due to efficiency drawbacks.

Another major concern about multimedia in IoT is that of supporting public verifiability for the encrypted multimedia data stored in remote network servers. The property of public verifiability enable anyone to complete the validity verification tasks without disclosing any private information. This property adds flexibility in various applications in IoT setting, especially multimedia data sharing. Although the property of public verifiability of ciphertexts is important, it received insufficient care from researchers.

If the validity of a ciphertext can only be checked by the receiver (delegatee) with his private key, the scheme is vulnerable ciphertext-malleable attack. The right ciphertext transferred in the network can be easily modified by the attackers, then lots of malicious ciphertexts can be created to instead the right ones. While these malicious created ciphertexts can be rejected by the receiver at the last minute, they have already caused great problem which can affect the users' feeling on using the scheme, even bring damage to the service providing corporations. If the validity of these ciphertexts can be checked publicly, the above problems can be easily solved, the routers or the access infrastructure can drop these maliciously created ciphertexts, and the bandwidth has been effectively preserved [29].

Canetti et al. [13] introduced the concept of public verifiability in PRE scheme. Libert et al. [38] proposed a publicly verifiable PRE which is unidirectional. However, Chow et al. [17] pointed out that, the security assurance supplied by [38] only against a weakened CCA [14].

Deng et al. [22] presented a construction of PRE which enabled the original ciphertext to be public verified, but it suffers from the attack in Remark 2 in [42]. Shao et al. [42] constructed a PRE by using signature of knowledge [3] to obtain public verifiability, but their public verifiability is only for original ciphertexts, and it is vulnerable to chosen-ciphertext attack [17].

So, public verifiability should be an essential property of CCA-secure PRE [4, 5, 13, 30, 51, 52]. Active attackers can issue queried to the data owner and receivers decryption oracle arbitrarily. If the proxy forward an invalid ciphertext to the receiver, and the receiver has decrypted it, some useful information can be derived and then used for breaking CCA security by the attackers. Although the proxy doesn't have the private key, he has to firstly verify the integrity of ciphertexts, so the property of publicly verify is essential to achieve the CCA security of PRE.

Moreover, most existing PRE schemes are constructed by pairings. Ateniese et al. [4, 5], Hohenberger et al. [31], Libert et al. [38] and Ateniese et al. [6] presented collusion resistant unidirectional PRE scheme respectively, those scheme are relied on pairings. However, those schemes are CPA secure.

It is worth noting that bilinear pairing is an important tool to construct PRE scheme, but it's implementation speed is relatively slower, especially in computational resource-constrained devices. Canetti et al. [13] raised an open problem that how to design a pairing-free PRE scheme. Afterwards, many researchers become interested in removing paring from the construction of PRE. Deng et al. [22], Shao [42] and Chow et al. [17] removed pairings from their PRE scheme respectively, but didn't achieve public verifiability.

Zhang et al. [53] care about how to construct publicly verifiable paring-free PKE scheme. They find it is very easy to construct publicly verifiable scheme for PKE.

Therefore, we want a PRE scheme simultaneously satisfy the following features: CCA-secure, high efficiency, public verifiability, paring-free and simple design.

1.5 Our contributions

In this paper, we research on the privacy and security protection mechanism of multimedia data in IoT by employing proxy re-encryption. We target to ensure the privacy and security of shared multimedia data, and at the same time, to reduce the multimedia data owner's computational burden. Our contributions are summarized below:

1. We propose a basic CPA-secure PRE scheme in which the bilinear parings is removed from the construction for efficiency and practical use.
2. To enhance the security of shared multimedia data in IoT, we propose a new CCA-secure paring-free PRE scheme based on the resulting CPA-secure one.
3. To ensure the validity of shared multimedia data in IoT, we construct a publicly verifiable PRE scheme which is CCA-secure, and also bilinear pairings is removed.

1.6 Organization

Section 2 introduces the definition of PRE scheme and its security models. In Section 3, we describe three PRE schemes without parings meet different security requirements, and the security proof and efficiency comparison are provided. Section 4 is the conclusion.

2 Definition and security models

We give the definition of PRE and its CCA security model as follows. A PRE is unidirectional, means that a ciphertext can be converted from one user to another without the opposite direction. If a ciphertext can only be transformed one time, the PRE scheme is single-hop. Namely that, a user can't further re-encrypt a re-encrypted ciphertext.

2.1 Definition of PRE

Definition 1 A PRE scheme is composed of 7 algorithms as follows:

1. $\mathbf{par} \leftarrow Setup(1^k)$: given a system parameter $k \in \mathbb{N}$, output a group of system parameters **par**.
2. $(pk_i, sk_i) \leftarrow KeyGen(\mathbf{par})$: given **par**, a pair of public/private key (pk_i, sk_i) is outputted. For ease of description, other algorithms take **par** as an implicitly input.
3. $rk_{i \to j} \leftarrow ReKeyGen(sk_i, pk_j)$: given sk_i (i.e. user i's private key) and pk_j (i.e. user j's public key), output $rk_{i \to j}$ as the re-encryption key. With this key, a ciphertext encrypted by pk_i will be transformed to another ciphertext encrypted by pk_j, here $poly(1^k)$ is some polynomial in k, $i \neq j$ and $i, j \in \{1, \ldots, poly(1^k)\}$.
4. $C_i \leftarrow Enc(pk_i, m)$: given pk_i and $m \in \mathcal{M}(pk_i)$, output an original ciphertext C_i, where $\mathcal{M}(pk_i)$ is a space of message.
5. $C_j \leftarrow ReEnc(rk_{i \to j}, C_i)$: given $rk_{i \to j}$ and C_i, a re-encrypted ciphertext C_j is outputted. Here $rk_{i \to j}$ is the re-encryption key, C_i is an original ciphertext under pk_i.
6. $m/ \perp \leftarrow Dec(sk_i, C_i)$: given sk_i and C_i, output message m if C_i is valid, otherwise, a symbol $\perp$ is outputted. Here sk_i is user i's private key, C_i is an original ciphertext under pk_i.

7. $m/\perp \leftarrow Dec_R(sk_j, C_j)$: given sk_j and C_j, output message m if C_j is valid, otherwise, a symbol $\perp$ is outputted. Here sk_j is user j's private key and C_j is a re-encrypted ciphertext.

The definition 1 is correct, because that for any $k \in \mathbb{N}$, any users $i \neq j \in \{1, ..., poly(1^k)\}$, and any message $m \in \mathcal{M}(pk_i)$, if $\mathbf{par} \leftarrow Setup(1^k)$ and $(pk_i, sk_i) \leftarrow KeyGen(\mathbf{par})$, then we have

$$C_i = Enc(pk_i, m), Dec(sk_i, C_i) = m;$$

$$C_j = ReEnc(ReKeyGen(sk_i, pk_j), C_i), Dec_R(sk_j, C_j) = m.$$

2.2 Security models

Definition 2 (Unidirectional Single-hop PRE IND-CCA Game) Denote k as the security parameter. Let $\mathcal{A}$ be a probabilistic polynomial time (PPT) adversary and let $\mathcal{B}$ be a game challenger. This game is composed of the following oracles executed by $\mathcal{A}$ and $\mathcal{B}$. These oracles can be invoked more than once and regardless of the order.

1. **Setup.** The challenger $\mathcal{B}$ runs $\mathbf{par} \leftarrow Setup(1^k)$ to generate **par** and give to $\mathcal{A}$ the output **par**.
2. **Phase 1.** The adversary $\mathcal{A}$ can issue the following oracles.
   (1) $\mathcal{O}_{pk}(i)$: given an index $i \in \{1, ..., poly(1^k)\}$, $\mathcal{B}$ runs $(pk_i, sk_i) \leftarrow KeyGen$ (**par**), then returns to $\mathcal{A}$ the pk_i.
   (2) $\mathcal{O}_{sk}(i)$: given a public key pk_i, $\mathcal{B}$ passes to $\mathcal{A}$ the private key sk_i, here $(pk_i, sk_i) \leftarrow KeyGen(\mathbf{par})$.
   (3) $\mathcal{O}_{rk}(pk_i, pk_j)$: given public keys pk_i and pk_j, $\mathcal{B}$ returns $rk_{i\to j} \leftarrow ReKeyGen(sk_i, pk_j)$ to $\mathcal{A}$, here $(pk_i, sk_i) \leftarrow KeyGen(\mathbf{par})$, $(pk_j, sk_j) \leftarrow KeyGen(\mathbf{par})$.
   (4) $\mathcal{O}_{re}(pk_i, pk_j, C_i)$: given public keys pk_i, pk_j and a user i's ciphertext C_i, $\mathcal{B}$ returns a re-encrypted ciphertext $C_j \leftarrow ReEnc(rk_{i\to j}, C_i)$ to $\mathcal{A}$, where $rk_{i\to j} \leftarrow ReKeyGen(sk_i, pk_j)$, $(pk_i, sk_i) \leftarrow KeyGen(\mathbf{par})$ and $(pk_j, sk_j) \leftarrow KeyGen(\mathbf{par})$.
   (5) $\mathcal{O}_{dec}(sk_i, C_i)$: on input C_i and pk_i, $\mathcal{B}$ returns $m \leftarrow Dec(sk_i, C_i)$, where $(pk_i, pk_j) \leftarrow KeyGen(\mathbf{par})$.
   (6) $\mathcal{O}_{dec_R}(sk_j, C_j)$: on input C_j and pk_j, $\mathcal{B}$ returns $m \leftarrow Dec_R(sk_j, C_j)$, where $(pk_j, sk_j) \leftarrow KeyGen(\mathbf{par})$.
3. **Challenge.** The adversary $\mathcal{A}$ returns two messages, say m_0, m_1, and a challenged public key pk_{i^*}. If the following queries
   (1) $\mathcal{O}_{sk}(pk_{i^*})$; and
   (2) $\mathcal{O}_{rk}(pk_{i^*}, pk_j)$ and $\mathcal{O}_{sk}(pk_j)$ for any index pk_j,

   are never made, $\mathcal{B}$ outputs $C_{i^*} = Enc(pk_{i^*}, m_b)$ for $\mathcal{A}$, here b is randomly choosen from $\{0, 1\}$. and pk_{i^*} is output by $\mathcal{O}_{pk}(i^*)$.
4. **Phase 2.** The adversary $\mathcal{A}$ issues queries as he did in **Phase 1**. However, the following queries are not issued:
   (1) $\mathcal{O}_{sk}(pk_{i^*})$;
   (2) $\mathcal{O}_{rk}(pk_{i^*}, pk_j)$ and $\mathcal{O}_{sk}(pk_j)$ for any index j;

(3) $\mathcal{O}_{re}(pk_{i^*}, pk_j, C_{i^*})$ and $\mathcal{O}_{sk}(pk_j)$ for any index $i \neq j$, here $(i, j \in \{1, ..., poly(1^k)\})$;
(4) $\mathcal{O}_{dec}(pk_{i^*}, C_{i^*})$; and
(5) $\mathcal{O}_{dec_R}(pk_j, C_j)$ for any pk_j and C_j, if (pk_j, C_j) is derived from (pk_{i^*}, C_{i^*}). As of [13], we define the derivative of (pk_{i^*}, C_{i^*}) as shown below.

   (a) (pk_{i^*}, C_{i^*}) is derived from itself.
   (b) If a query $\mathcal{O}_{rk}$ has been made on (pk_{i^*}, pk_j) by $\mathcal{A}$, a $rk_{i^*\to j}$ will be returned as the re-encryption key, then computed $C_j \leftarrow ReEnc(rk_{i^*\to j}, C_{i^*})$, we say (pk_j, C_j) is derived from (pk_{i^*}, C_{i^*}).
   (c) If a query $\mathcal{O}_{re}$ has been made on $(pk_{i^*}, pk_j, C_{i^*})$ by $\mathcal{A}$, and obtained C_j, then (pk_j, C_j) is a derivative of (pk_{i^*}, C_{i^*}).

5. **Guess.** The adversary $\mathcal{A}$ returns a value b' from $\{0, 1\}$ as his conjecture. If b' equals to b, $\mathcal{A}$ wins.

Definition 3 (CCA Security of Original Ciphertext) : Let $Adv_{PRE,\mathcal{A}}^{IND\text{-}CCA\text{-}Or}(1^k) = |Pr[b' = b] - \frac{1}{2}|$ be $\mathcal{A}$'s advantage in the game described in Definition 2. An unidirectional single-hop PRE scheme is $(t, q_{pk}, q_{sk}, q_{rk}, q_{re}, q_d, q_{d_R}, \epsilon)$-IND-CCA secure at original ciphertext, means that if any t-time IND-CCA adversary $\mathcal{A}$ is given at most q_{pk} queries to $\mathcal{O}_{pk}$, q_{sk} queries to $\mathcal{O}_{sk}$, q_{rk} queries to $\mathcal{O}_{rk}$, q_{re} queries to $\mathcal{O}_{re}$, q_d queries to $\mathcal{O}_{dec}$ and q_{d_R} queries to $\mathcal{O}_{dec_R}$, then we have $Adv_{PRE,\mathcal{A}}^{IND\text{-}CCA\text{-}Or} \leq \epsilon$.

Remark 1 The IND-CPA security at original ciphertext can be easily achieved from the above notion by only providing $\mathcal{O}_{pk}$, $\mathcal{O}_{sk}$ and $\mathcal{O}_{rk}$ for $\mathcal{A}$.

Definition 4 (CCA Security of Re-encrypted Ciphertext) Set $\bar{O} = \{\mathcal{O}_{pk}, \mathcal{O}_{sk}, \mathcal{O}_{rk}, \mathcal{O}_{dec}, \mathcal{O}_{dec_R}\}$. We define the advantage of $\mathcal{A}$ in the following experiment with given security parameter k and state information $State$,

$$\begin{aligned} Adv_{PRE,\mathcal{A}}^{IND\text{-}CCA\text{-}Re}(1^k) = |Pr[b = b' : \mathbf{par} \leftarrow Setup(1^k);\\ (C_0, C_1, pk_i, pk_{j^*}, State) \leftarrow \mathcal{A}^{\bar{O}}(\mathbf{par});\\ b \in_R \{0, 1\}; C_{j^*} \leftarrow ReEnc(rk_{i\to j^*}, C_b);\\ b' \leftarrow \mathcal{A}^{\bar{O}}(C_{j^*}, State)] - \frac{1}{2}|, \end{aligned}$$

here i and j^* are two distinct indices, pk_i and pk_j are outputted by $\mathcal{O}_{pk}$, $rk_{i\to j^*}$ is generated by $ReKeyGen(sk_i, pk_{j^*})$. C_0 and C_1 are valid ciphertexts constructed under pk_i by $\mathcal{A}$. Those oracles $\mathcal{O}_{pk}$, $\mathcal{O}_{sk}$, $\mathcal{O}_{rk}$, $\mathcal{O}_{dec}$, $\mathcal{O}_{dec_R}$ are defined in the above with limitation of the following constraints: if $\mathcal{A}$ makes queries $\mathcal{O}_{sk}$ on pk_{j^*}, $\mathcal{O}_{sk}$ outputs $\perp$. For $\mathcal{O}_{dec_R}$, the query on (pk_{j^*}, C_{j^*}) is forbidden to issue. There is no restriction on $\mathcal{O}_{rk}$ and $\mathcal{O}_{dec}$. Besides, $\mathcal{O}_{re}$ is unnecessary as $\mathcal{A}$ is allowed for querying any re-encryption key. An unidirectional single-hop PRE scheme is $(t, q_{pk}, q_{sk}, q_{rk}, q_d, q_{d_R}, \epsilon)$-IND-CCA secure at re-encrypted ciphertext, means that if any t-time IND-CCA adversary $\mathcal{A}$ can issue at most q_{pk} queries to $\mathcal{O}_{pk}$, q_{sk} queries to $\mathcal{O}_{sk}$, q_{rk} queries on $\mathcal{O}_{rk}$, q_d queries to $\mathcal{O}_{dec}$ and q_{d_R} queries to $\mathcal{O}_{dec_R}$, we have $Adv_{PRE,\mathcal{A}}^{IND\text{-}CCA\text{-}Re} \leq \epsilon$.

Remark 2 In Definition 3, if $\mathcal{A}$ can deduce the private key sk_{i^*} from $rk_{i^*\to j}$ (resp. $rk_{j\to i^*}$), $\mathcal{A}$ can definitely win the game above, where j is a corrupted user. Therefore, Definition 3 implies collusion resistance that the whole private key of the data owner can't be compromised by proxy even after compromising the corresponding receiver. We can deduce the IND-CPA security at re-encrypted ciphertext from the above notion by providing $\mathcal{O}_{pk}$, $\mathcal{O}_{sk}$ and $\mathcal{O}_{rk}$ for $\mathcal{A}$ only.

3 Our constructions

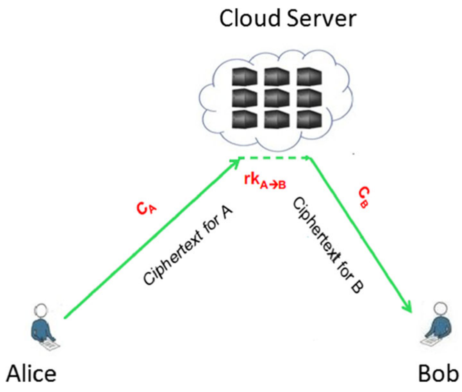
The core technology in our solutions is PRE which allows Alice to generate a re-encryption key $rk_{A\rightarrow B}$ with her decryption key and a friend's public key where the friend is whom that Alice intends to share her encrypted multimedia data with. For example, the friend is Bob. Then this re-encryption key $rk_{A\rightarrow B}$ will allow Bob to decrypt the encrypted content keys when used together with his own private key.

On the one hand, any multimedia data will be transformed into binary before travelling over the network, and on the other hand, proxy re-encryption (PRE) can be used as long as the data are in binary, so we can implement secure multimedia data sharing by design efficient PRE scheme. More specifically, in our PRE schemes, a message m represents a binary multimedia file.

Our system model is shown in Fig. 1. There are three roles in our scheme: multimedia data owner, a receiver and cloud server. The data owner, say Alice, will encrypt her multimedia files (such as images, audio, video etc.) using a PKE scheme (e.g. ElGamal encryption over an Elliptic Curve Group) before uploading them to the cloud server. These encrypted multimedia data (i.e. original ciphertext) are stored in the cloud. At a certain time, Alice would like to share her multimedia files with a friend Bob whereas share her private key. Therefore, Alice can use her privacy key and her friend's public key to generate $rk_{A\rightarrow B}$ by running the ReKeyGen algorithm of our scheme. The key $rk_{A\rightarrow B}$ allows the cloud server to transform the original ciphertext into another ciphertext (i.e. re-encrypted ciphertext). This re-encrypted ciphertext can only be decrypt by Bob with his privacy key. The detail explanation of our protocol is as follows.

In the following section, we propose three efficient PRE schemes meet different security requirements to guarantee the privacy and security of shared multimedia data. In our schemes, encrypted multimedia files can be shared between a user, Alice, and her friend without share Alice's private key. Each of Alice's friend can decrypt the multimedia files using his own privata key.

Fig. 1 System model



3.1 A basic CPA secure PRE scheme without parings

3.1.1 Construction

A basic CPA-secure PRE scheme without parings is proposed in this part. In this scheme, the cloud server is deemed to be semi-honest, meaning that it is honest but curious about the plaintext of multimedia data owner. In this scheme, with the re-encryption key $rk_{A\to B}$, the multimedia data owner's encrypted data will be transformed into another form that an anticipant receiver can decrypt. Finally, The receiver decrypt the re-encrypted data by his decryption key to obtain the plaintext which is the real multimedia files Alice intends to share. This scheme is follows:

1. $Setup(k)$: for a given security parameter $k \in \mathbb{N}$, a group G is generated with order q and $|q| = k$. Denote by g the G's generator. This algorithm also generates two hash functions H_1 and H_2, each of which maps from G to Z_q. The message space is defined as G. The system parameters of the PRE is set to $\mathbf{par} = (G, q, g, H_1, H_2)$.
2. $KeyGen(\mathbf{par})$:
    (1) Pick two random values $x_{i,1}, x_{i,2} \in_R Z_q$, set the private key $sk_i = (x_{i,1}, x_{i,2})$.
    (2) Set the public key $pk_i = (pk_{i,1}, pk_{i,2}) = (g^{x_{i,1}}, g^{x_{i,2}})$.
3. $Enc(pk_i, m)$: given $pk_i = (pk_{i,1}, pk_{i,2})$ and $m \in G$, a ciphertext C_i is generated as shown below. Here pk_i is user i's public key, $m \in G$ is a message.
    (1) Pick r randomly from Z_q.
    (2) Compute $E = mg^r$.
    (3) Compute $F = (pk_{i,1}^{H_2(pk_{i,2})} pk_{i,2})^r$.
    (4) Set $C_i = (E, F)$.
4. $ReKeyGen(sk_i, pk_j)$:
    (1) Randomly pick V from G and u from Z_q.
    (2) Compute $v = H_1(V)(x_{i,1}H_2(pk_{i,2}) + x_{i,2})^{-1} mod q$.
    (3) Compute $U = Vg^u$.
    (4) Compute $W = pk_{j,2}^u$.
    (5) Output $rk_{i\to j} = (v, U, W)$.
5. $ReEnc(rk_{i\to j}, C_i)$: given $rk_{i\to j} = (v, U, W)$ and $C_i = (E, F)$, a re-encrypted ciphertext C_j ia generated as shown below. Here $rk_{i\to j}$ is a re-encryption key and C_i is a ciphertext under pk_i.
    (1) Compute $F' = F^v$.
    (2) Output $C_j = (E, F', U, W)$.
6. $Dec(sk_i, C_i)$: given $sk_i = (x_{i,1}, x_{i,2})$ and $C_i = (E, F)$, the message m is recovered. Here sk_i is user i's private key, C_i is the original ciphertext under pk_i.
    (1) Compute $t = x_{i,1}H_2(pk_{i,2}) + x_{i,2}(mod q)$.
    (2) Output $m = E(F^{1/t})^{-1}$.
7. $Dec_R(sk_j, C_j)$:given $sk_j = (x_{j,1}, x_{j,2})$ and $C_j = (E, F', U, W)$, the message m is recovered. Here sk_j is user j's private key, C_j is a re-encrypted ciphertext.
    (1) Compute $V = U(W^{1/x_{j,2}})^{-1}$.
    (2) Output $m = E(F'^{1/H_1(V)})^{-1}$.

3.1.2 Security analysis

In this scheme, the multimedia data owner's files are encrypted by ElGamal encryption scheme. Therefore, the security of ElGamal encryption scheme can ensure that our scheme is secure. Moreover, throughout the whole process, as this re-encryption key $rk_{i\to j}$ alone does not allow anyone to recover the multimedia files, the network server gains no information about the multimedia data owner's files and private key.

3.1.3 Efficiency analysis

Let EXP represents the exponentiation operation in G (assuming that G is a multiplicative group, otherwise, if G is an additive group such as an elliptic curve group, then EXP represents the elliptic curve scalar multiplication), PreEXP denotes pre-computable exponentiation operation in G. Decrypt^O denotes the cost of decrypting an original ciphertext, Decrypt^R represents the decryption cost of a re-encrypted message. $|C^O|$ denotes the original ciphertext size, $|C^R|$ denotes the size of a re-encrypted message. $|ReKey|$ denotes the size of a re-encryption key.

From Table 1, we can conclude that the number of exponentiation operations needed in each algorithm is small (say one or two), the size of the ciphertext is at most 4 elements in G, and the $ReKey$ contains 3 elements (2 elements in G and 1 element in Z_q).

3.2 A CCA-secure paring-free PRE scheme

3.2.1 Construction

We proposed a CPA-secure PRE scheme in the above section. However, the CCA security is usually demanded in applications. For this purpose, we design a CCA-secure PRE scheme consisting of 7 algorithms:

1. $Setup(k)$: for a given security parameter $k \in \mathbb{N}$, the following steps are invoked:
   (1) Generate a group G with order q such that $|q| = k$, and picks a generator $g \in_R G$.
   (2) Set the massage space as $\{0, 1\}^k$.
   (3) Set four hash functions:
   $H_1 : G \to Z_q^*$, $H_2 : G \to Z_q^*$, $H_3 : G \to \{0, 1\}^k$, $H_4 : \{0, 1\}^k \times G \to Z_q^*$.
   (4) Output the public parameters $\mathbf{par} = (G, q, g, H_i)$ $(i = 1, \cdots, 4)$.

Table 1 Efficiency analysis

Algorithm	Operation
Encrypt	2 EXP + 1 PreEXP
ReEncrypt	1 EXP
Decrypt^O	1 EXP
Decrypt^R	2 EXP
$\lvert C^O\rvert$	2 $\lvert G\rvert$
$\lvert C^R\rvert$	4 $\lvert G\rvert$
$\lvert ReKey\rvert$	2 $\lvert G\rvert$ + $\lvert Z_q\rvert$

2. $KeyGen(\mathbf{par})$:
   (1) Pick two random values $x_{i,1}, x_{i,2} \in_R Z_q$, sets the private key $sk_i = (x_{i,1}, x_{i,2})$.
   (2) Set the public key $pk_i = (pk_{i,1}, pk_{i,2}) = (g^{x_{i,1}}, g^{x_{i,2}})$.

3. $Enc(pk_i, m)$: given $pk_i = (pk_{i,1}, pk_{i,2})$ and $m \in \{0, 1\}^k$, it carries out the following steps to generate a ciphertext C_i. Here pk_i is user i's public key, $m \in \{0, 1\}^k$ is a message.
   (1) Pick σ randomly from G, then compute $r = H_4(m, \sigma)$.
   (2) Compute $E = \sigma g^r$.
   (3) Compute $F = (pk_{i,1}^{H_2(pk_{i,2})} pk_{i,2})^r$.
   (4) Compute $J = m \oplus H_3(\sigma)$.
   (5) Set $C_i = (E, F, J)$.

4. $ReKeyGen(sk_i, pk_j)$:
   (1) Randomly pick V from G and u from Z_q.
   (2) Compute $v = H_1(V)(x_{i,1}H_2(pk_{i,2}) + x_{i,2})^{-1} mod q$.
   (3) Compute $U = Vg^u$.
   (4) Compute $W = pk_{j,2}^u$.
   (5) Output $rk_{i\to j} = (v, U, W)$.

5. $ReEnc(rk_{i\to j}, C_i)$: given $rk_{i\to j} = (v, U, W)$ and $C_i = (E, F, J)$, a re-encrypted ciphertext C_j is generated. Here $rk_{i\to j}$ is the re-encryption key, C_i is an original ciphertext under pk_i.
   (1) Compute $F' = F^v$.
   (2) Output $C_j = (E, F', J, U, W)$.

6. $Dec(sk_i, C_i)$: given $sk_i = (x_{i,1}, x_{i,2})$ and $C_i = (E, F, J)$, the message m is recovered. Here sk_i is user i's private key, C_i is the original ciphertext under pk_i.
   (1) Compute $t = x_{i,1}H_2(pk_{i,2}) + x_{i,2} mod q$.
   (2) Compute $\sigma' = E(F^{1/t})^{-1}$.
   (3) Compute $m' = J \oplus H_3(\sigma')$.
   (4) If $E = \sigma' g^{H_4(m',\sigma')}$ holds, output $m = m'$, otherwise output $\perp$.

7. $Dec_R(sk_j, C_j)$: given $sk_j = (x_{j,1}, x_{j,2})$ and $C_j = (E, F', J, U, W)$, the message m is recovered. Here sk_j is user j's private key, C_j is a re-encrypted ciphertext.
   (1) Compute $V = U(W^{1/x_{j,2}})^{-1}$.
   (2) Compute $\sigma' = E(F'^{1/H_1(V)})^{-1}$.
   (3) Compute $m' = J \oplus H_3(\sigma')$.
   (4) If $E = \sigma' g^{H_4(m',\sigma')}$ holds, output $m = m'$, otherwise output $\perp$.

3.2.2 Security analysis

Obviously, this scheme is CCA secure due to the underlying CCA-secure ElGamal encryption which generates the original ciphertext and re-encrypted ciphertext. Furthermore, since the re-encryption key $rk_{i\to j}$ alone does not allow anyone to recover the files from the encrypted files, it can ensure that the encrypted files will still remain secure even if an adversary has compromised cloud server and also obtained a copy of ReKey. In other words, the secrecy of the encrypted files is still relying on the secrecy of the private keys of multimedia data owner and her friend Bob (even after the encrypted files are shared). The detailed proof can be deduced from the security analysis described in Sections 3.3.3, 3.3.4. We then focus on the generating of re-encryption key.

1. Only sk_i has been taken as an input(sk_i is not involved), thus, our scheme is unidirectional.
2. Even if someone can obtained sk_j and $rk_{i\to j}$ simultaneously, the true value of $x_{i,1}$ or $x_{i,2}$ are still remain secure, as the $H_1(V)$ can be recovered only leak information about the value of $x_{i,1}H_2(pk_{i,2})+x_{i,2}$, so that the secret security of the multimedia data owner is ensured.

Remark 3 IoT has the merits of low cost and effective accessibility. However, network servers may not be fully trusted. The validity related to the data shared between users is problematic. In Internet, the users are resources-limited and hence cannot afford excessive validity checks. Therefor, in practice, it is more reasonable to add public verifiability into the construction of scheme. With public verifiability, anyone, not just the data owner, is allowed to complete the validity verification tasks without keeping any private information. Let's consider the goal that allowing network servers to verify the correctness of ciphertext on behalf of the multimedia data owner. In the next section, we give an improved CCA secure PRE scheme, and give an thorough security analysis.

3.3 A CCA-secure publicly verifiable PRE scheme without paring (PVPRE)

3.3.1 Main idea

In practice, the multimedia data in IoT is stored in remote servers and exposed to malicious attackers. Moreover, in the context of PRE, the remote server is asked to complete the transformation from an encrypted multimedia data under the owner's public key to another form that an anticipated recipient can decrypt, it is probable for an attacker to derive sensitive information or even tamper with the encrypted multimedia data with his own sake. The following attack gives an explanation.

Let $C' = (E', F', J')$ be a challenged ciphertext encrypted by a challenged public key $pk' = (pk'_{i,1}, pk'_{i,2}) = (g^{x'_{i,1}}, g^{x'_{i,2}})$, where $sk' = (x'_{i,1}, x'_{i,2})$ is the challenged private key, $E' = \sigma' g^{r'}$, $F' = ({pk'_{i,1}}^{H_2(pk'_{i,2})} pk'_{i,2})^{r'}$, $J' = m \oplus H_3(\sigma')$. Suppose C' is given to the adversary $\mathcal{A}$ and he will win the IND-CCA secure game as the following: Firstly, $\mathcal{A}$ chooses a random t from $\{0, 1\}^l$, and creates a new malicious ciphertext $C_1 = (E_1, F_1, J_1)$ instead of C', here $E_1 = E'$, $F_1 = F'$, $J_1 = J \oplus t$. Obviously, C_1 is an invalid one. Secondly, $\mathcal{A}$ get a key pair (pk'', sk'') by making a corrupted-key generation query, and also get re-encrypted ciphertext $C_2 = (E_2, F_2, J_2, U, W)$ by making a re-encryption query on pk'', here $pk'' = (pk''_{i,1}, pk''_{i,2}) = (g^{x''_{i,1}}, g^{x''_{i,2}})$, $sk'' = (x''_{i,1}, x''_{i,2})$, $E_2 = E_1$, $F_2 = F_1^v$, $J_2 = J_1$, (v, U, W) is the re-encryption key. Finally, $\mathcal{A}$ can use private key $x''_{i,2}$ to obtain $V = U(W^{1/x''_{i,2}})^{-1}$ and $\sigma_2 = E_2({F_2}^{1/H_1(V)})^{-1}$, then recover m as $m = t \oplus H_3(\sigma_2) \oplus J_2$. And then $\mathcal{A}$ can recover the bit δ which means $\mathcal{A}$ wins the game. We note that the queries $\mathcal{A}$ issued above are legal, because they follows the restraints in definition 2.

The adversary $\mathcal{A}$'s attack is successful due to the reason that the validity of re-encrypted ciphertext can not be verified by the proxy (server). Thus, it is fascinating to embed public verifiability into a CCA-secure PRE scheme.

Next, we briefly describe how the public verifiability is used. Firstly, we modifies the above scheme slightly such that the original ciphertext generated by algorithm $Enc(pk_i, m)$ is the form $C_i = (E, F, J, s)$. Suppose the proxy(server) is asked to perform a ciphertext transformation. The proxy verifies $(pk_{i,1}^{H_2(pk_{i,2})} pk_{i,2})^s = E \cdot F^{H_5(E,F,J)}$ firstly to guarantee the validity of C_i, and then outputs the re-encryption ciphertext $C_j = (E', F', J, s', U, W)$.

The validity of the re-encryption ciphertext can be verified before being re-encrypted and decrypted. Thus, it is impossible for malicious attackers to obtain any advantage through tampering with the re-encrypted ciphertext.

3.3.2 Construction

Now, we give the details of our construction. We proposed a refined publicly verifiable PRE scheme called PVPRE in which the validity of ciphertexts can be publicly verified by anyone [32]. Although capturing this useful property comes at a price: three more components need to be computed in Enc and $ReEnc$, the public verifiability feature is attractive, it is worth the performance tradeoff. We now shown the construction of the publicly verifiable scheme.

1. $Setup(k)$: for a given security parameter k, the following steps are invoked:

   (1) Generate a group G with order q such that $|q| = k$, and picks a generator $g \in_R G$.
   (2) Define the massage space as $\{0, 1\}^k$.
   (3) Define five hash functions:

   $$H_1 : G \to Z_q^*, H_2 : G \to Z_q^*, H_3 : G \to \{0, 1\}^k,$$
   $$H_4 : \{0, 1\}^k \times G \to Z_q^*, H_5 : G \times G \times G \to Z_q^*.$$

   (4) Output the public parameter $\textbf{par} = (G, q, g, H_i)$ $(i = 1, \cdots, 5)$.

2. $KeyGen(\textbf{par})$:

   (1) Pick two random values $x_{i,1}, x_{i,2} \in_R Z_q$, set $sk_i = (x_{i,1}, x_{i,2})$ as the private key.
   (2) Define $pk_i = (pk_{i,1}, pk_{i,2}) = (g^{x_{i,1}}, g^{x_{i,2}})$ as the public key.

3. $Enc(pk_i, m)$: given $pk_i = (pk_{i,1}, pk_{i,2})$ and $m \in \{0, 1\}^k$, it carries out the following steps to generate a ciphertext C_i. Here pk_i is user i's public key and $m \in \{0, 1\}^k$ is a message.

   (1) Randomly pick σ from Z_q^*, then compute $r = H_4(m, g^\sigma)$.
   (2) Compute $E = (pk_{i,1}^{H_2(pk_{i,2})} pk_{i,2})^\sigma$.
   (3) Compute $F = (pk_{i,1}^{H_2(pk_{i,2})} pk_{i,2})^r$.
   (4) Compute $J = m \oplus H_3(g^\sigma)$.
   (5) Compute $s = \sigma + rH_5(E, F, J) mod q$.
   (6) Output $C_i = (E, F, J, s)$.

4. $ReKeyGen(sk_i, pk_j)$:

   (1) Randomly pick $V \leftarrow G$, then compute $u = H_1(V)$.
   (2) Compute $v = H_2(V)(x_{i,1}H_2(pk_{i,2}) + x_{i,2})^{-1} mod q$.
   (3) Compute $U = Vg^u$.
   (4) Compute $W = pk_{j,2}^u$.
   (5) Output $rk_{i \to j} = (v, U, W)$.

5. $ReEnc(rk_{i\to j}, C_i)$: given $rk_{i\to j} = (v, U, W)$ and $C_i = (E, F, J, s)$, a re-encrypted ciphertext C_j is generated. Here $rk_{i\to j}$ is a re-encryption key, C_i is the original ciphertext under pk_i.

   (1) If $(pk_{i,1}^{H_2(pk_{i,2})} pk_{i,2})^s = E \cdot F^{H_5(E,F,J)}$ is not satisfied, then return $\perp$. Otherwise,
   (2) Compute $E' = E^v$ and $F' = F^v$.
   (3) Compute $s' = sv(modq)$.
   (4) Output $C_j = (E', F', J, s', U, W)$.

6. $Dec(sk_i, C_i)$: given $sk_i = (x_{i,1}, x_{i,2})$ and $C_i = (E, F, J, s)$, the message m is recovered. Here sk_i is user i's private key, and C_i is the original ciphertext under pk_i.

   (1) If $(pk_{i,1}^{H_2(pk_{i,2})} pk_{i,2})^s = E \cdot F^{H_5(E,F,J)}$ is not satisfied, then return $\perp$. Otherwise,
   (2) Compute $t = x_{i,1}H_2(pk_{i,2}) + x_{i,2}(modq)$.
   (3) Compute $g^{\sigma'} = E^{1/t}$.
   (4) Compute $m' = J \oplus H_3(g^{\sigma'})$.
   (5) If $F = (pk_{i,1}^{H_2(pk_{i,2})} pk_{i,2})^{H_4(m', g^{\sigma'})}$ holds, output $m = m'$, otherwise output $\perp$.

7. $Dec_R(sk_j, C_j)$: given $sk_j = (x_{j,1}, x_{j,2})$ and $C_j = (E', F', J, s', U, W)$, the message m is recovered. Here sk_j is user j's private key, C_j is a re-encrypted ciphertext.

   (1) If $(pk_{i,1}^{H_2(pk_{i,2})} pk_{i,2})^{s'} = E' \cdot F'^{H_5(E',F',J)}$ is not satisfied, then return $\perp$. Otherwise,
   (2) Compute $V = U(W^{1/x_{j,2}})^{-1}$.
   (3) Compute $g^{\sigma'} = E'^{1/H_2(V)}$.
   (4) Compute $m' = J \oplus H_3(g^{\sigma'})$.
   (5) If $F' = g^{H_4(m', g^{\sigma'})H_2(V)}$ and $W = pk_{j,2}^{H_1(V)}$ hold, output $m = m'$, otherwise output $\perp$.

Remark 4 The refined scheme PVPRE can guarantee both the multimedia data owner's and the recipient's anonymity simultaneously. Each proxy or anticipant recipient can easily check the validity of ciphertexts without disclosing any sensitive information.

3.3.3 Original ciphertext security analysis

Definition 5 (Decisional Diffie-Hellman (DDH) Assumption) Consider a group G of order q, and let g be a generator of G. The DDH assumption states that, given a tuple (g, g^a, g^b, g^d) for uniformly and independently chosen $a, b, d \in Z_{q^*}$, decide whether $d = ab$.

For a given $\mathcal{A}$ with at most q_{H_i} queries to H_i $(i \in \{1, 3, 4, 5\})$ to break the $(t, q_{pk}, q_{sk}, q_{rk}, q_{re}, q_d, q_{d_R}, \epsilon)$-IND-CCA security of PVPRE, a polynomial time algorithm $\mathcal{B}$ will be constructed who can break the DDH assumption in G.

Our proofs works under the random oracle model. Those oracles simulated by $\mathcal{B}$ are depicted in Table 2. The tuple (G, q, g, H_i) $(i \in \{1, 3, 4, 5\})$ is given to $\mathcal{A}$. $\mathcal{B}$ controls the random oracles $H_i (i \in \{1, 3, 4, 5\})$ and also keeps hash lists H_i^{list} $(i \in \{1, 3, 4, 5\})$, they are initialized empty. $\mathcal{B}$ answers the queries whenever $\mathcal{A}$ issues. These answers are shown in Table 2.

$\mathcal{B}$ also keeps two lists K^{list} and R^{list}, they are initialized empty. Here the lists K^{list} is used to store key-pair (i.e. public key and private key) and the re-encryption key is stored in list R^{list}.

Table 2 Simulations of H_i $(i = 1, 3, 4, 5)$

$H_i(.)$	Simulations
$H_1(Q)$	Given a tuple (Q, ρ), the H_1^{list} return the predefined value ρ. Otherwise, pick ρ randomly from Z_q^*, add tuple (Q, ρ) to H_1^{list}, then return $H_1(Q) = \rho$.
$H_3(R)$	Given a tuple (R, ξ), the H_3^{list} return the predefined value ξ. Otherwise, pick ξ randomly from $\{0, 1\}^k$, add tuple (R, ξ) to H_3^{list}, then return $H_3(R) = \xi$.
$H_4(m, g^{\sigma})$	Given a tuple $(m, , \sigma, g^{\sigma}, r)$, the H_4^{list} return the predefined value r. Otherwise, pick r randomly from Z_q^*, add tuple $(m, , \sigma, g^{\sigma}, r)$ to H_4^{list}, then return $H_4(m, g^{\sigma}) = r$.
$H_5(E, F, J)$	Given a tuple (E, F, J, γ), the H_5^{list} return the predefined value γ. Otherwise, pick γ randomly from Z_q^*, add tuple (E, F, J, γ) to H_5^{list}, then return $H_5(E, F, J) = \gamma$.

Theorem 1 *The scheme PVPRE is IND-CCA secure at the original ciphertext, if the DDH assumption hold in group G.*

Proof **Phase 1** A number of queries are issued by adversary $\mathcal{A}$, $\mathcal{B}$ responses to $\mathcal{A}$ as follows.

1. $\mathcal{O}_{pk}(i)$: the uncorrupted-keys and corrupted-keys are generated by $\mathcal{B}$ as shown below.
   (1) Uncorrupted-key. $\mathcal{B}$ choose $x_{i,1}, x_{i,2} \leftarrow Z_q^*$ randomly and draws a coin $c_i \in \{0, 1\}$ that generates 1 with probability θ and 0 otherwise [20].
      (a) If $c_i = 1$, define $pk_i = (pk_{i,1}, pk_{i,2}) = (g^{x_{i,1}}, g^{x_{i,2}})$.
      (b) If $c_i = 0$, define $pk_i = (pk_{i,1}, pk_{i,2}) = ((g^{\frac{1}{a}})^{x_{i,1}}, (g^{\frac{1}{a}})^{x_{i,2}})$.

      Then, the tuple $(pk_i, x_{i,1}, x_{i,2}, c_i)$ is added to K^{list} and pk_i is returned to $\mathcal{A}$.
   (2) Corrupted-key. $\mathcal{B}$ choose $x_{i,1}, x_{i,2} \leftarrow Z_q^*$ randomly, and set $pk_i = (g^{x_{i,1}}, g^{x_{i,2}})$, $c_i =' -'$. Then the tuple $(pk_i, x_{i,1}, x_{i,2}, c_i)$ is added to K^{list} and output $(pk_i, (x_{i,1}, x_{i,2}))$ to $\mathcal{A}$.
2. $\mathcal{O}_{sk}(i)$: $\mathcal{B}$ recovers $(pk_i, x_{i,1}, x_{i,2}, c_i)$ firstly from K^{list}. If $c_i = 1$, output $(pk_i, (x_{i,1}, x_{i,2}))$ to $\mathcal{A}$, else return a bit $b \in_R \{0, 1\}$ then **aborts**.
3. $\mathcal{O}_{rk}(pk_i, pk_j)$: If there is a tuple (pk_i, pk_j) in R^{list}, it outputs $\mathcal{A}$ the predefined re-encryption key. Otherwise, $\mathcal{B}$ takes action as shown below:
   (1) Extract two tuple $(pk_i, x_{i,1}, x_{i,2}, c_i)$, $(pk_j, x_{j,1}, x_{j,2}, c_j)$ by searching K^{list}.
   (2) Randomly choose $V \leftarrow G$, compute $u = H_1(V)$ and $h = H_2(V)$.
   (3) Compute $U = Vg^u$ and $W = pk_{j,2}^u$.
   (4) Compute v according to the following case:
      (a) $(c_i = 0 \wedge c_j =' -')$, output $\perp$ and **aborts**.
      (b) $(c_i = 1 \vee c_j =' -')$, sets $v = h(x_{i,1}H_2(pk_{i,2}) + x_{i,2})^{-1} mod q$ and set $\tau = 1$. In this case, v is obviously correct due to $sk_i = (x_{i,1}, x_{i,2})$.
      (c) $(c_i = 0 \wedge c_j \neq' -')$, randomly pick $v \leftarrow Z_q^*$ and set $\tau = 0$. In this case, the value h, which related to U, W, would not match with a random v, this depends on the CCA security of ElGamal encryption scheme.

(5) If $\mathcal{B}$ does not **aborts**, add $(pk_i, pk_j, (v, U, W), h, \tau)$ to R^{list}.
(6) Output $rk_{i\rightarrow j} = (v, U, W)$ to $\mathcal{A}$.

4. $\mathcal{O}_{re}(pk_i, pk_j, C_i)$:

(1) If $(pk_{i,1}^{H_2(pk_{i,2})} pk_{i,2})^s \neq E \cdot F^{H_5(E,F,J)}$, return symbol $\perp$ which means C_i is invalid.
(2) Otherwise, extracts tuples $(pk_i, x_{i,1}, x_{i,2}, c_i)$ and $(pk_j, x_{j,1}, x_{j,2}, c_j)$ by searching K^{list}.
(3) If condition $c_i = 0$ and $c_j =' -'$ are not satisfied simultaneously, the query $\mathcal{O}_{rk}(pk_i, pk_j)$ is issued to generate a $rk_{i\rightarrow j} = (v, U, W)$ for $\mathcal{A}$.
(4) Else, searching the tuple $(R, \beta) \in H_3^{list}$ and $(m, \sigma, g^\sigma, r) \in H_4^{list}$ such that $(pk_{i,1}^{H_2(pk_{i,2})} pk_{i,2})^\sigma = E$, $(pk_{i,1}^{H_2(pk_{i,2})} pk_{i,2})^r = F$. If no eligible tuple exist, output $\perp$.
(5) Extract $(pk_i, pk_j, (v, U, W), h, \tau)$ from R^{list}, define $E' = g^{\sigma h}$, $F' = g^{rh}$, $s' = sv$.
(6) Return $C_j = (E', F', J, s', U, W)$ to $\mathcal{A}$.

5. $\mathcal{O}_{dec}(pk_i, C_i)$: $\mathcal{B}$ first parses $pk_i = (pk_{i,1}, pk_{i,2})$ and extract tuple $(pk_i, x_{i,1}, x_{i,2}, c_i)$ by searching K^{list}.

(1) If $(c_i = 1 \vee c_j =' -')$, $\mathcal{B}$ runs $Dec((x_{i,1}, x_{i,2}), c_i)$, then output the result to $\mathcal{A}$.
(2) Else,

(a) if $(pk_{i,1}^{H_2(pk_{i,2})} pk_{i,2})^s \neq E \cdot F^{H_5(E,F,J)}$, output symbol $\perp$ which indicates C_i is invalid.
(b) else, search list H_3^{list} and H_4^{list} to find tuples $(R, \beta) \in H_3^{list}$ and $(m, \sigma, g^\sigma, r) \in H_4^{list}$ such that $(pk_{i,1}^{H_2(pk_{i,2})} pk_{i,2})^\sigma = E$, $(pk_{i,1}^{H_2(pk_{i,2})} pk_{i,2})^r = F$, $\beta \oplus m = J$, $R = g^\sigma$. if such two tuples are exist, output m to $\mathcal{A}$. else output $\perp$.

6. $\mathcal{O}_{dec_R}(pk_j, C_j)$: $\mathcal{B}$ first parses $pk_j = (pk_{j,1}, pk_{j,2})$ and recovers tuple $(pk_j, x_{j,1}, x_{j,2}, c_j)$ from K^{list}. If $(c_j = 1 \vee c_j =' -')$, $\mathcal{B}$ runs $Dec_R((x_{j,1}, x_{j,2}), C_j)$ and returns the result to $\mathcal{A}$. Otherwise,

(1) if $(pk_{i,1}^{H_2(pk_{i,2})} pk_{i,2})^{s'} \neq E' \cdot F'^{H_5(E',F',J)}$, output symbol $\perp$ which indicates C_j is invalid.
(2) Else, if there exists a tuple $(pk_i, pk_j, (v, U, W), V, 0) \in R^{list}$, compute $E = E'^{\frac{1}{v}}$, $F = F'^{\frac{1}{v}}$, search to see whether there exist $(R, \beta) \in H_3^{list}$ and $(m, \sigma, g^\sigma, r) \in H_4^{list}$ such that $(pk_{i,1}^{H_2(pk_{i,2})} pk_{i,2})^\sigma = E$, $(pk_{i,1}^{H_2(pk_{i,2})} pk_{i,2})^r = F$, $\beta \oplus m = J$, $R = g^\sigma$. If such two tuples are exist, output m to $\mathcal{A}$, otherwise, output $\perp$. Actually, the value of each U, W in R^{list} is correct.

Challenge If $\mathcal{A}$ find that Phase 1 is finish, it returns 3 contents (pk_{i^*}, m_0, m_1), here $pk_{i^*} = (pk_{i^*,1}, pk_{i^*,2})$ is a challenged public key, $m_0, m_1 \in \{0, 1\}^k$ are messages. Algorithm $\mathcal{B}$ extract tuple $(pk_{i^*}, x_{i^*,1}, x_{i^*,2}, c_{i^*})$ by searching K^{list}. In accordance with the constraints in definition 2, we obtain $c_{i^*} \in \{0, 1\}$, $\mathcal{B}$ chooses $\delta \in \{0, 1\}$ and acts as shown below:

1. If $c_{i^*} = 1$, challenger $\mathcal{B}$ picks a value $b \in_R \{0, 1\}$, then **aborts**.
2. Else, compute $E^* = (g^b)^{x_{i^*,1} H_2(pk_{i^*,2}) + x_{i^*,2}}$.

3. Choose $e^*, t^* \leftarrow Z_q^*$, set $s^* = e^* t^*$ randomly and compute
$$F^* = (g^b)^{-(x_{i^*,1} H_2(pk_{i^*,2}) + x_{i^*,2})\frac{1}{e^*}} \times (g^{\frac{1}{a}})^{(x_{i^*,1} H_2(pk_{i^*,2}) + x_{i^*,2}) t^*}.$$
4. Choose $J^* \leftarrow \{0,1\}^k$ randomly, set $H_5(E^*, F^*, J^*) = e^*$.
5. Choose $\sigma^* \leftarrow Z_q^*$ randomly, implicitly define $\sigma^* = d$ and $H_3(g^d) = m_\delta \oplus J^*$.
6. Output a challenge original ciphertext $C^* = (E^*, F^*, J^*, s^*)$ for $\mathcal{A}$.

According to the construction, if $d = ab$, the challenge ciphertext C^* is indistinguishable from the real one. This can be demonstrated as follows. Let $\sigma^* \triangleq ab$, $r^* \triangleq (t^* - \frac{ab}{e^*})$, we have

$$\begin{aligned}
E^* &= (g^b)^{x_{i^*,1} H_2(pk_{i^*,2}) + x_{i^*,2}} \\
&= ((g^{\frac{1}{a}})^{x_{i,1} H_2(pk_{i,2}) + x_{i,2}})^{ab} \\
&= (pk_{i^*,1}^{H_2(pk_{i^*,2})} pk_{i^*,2})^{\sigma^*} \\
F^* &= (g^b)^{-(x_{i^*,1} H_2(pk_{i^*,2}) + x_{i^*,2})\frac{1}{e^*}} \times (g^{\frac{1}{a}})^{(x_{i^*,1} H_2(pk_{i^*,2}) + x_{i^*,2}) t^*} \\
&= ((g^{\frac{1}{a}})^{x_{i^*,1} H_2(pk_{i^*,2}) + x_{i^*,2}})^{(t^* - \frac{ab}{e^*})} \\
&= (pk_{i^*,1}^{H_2(pk_{i^*,2})} pk_{i^*,2})^{r^*} \\
J^* &= H_3(g^{ab}) \oplus m_\delta = H_3(g^{\sigma^*}) \oplus m_\delta \\
s^* &= ab + (e^* t^* - ab) \\
&= ab + (t^* - \frac{ab}{e^*}) e^* \\
&= \sigma^* + r^* \cdot H_5(E^*, F^*, J^*).
\end{aligned}$$

Phase 2 $\mathcal{A}$ makes queries continuously according to the constraints in definition 2. Challenger $\mathcal{B}$ answers to $\mathcal{A}$'s queries.

Guess $\mathcal{A}$ passes to $\mathcal{B}$ a bit $\delta' \in \{0,1\}$ as its conjecture. If δ' equals to δ, $\mathcal{B}$ return 1 meaning $d = ab$; otherwise returns 0 meaning random value $d \in_R Z_q^*$.

The description of the simulation is completed. Next, the correctness of the simulation above will be demonstrated.

Analysis With adversary $\mathcal{A}$, the DDH problem can be solved with the advantage ϵ' by algorithm $\mathcal{B}$ within time t', here

$$\begin{aligned}
\epsilon' &\geq \frac{1}{q_{H_3}} \left(\frac{2\epsilon}{e(1+q_{rk})} - \frac{q_d + q_{d_R} + 2q_{re}}{q} - \frac{(q_{H_3} + q_{H_4})(q_d + q_{d_R}) + q_{H_5}}{2^k} - \epsilon_1 \right), \\
t' &\leq t + \left(\sum_{i=1}^{5} q_{H_i} + q_{pk} + q_{sk} + q_{rk} + q_{re} + q_d + q_{d_R} \right) O(1) \\
&+ (7q_{re} + 2q_{rk} + (q_{H_3} + q_{H_4}) q_{re} + (6 + q_{H_3} + q_{H_4})(q_d + q_{d_R})) t_{exp}.
\end{aligned}$$

Denotes t_{exp} as the time cost that an exponentiation operation needed in G, let ϵ_1 be the advantage of breaking the CCA security of ElGamal encryption.

The key point of our correctness proof is referenced from [17]. We analysis these simulations firstly. Obviously, according to the construction of H_4, the corresponding simulation is perfect. Denote by $AskH_4^*$ a event that issue query to H_4 on (m, g^σ), similarly, let $AskH_5^*$ be a event that (E^*, F^*, J^*) has been queried to H_5 before Challenge phase. The simulation of H_4 and H_5 are prefect, only if $\mathcal{A}$ neither query (m, g^{σ}) to H_4 nor (E^*, F^*, J^*) to H_5. In Challenge phase, since J^* is chosen from $\{0,1\}^k$ randomly, we have $Pr[AskH_5^*] \leq \frac{q_{H_5}}{2^k}$.

Let $AskH_3$ be the event that g^d has been queried to H_3. The corresponding simulation is also prefect, only if g^d is not queried to H_3 by $\mathcal{A}$ during the Challenge phase.

It is obvious that the simulated queries for public/private key generation are perfect. Denote **Aborts** be the event that $\mathcal{B}$ aborts when interacts with $\mathcal{A}$ in a query $\mathcal{O}_{rk}$ or challenge phase. Notice that the probability $Pr[\neg\textbf{Aborts}]$ is given by $\theta^{q_{rk}}(1-\theta)$ with the upper bound $\frac{q_{rk}}{1+q_{rk}}$, then we have $\theta^{q_{rk}}(1-\theta) \geq \frac{1}{e(1+q_{rk})}$.

Then, we can see that the simulation of query $\mathcal{O}_{rk}$ is not different from the real one, except for the case $(c_i = 0 \wedge c_j \neq' -')$, here the component v is chosen randomly. If the event **Aborts** did not happen, the real one and its corresponding simulation are computationally indistinguishable for the following truth:

1. $c_j \neq' -'$ indicate that the private key sk_j is unknown to $\mathcal{A}$.
2. $(pk_{j,2}^u, Vg^u)$ with $u = H_1(V)$ is actually an ciphertext of V encrypted under $pk_{j,2}$ by using the underlying ElGamal encryption scheme based on the DDH assumption.

Now, we analyze the simulation of query $\mathcal{O}_{re}$. If $\mathcal{A}$ cannot submit a valid original ciphertext without querying H_3 and H_4 (denoted by **REErr**), the simulation of re-encryption query $\mathcal{O}_{re}$ is perfect too. However, since H_3 and H_4 act as random oracles, we have $Pr[REErr] \leq \frac{2q_{re}}{q}$.

The simulations of the decryption oracles, namely $\mathcal{O}_{dec}$ and $\mathcal{O}_{dec_R}$, are perfect, unless the simulation errors happened in the situation that a valid ciphertext is rejected. But, it is not significant for these errors happening, the reason is as follows. Assume that a decryption query Q has been issued. Even if Q is a valid query, it is possible to generate Q with a probability without querying H_3 on g^{σ}.

Denote **Valid** as the event indicating Q is a valid query, denote $AskH_3$ as the event that g^{σ} has been queried to H_3. We note that the probability that $\mathcal{A}$ can lead to a valid J with reference to the output of H_3 without querying H_3 is $\frac{1}{q}$. Then, we have$Pr[Valid|\neg AskH_3] \leq \frac{1}{q}$.

Denote $DecErr$ as the event that $Valid|\neg AskH_3$ occurred during the whole simulation. As the decryption oracles issued by $\mathcal{A}$ is at most $(q_d + q_{d_R})$, we get $Pr[DecErr] \leq \frac{(q_{H_3}+q_{H_4})(q_d+q_{d_R})}{2^k} + \frac{q_d+q_{d_R}}{q}$.

Finally, let Err be the event $(AskH_3^* \vee AskH_5^* \vee REErr \vee DecErr)|\neg\textbf{Aborts}$. If Err dose not happen, since the output of H_3 is random, the advantage of $\mathcal{A}$ in guessing a δ is less than $\frac{1}{2}$. In other word, $Pr[\delta' = \delta|\neg Err] = \frac{1}{2}$ holds. Therefore, we expand $Pr[\delta = \delta']$ to obtain

$$\begin{aligned} Pr[\delta = \delta'] &= Pr[\delta = \delta'|Err]Pr[Err] + Pr[\delta = \delta'|\neg Err]Pr[\neg Err] \\ &\leq Pr[Err] + \frac{1}{2}Pr[\neg Err] = \frac{1}{2}Pr[Err] + \frac{1}{2} \end{aligned}$$

and $Pr[\delta = \delta'] \geq Pr[\delta = \delta'|\neg Err]Pr[\neg Err] = \frac{1}{2} - \frac{1}{2}Pr[Err]$.

By the definition of ϵ, we have

$$\begin{aligned} \epsilon &\leq |Pr[\delta = \delta'] - \frac{1}{2}| \leq \frac{1}{2}Pr[Err] \\ &= \frac{1}{2}Pr[(AskH_3^* \vee AskH_5^* \vee REErr \vee DecErr)|\neg\textbf{Aborts}] \\ &\leq (Pr[AskH_5^*] + Pr[AskH_3^*] + Pr[REErr] + Pr[DecErr])/2Pr[\neg\textbf{Aborts}], \end{aligned}$$

then we have

$$\begin{aligned} Pr[AskH_3^*] &\geq 2Pr[\neg\textbf{Aborts}]\cdot\epsilon - Pr[AskH_5^*] - Pr[DecErr] - Pr[REErr] \\ &\geq \frac{2\epsilon}{e(1+q_{rk})} - \frac{q_{H_5}}{2^k} - \frac{(q_{H_3}+q_{H_4})(q_d+q_{d_R})}{2^k} - \frac{q_d+q_{d_R}}{q} - \frac{2q_{re}}{q} \\ &= \frac{2\epsilon}{e(1+q_{rk})} - \frac{q_d+q_{d_R}+2q_{re}}{q} - \frac{(q_{H_3}+q_{H_4})(q_d+q_{d_R})+q_{H_5}}{2^k}. \end{aligned}$$

Meanwhile, if $AskH_3^*$ occur, algorithm $\mathcal{B}$ can solve the DDH instance. Therefore, we have

$$\begin{aligned} \epsilon' &\geq \frac{1}{q_{H_3}} Pr[AskH_3^*] \\ &= \frac{1}{q_{H_3}}\left(\frac{2\epsilon}{e(q_{rk}+1)} - \frac{q_d+q_{d_R}+2q_{re}}{q} - \frac{(q_{H_3}+q_{H_4})(q_d+q_{d_R})+q_{H_5}}{2^k}\right). \end{aligned}$$

Base on the above simulations, the bound on algorithm $\mathcal{B}$'s running time is given by

$$\begin{aligned} t' \leq t &+ \left(\sum_{i=1}^{5} q_{H_i} + q_{pk} + q_{sk} + q_{rk} + q_{re} + q_d + q_{d_R}\right) O(1) \\ &+ (2q_{rk} + 7q_{re} + (q_{H_3}+q_{H_4})q_{re} + (6+q_{H_3}+q_{H_4})(q_d+q_{d_R}))t_{exp}. \end{aligned}$$

□

3.3.4 Re-encrypted ciphertext security analysis

For re-encrypted ciphertext security, the task is to decide decide whether $d = \frac{b}{a}$ given $(g, g^a, g^b, g^d) \in G^3$ with unknown $a, b \leftarrow Z_q^*$. $H_i (i \in \{1, 3, 4, 5\})$ is the same as proof of Theorem 1.

Theorem 2 *Our scheme PVPRE is IND-CCA secure at the re-encrypted ciphertext, if the DDH assumption holds in group G.*

Proof **Phase 1** A number of queries issued by adversary $\mathcal{A}$, $\mathcal{B}$ responses to $\mathcal{A}$ as follows.

1. $\mathcal{O}_{pk}(i)$: the uncorrupted-keys and corrupted-keys are generated by $\mathcal{B}$ as shown below .
   (1) Uncorrupted-key. $\mathcal{B}$ randomly choose $x_{i,1}, x_{i,2} \leftarrow Z_q^*$ and draws a coin $c_i \in \{0, 1\}$ that generates 1 with probability θ and 0 otherwise [20].
      (a) If $c_i = 1$, set $pk_i = (pk_{i,1}, pk_{i,2}) = \left((g^a)^{\frac{1}{H_2(pk_{i,2})}} \cdot g^{x_{i,1}}, \frac{g^{x_{i,2}}}{g^a}\right)$.
      (b) If $c_i = 0$, set $pk_i = (pk_{i,1}, pk_{i,2}) = ((g^a)^{x_{i,1}}, (g^a)^{x_{i,2}})$.

      Next, the tuple $(pk_i, x_{i,1}, x_{i,2}, c_i)$ is added to K^{list} and return pk_i to the adversary $\mathcal{A}$.
   (2) Corrupted-key. $\mathcal{B}$ acts as the same in Theorem 1.
2. $\mathcal{O}_{rk}(pk_i, pk_j)$: If there is a tuple (pk_i, pk_j) in R^{list}, output to $\mathcal{A}$ a re-encryption key which is predefined. Otherwise, $\mathcal{B}$ takes action as shown below:
   (1) Extract two tuple $(pk_i, x_{i,1}, x_{i,2}, c_i)$, $(pk_j, x_{j,1}, x_{j,2}, c_j)$ by searching K^{list}.
   (2) Compute $rk_{i\to j}$ under the following situation:

1) If $(c_i = 1 \vee c_i =' -')$:
   - (a) Randomly pick $V \leftarrow G$, compute $u = H_1(V)$ and $h = H_2(V)$.
   - (b) Set $v = h(x_{i,1}H_2(pk_{i,2}) + x_{i,2})$ and $\tau = 1$.
   - (c) Compute $U = Vg^u$ and $W = pk_{j,2}^u$.

2) If $(c_i = 0 \wedge c_j = 0)$
   - (a) Randomly pick $v \leftarrow Z_q^*$, set $\tau = 0$.
   - (b) Randomly pick $z \leftarrow Z_q^*$, set $g^u = (g^{\frac{b}{a}})^{\frac{z}{x_{j,2}}}$, which defines $W = (g^b)^z$.
   - (c) Randomly pick $U \leftarrow \{0, 1\}^k$, implicitly define $V = \frac{U}{g^u}$.

3) If $(c_i = 0 \wedge c_j \neq 0)$: output $\perp$ and **aborts**.

(3) If $\mathcal{B}$ does not **aborts**, add $(pk_i, pk_j, (v, U, W), z, \tau)$ into list R^{list}.
(4) Return $rk_{i\to j} = (v, U, W)$ to the adversary $\mathcal{A}$.

3. $\mathcal{O}_{dec}(pk_i, C_i)$: $\mathcal{B}$ acts as the same in Theorem 1.
4. $\mathcal{O}_{dec_R}(pk_j, C_j)$: $\mathcal{B}$ acts as the same in Theorem 1.

Challenge If $\mathcal{A}$ find that Phase 1 is finish, it returns 4 components $(pk_i, pk_{j^*}, m_0, m_1)$, here pk_i is a public key, pk_{j^*} is a challenge public key, m_0 and m_1 are messages. Algorithm $\mathcal{B}$ extract two tuples $(pk_i, x_{i,1}, x_{i,2}, c_i)$ and $(pk_{j^*}, x_{j^*,1}, x_{j^*,2}, c_{j^*})$ by searching K^{list}. Based on the restriction in definition 2, we have $c_i, c_{j^*} \in \{0, 1\}$, $\mathcal{B}$ chooses a $\delta \in \{0, 1\}$ then make simulations for a challenged ciphertext. More specifically,

1. If $c_i = 1$ or $c_{j^*} = 1$, algorithm $\mathcal{B}$ returns a value $b \in_R \{0, 1\}$, then **aborts**.
2. If $c_i = 0 \wedge c_{j^*} = 0$, algorithm $\mathcal{B}$ generates the challenge ciphertext by the following steps:

(1) Retrieve $(pk_i, pk_{j^*}, (v^*, U^*, W^*), z^*, 0)$ from R^{list}.
(2) Randomly pick $t \leftarrow Z_q^*$, set $E'^* = (g^b)^t$, implicitly define $\sigma^* h^* = bt$, i.e. $\sigma^* = \frac{bt}{h^*}$.
(3) Randomly pick $e^* \leftarrow Z_q^*$, set $F'^* = (g^b)^e$, implicitly define $r^* h^* = be$, i.e. $r^* = \frac{be}{h^*}$.
(4) Randomly pick $J^* \leftarrow \{0, 1\}^k$, implicitly define

$$J^* = H_3((g^{\frac{b}{a}})^{\frac{t}{v^*(x_{i,1}H_2(pk_{i,2})+x_{x_i,2})}}) \oplus m_\delta.$$

Recall that $h^* = H_2(V^*) = v^* a(x_{i,1}H_2(pk_{i,2}) + x_{i,2})$ for $c_i = 0$.
(5) Randomly pick $k^* \leftarrow Z_q^*$, set $H_5(E'^*, F'^*, J^*) = k^*$, which defines $s'^* = t^* h^* + e^* k^* h^*$.
(6) Otherwise, take the following steps to set U^*, W^* and z^*.
   - (a) Randomly pick $z^* \leftarrow Z_q^*$, set $g^{u^*} = (g^d)^{\frac{z^*}{x_{j,2}}}$, implicitly define $W^* = (g^b)^{z^*}$.
   - (b) Randomly pick $U^* \leftarrow \{0, 1\}^k$, implicitly define $V^* = \frac{U^*}{g^{u^*}}$
   - (c) Add U^*, W^* and z^* into R^{list}.

(7) Pass to $\mathcal{A}$ the $C^* = (E'^*, F'^*, J^*, s'^*, U^*, W^*)$ as the challenged re-encrypted ciphertext.

Phase 2 $\mathcal{A}$ makes queries continuously according to the restrictions in definition 2. $\mathcal{B}$ answers to $\mathcal{A}$'s queries.

Guess $\mathcal{A}$ passes to $\mathcal{B}$ a bit $\delta' \in \{0, 1\}$ as its conjecture. If δ' equals to δ, $\mathcal{B}$ return 1 meaning $d = \frac{b}{a}$; otherwise returns 0 meaning $d \in_R Z_q^*$.

The description of the simulation is completed. We now show the correctness of the above simulation.

Analysis With adversary $\mathcal{A}$, the DDH problem can be solved with advantage ϵ' by $\mathcal{B}$ within time t', here

$$\epsilon' \geq \frac{1}{q_{H_3}}\left(\frac{2\epsilon}{e(1+q_{rk})} - \frac{q_d + q_{d_R}}{q} - \frac{(q_{H_3}+q_{H_4})(q_d+q_{d_R}) + q_{H_5}}{2^k} - \epsilon_1\right),$$

$$t' \leq t + \left(\sum_{i=1}^{5} q_{H_i} + q_{pk} + q_{sk} + q_{rk} + q_{re} + q_d + q_{d_R}\right) O(1)$$
$$+((6 + q_{H_3} + q_{H_4})(q_d + q_{d_R}) + 2q_{rk})t_{exp}.$$

Denote t_{exp} as the time cost that an exponentiation operation needed in G, let ϵ_1 be the advantage of breaking the CCA security of ElGamal encryption.

As described in Theorem 1, it is clear that the answers passed to $\mathcal{A}$ are all perfect, including the queries of public and private key generation, re-encryption, and also re-encryption key generation. The simulations of the two decryption queries are perfect too, unless the simulation errors happened in the case of rejecting some valid ciphertext which denoted by $DecErr$. As in Theorem 1, a similar analysis can yield $Pr[DecErr] \leq \frac{(q_{H_3}+q_{H_4})(q_d+q_{d_R})}{2^k} + \frac{q_d+q_{d_R}}{q}$.

Based on the above analysis, we can conclude that the simulations for H_i $(i = 1, 2, 4)$ are perfect too.

Denote $AskH_3^*$ as the event that $(g^{\frac{b}{a}})^{\frac{t}{v^*(x_{i,1}H_2(pk_i,2)+x_{x_i},2)}}$ has been queried to H_3, $AskH_5^*$ denotes the event that (E^*, F^*, J^*) has been queried to H_5. The simulation of H_3 and H_5 are perfect too, only if $AskH_3^*$ and $AskH_5^*$ doesn't occur, here t and J^* are chosen randomly. Denote Err as the even $(AskH_3^* \vee AskH_5^* \vee REErr \vee DecErr)|\neg$**Aborts**. As in Theorem 1, A similar analysis can yield

$$\begin{aligned}
Pr[AskH_3^*] &\geq 2Pr[\neg\textbf{Aborts}] \cdot \epsilon - Pr[AskH_5^*] - Pr[DecErr] \\
&\geq \frac{2\epsilon}{e(1+q_{rk})} - \frac{q_{H_5}}{2^k} - \frac{(q_{H_3}+q_{H_4})(q_d+q_{d_R})}{2^k} - \frac{q_d+q_{d_R}}{q} \\
&= \frac{2\epsilon}{e(1+q_{rk})} - \frac{q_d+q_{d_R}}{q} - \frac{(q_{H_3}+q_{H_4})(q_d+q_{d_R})+q_{H_5}}{2^k}.
\end{aligned}$$

Meanwhile, if $AskH_3^*$ occurred, the DDH instance can be solved by $\mathcal{B}$. Therefore, we have

$$\begin{aligned}
\epsilon' &\geq \frac{1}{q_{H_3}} Pr[AskH_3^*] \\
&= \frac{1}{q_{H_3}}\left(\frac{2\epsilon}{e(1+q_{rk})} - \frac{q_d+q_{d_R}}{q} - \frac{(q_{H_3}+q_{H_4})(q_d+q_{d_R})+q_{H_5}}{2^k}\right).
\end{aligned}$$

Based on the above simulations, the bound on algorithm $\mathcal{B}$'s running time is given by

$$t' \leq t + \left(\sum_{i=1}^{5} q_{H_i} + q_{pk} + q_{sk} + q_{rk} + q_{re} + q_d + q_{d_R} \right) O(1) \\ + (2q_{rk} + (6 + q_{H_3} + q_{H_4})(q_d + q_{d_R}))t_{exp}.$$

□

3.3.5 *Efficiency analysis*

We now make an efficiency comparison between our PVPRE and Chow et al. [17].

Notations in Table 3, namely EXP, PreEXP, Decrypt^O, Decrypt^R, $|C^O|$, $|C^R|$, $|ReKey|$, are the same meaning as in Table 1. Specifically, $(pk_{i,1}^{H_2(pk_{i,2})} pk_{i,2})$ can be pre-computed in our scheme PVPRE.

Nine aspects are compared in Table 3. These are described in more detail below.

- **Encrypt algorithm:** This algorithm have the same exponentiation operations both in [17] and ours.
- **ReEncrypt algorithm:** In scheme [17], there are 6 exponentiation operations needed to be calculated, while 4 exponentiation operations needed in our scheme.
- **Decrypt^O algorithm:** This algorithm have the same exponentiation operations both in [17] and ours.
- **Decrypt^R algorithm:** This algorithm have 4 exponentiation operations, while 6 in ours.
- C^O **Size:** The original ciphertext contains 4 components (3 in G and 1 in Z_q) both in [17] and ours.
- C^R **Size:** In scheme [17], the re-encrypted ciphertext has 4 G components. In our scheme, there are 6 components (5 in G and 1 in Z_q).
- $ReKey$ **Size:** The re-encryption key size is the same both in [17] and ours.
- **Pairing-Free Feature:** Both the scheme [17] and ours are removing pairing from the construction.
- **Public Verifiability Feature:** In scheme [17], only the original ciphertext can be verified. In our scheme, the validity of ciphertexts can be publicly verified, that is, anyone can check the validity of an original ciphertext as well as a re-encrypted ciphertext.

From Table 3, compared with the Chow et al. scheme [17], our PVPRE scheme is more efficient by saving two exponentiation operations in G at the algorithm **ReEncrypt** of our

Table 3 Efficiency comparison

Algorithm	Chow et al. [17]	Our scheme PVPRE
Encrypt	3 EXP + 1 PreEXP	3 EXP + 1 PreEXP
ReEncrypt	6 EXP + 1 PreEXP	4 EXP + 1 PreEXP
Decrypt^O	4 EXP + 1 PreEXP	4 EXP + 1 PreEXP
Decrypt^R	4 EXP	6 EXP + 1 PreEXP
$\lvert C^O \rvert$	$3\ \lvert G\rvert + \lvert Z_q\rvert$	$3\ \lvert G\rvert + \lvert Z_q\rvert$
$\lvert C^R \rvert$	$4\ \lvert G\rvert$	$5\ \lvert G\rvert + \lvert Z_q\rvert$
$\lvert ReKey \rvert$	$5\ \lvert G\rvert + \lvert Z_q\rvert$	$5\ \lvert G\rvert + \lvert Z_q\rvert$
Pairing-Free	Y	Y
Public verifiability	N	Y

scheme PVPRE. More importantly, our scheme PVPRE achieves the public verifiability by using only two more exponentiations in G at the Decrypt^R phase. It is worth the performance tradeoff, since the public verifiability feature is attractive, which makes our scheme PVPRE more flexible in various applications, such as multimedia data sharing.

Remark 5 In our scheme PVPRE, the computational complexity incurred by generating ReKey as well as the ReKey size are both independent to the number of encrypted files to be shared with Bob, and the validity check of ciphertexts can be offloaded from Alice to the semi-honest cloud. Hence, we solve both of the technical problems described in Section 1. First, we significantly reduce the computational burden of Alice during multimedia data sharing. Second, as this re-encryption key ReKey alone does not allow anyone to recover the files from the encrypted files, it can ensure that the encrypted files will still remain secure even if an adversary has compromised Dropbox and also obtained a copy of ReKey. In other words, the secrecy of the encrypted files is still relying on the private keys secrecy of multimedia data owner and his friend, even after the encrypted files are shared.

4 Conclusion

In this work, we address the privacy and security problem of multimedia data sharing in IoT by developing new PRE schemes. In contrast to all existing CCA-secure schemes in which the public verifiability is depended on bilinear parings, we construct a new publicly verifiable CCA-secure PRE scheme in which the costly pairings is removed. And the efficiency comparison demonstrates that our proposed scheme is highly efficient than most existing pairing-base PRE schemes. More importantly, our constructions satisfy the following features simultaneously: (1) CCA-secure; (2) paring-free; (3) public verifiability; (4)simple design. We believe that our design will be useful for fostering multimedia data security and also improving the usability of secure IoT.

We also raise some open problems, such as constructing PRE scheme with the following features: (1)multi-hop, (2)bidirectional, (3)pairing-free, (4)CCA-secure and (5)publicly verifiable.

Acknowledgement I would like to thank Prof. Duncan S. WONG for his instructive suggestions, which have significantly improved the design and revised the manuscript. Hu's research was supported in part by the Scientific Research Fund of Hunan Provincial Education Department (No. 15C0536), Tang's research was supported in part by the Foundation of National Natural Science of China (No. 11271003), Guangdong Province Natural Science Foundation of major basic research and Cultivation project (No. 2015A030308016), Project of Ordinary University Innovation Team Construction of Guangdong Province (No. 2015KCXTD014), Basic Research Major Projects of Department of education of Guangdong Province (No. 2014KZDXM044) and Collaborative Innovation Major Projects of Bureau of Education of Guangzhou City (No. 1201610005). The authors declare that they have no conflict of interest.

References

1. Amin R, Kumar N, Biswas GP, Iqbal R., Chang V (2016) A light weight authentication protocol for IoT-enabled devices in distributed Cloud Computing environment. Future Generation Computer Systems

2. Armknecht F, Sadeghi AR (2008) A new approach for algebraically homomorphic encryption. IACR Cryptology ePrint Archive 2008:422
3. Ateniese G, Camenisch J, Joye M, Tsudik G (2000) A practical and provably secure coalition-resistant group signature scheme. In: Annual international cryptology conference. Springer, Berlin, pp 255–270
4. Ateniese G, Fu K, Green M, Hohenberger S (2005) Improved proxy re-encryption schemes with applications to secure distributed storage. In: IN NDSS
5. Ateniese G, Fu K, Green M, Hohenberger S (2006) Improved proxy re-encryption schemes with applications to secure distributed storage. ACM Trans Inf Syst Secur (TISSEC) 9(1):1–30
6. Ateniese G, Benson K, Hohenberger S (2009) Key-private proxy re-encryption. In: Cryptographers Track at the RSA Conference. Springer, Berlin, pp 279–294
7. Bianchi T, Piva A (2013) Secure watermarking for multimedia content protection: a review of its benefits and open issues. IEEE Signal Proc Mag 30(2):87–96
8. Bianchi T, Piva A, Barni M (2009) On the implementation of the discrete Fourier transform in the encrypted domain. IEEE Trans Inf Forensics Secur 4(1):86–97
9. Blaze M, Bleumer G, Strauss M (1998) Divertible protocols and atomic proxy cryptography. In: Advances in CryptologyEUROCRYPT'98, pp 127–144
10. Boneh D, Goh EJ, Nissim K (2005) Evaluating 2-DNF formulas on ciphertexts. In: TCC, vol 3378, pp 325–341
11. Bouslimi D, Coatrieux G, Roux C (2012) A joint encryption/watermarking algorithm for verifying the reliability of medical images: application to echographic images. Comput Methods Prog Biomed 106(1):47–54
12. Cancellaro M, Battisti F, Carli M, Boato G, De Natale FG, Neri A (2011) A commutative digital image watermarking and encryption method in the tree structured Haar transform domain. Signal Process Image Commun 26(1):1–12
13. Canetti R, Hohenberger S (2007) Chosen-ciphertext secure proxy re-encryption. In: Proceedings of the 14th ACM conference on computer and communications security. ACM, pp 185–194
14. Canetti R, Krawczyk H, Nielsen JB (2003) Relaxing chosen-ciphertext security. In: Annual international cryptology conference. Springer, Berlin, pp 565–582
15. Chang V, Kuo YH, Ramachandran M (2016) Cloud computing adoption framework: a security framework for business clouds. Futur Gener Comput Syst 57:24–41
16. Cheng H, Li X (2000) Partial encryption of compressed images and videos. IEEE Trans Signal Process 48(8):2439–2451
17. Chow SS, Weng J, Yang Y, Deng RH (2010) Efficient unidirectional proxy re-encryption. In: International conference on cryptology in Africa. Springer, Berlin, pp 316–332
18. Chu CK, Tzeng WG (2007) Identity-based proxy re-encryption without random oracles. In: International conference on information security. Springer, Berlin, pp 189-202
19. Cohen JD, Fischer MJ (1985) A robust and verifiable cryptographically secure election scheme. Yale University. Department of Computer Science, pp 372–382
20. Coron JS (2000) On the exact security of full domain hash. In: Annual international cryptology conference. Springer, Berlin, pp 229–235
21. Damgard I, Jurik M (2003) A length-flexible threshold cryptosystem with applications. In: ACISP, vol 3, pp 350–356
22. Deng RH, Weng J, Liu S, Chen K (2008) Chosen-ciphertext secure proxy re-encryption without pairings. In: International conference on cryptology and network security. Springer, Berlin, pp 1–17
23. ElGamal T (1985) A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Trans Inf Theory 31(4):469–472
24. Fouda JAE, Effa JY, Sabat SL, Ali M (2014) A fast chaotic block cipher for image encryption. Commun Nonlinear Sci Numer Simul 19(3):578–588
25. Goldwasser S, Micali S (1982) Probabilistic encryption and how to play mental poker keeping secret all partial information. In: Proceedings of the fourteenth annual ACM symposium on Theory of computing. ACM, pp 365–377
26. Goldwasser S, Micali S (1984) Probabilistic encryption. J Comput Syst Sci 28(2):270–299
27. Goldwasser S, Kharchenko D (2005) Proof of plaintext knowledge for the Ajtai-Dwork cryptosystem. In: TCC, pp 529–555
28. Gentry C (2009) Fully homomorphic encryption using ideal lattices. In: STOC, vol 9, pp 169–178
29. Goto K, Sasaki Y, Hara T, Nishio S (2013) Data gathering using mobile agents for reducing traffic in dense mobile wireless sensor networks. Mob Inf Syst 9(4):295–314

30. Green M, Ateniese G (2007) Identity-based proxy re-encryption. In: Applied Cryptography and Network Security. Springer, Berlin, pp 288–306
31. Hohenberger S, Rothblum GN, Vaikuntanathan V (2007) Securely obfuscating re-encryption. In: Theory of cryptography conference. Springer, Berlin, pp 233–252
32. Hu X, Tang C, Wong DS (2016) Highly efficient proxy re-encryption schemes for user-end encrypted cloud data sharing. In: 2016 15th International Symposium on Parallel and Distributed Computing (ISPDC). IEEE, pp 261–268
33. Ivan AA, Dodis Y (2003) Proxy cryptography revisited. In: NDSS
34. Kawachi A, Tanaka K, Xagawa K (2007) Multi-bit cryptosystems based on lattice problems. In: International workshop on public key cryptography. Springer, Berlin, pp 315–329
35. Khurana H, Hahm HS (2006) Certified mailing lists. In: Proceedings of the 2006 ACM Symposium on information, computer and communications security. ACM, pp 46–58
36. Khurana H, Slagell A, Bonilla R (2005) SELS: A secure e-mail list service. In: Proceedings of the 2005 ACM symposium on applied computing. ACM, pp 306–313
37. Libert B, Vergnaud D (2008) Unidirectional chosen-ciphertext secure proxy re-encryption. In: International workshop on public key cryptography. Springer, Berlin, pp 360–379
38. Libert B, Vergnaud D (2008) Tracing malicious proxies in proxy re-encryption. In: International conference on pairing-based cryptography. Springer, Berlin, pp 332–353
39. Melchor CA, Castagnos G, Gaborit P (2008) Lattice-based homomorphic encryption of vector spaces. In: IEEE international symposium on information theory, 2008. ISIT 2008. IEEE, pp 1858–1862
40. Paillier P (1999) Public-key cryptosystems based on composite degree residuosity classes. In: Eurocrypt, vol 99, pp 223–238
41. Peikert C, Waters B (2011) Lossy trapdoor functions and their applications. SIAM J Comput 40(6):1803–1844
42. Shao J, Cao Z (2009) CCA-Secure proxy re-encryption without pairings. In: International workshop on public key cryptography. Springer, Berlin, pp 357–376
43. Shao J, Xing D, Cao Z (2008) Analysis of cca secure unidirctional id-based pre scheme. Technical Report of TDT, Shanghai Jiao Tong University
44. Smith T (2005) DVD Jon: buy DRM-less tracks from apple itunes. 2012-10-01]. http://www.theregister.co.uk/2005/03/18/itunes_pymusique.
45. Talmy A, Dobzinski O (2006) Abuse freedom in access control schemes. In: 20th international conference on advanced information networking and applications, 2006. AINA 2006, vol 2. IEEE, pp 77–86
46. Vijayakumar P, Azees M, Chang V, Deborah J, Balusamy B (2017) Computationally efficient privacy preserving authentication and key distribution techniques for vehicular ad hoc networks. Clust Comput 12:1–12
47. Wang Z, Cao C, Yang N, Chang V (2017) ABE with improved auxiliary input for big data security. J Comput Syst Sci 89:41–50
48. Yang Y, Zheng X, Chang V, Ye S, Tang C (2017) Lattice assumption based fuzzy information retrieval scheme support multi-user for secure multimedia cloud. Multimedia Tools and Applications 1–15
49. Ye C, Ling H, Zou F, Lu Z, Xiong Z, Zhang K (2013) A novel JFE scheme for social multimedia distribution in compressed domain using SVD and CA. In: The international workshop on digital forensics and watermarking 2012. Springer, Berlin, pp 507–519
50. Ye C, Xiong Z, Ding Y, Wang G, Li J, Zhang K (2014) Joint fingerprinting and encryption in hybrid domains for multimedia sharing in social networks. J Vis Lang Comput 25(6):658–666
51. Zhang J, Wang XA (2012) On the security of a multi-use CCA-secure proxy re-encryption scheme. In: 2012 4th international conference on intelligent networking and collaborative systems (INCoS). IEEE, pp 571–576
52. Zhang J, Wang XA (2012) Security analysis of a multi-use identity based CCA-secure proxy re-encryption scheme. In: 2012 4th international conference on intelligent networking and collaborative systems (INCoS). IEEE, pp 581–586
53. Zhang M, Wang XA, Li W, Yang X (2013) CCA secure publicly verifiable public key encryption without pairings nor random oracle and its applications. JCP 8(8):1987–1994

Xing Hu received the B.S. degree in computer science from Central South University, Changsha, China, in 2003, the M.S. degree in Applied Mathematics from Hunan University of Science and Technology, Xiangtan, China, in 2009, and the Ph.D degree in Applied Mathematics from Guangzhou University, Guangzhou, China, in 2014. Her research interests include cryptography, proxy re-encryption, face recognition, and cloud computing. She is a member of Chinese Association for Cryptologic Research.

Chunming Tang received the B.S. degree in Xiangtan Normal University, China, in 1995, the M.S. degree in Xiangtan University, China, in 2001 and the Ph.D degree in Academy of Mathematics and Systems Science from Chinese Academy of Sciences, China, in 2004. He is currently a Professor of Guangzhou University. His research interests include Cryptography, Information Security, and Cloud Computing. Prof. Tang is a member of Chinese Association for Cryptologic Research.

Duncan S. Wong received the Ph. D. degree in Computer Science from Northeastern University, Boston, MA, USA in 2002. He was an associate Researcher at City University of Hong Kong. His research interests include applied cryptography, in particular, cryptographic protocols, encryption and signature schemes, and anonymous systems.

Xianghan Zheng received the M.S. degree in Distributed System from University of Agder, Norway, in 2007, and the Ph.D degree in Information Communication Technology from University of Agder, Norway, in 2011. His current research interests include New Generation Network with special focus on Cloud Computing Services and Applications, Big Data Processing and Security.