# Non-Transferable Proxy Re-Encryption

Yi-Jun He, L.C.K. Hui, and S.M. Yiu

Department of Computer Science, The University of Hong Kong
Chow Yei Ching Building, Pokfulam Road, Hong Kong
{yjhe,hui,smyiu}@cs.hku.hk

**Abstract.** A proxy re-encryption (PRE) scheme allows a proxy to re-encrypt a ciphertext for Alice (delegator) to a ciphertext for Bob (delegatee) without seeing the underlying plaintext. With the help of the proxy, Alice can delegate the decryption right to any delegatee. However, existing PRE schemes generally suffer from at least one of the followings. Some schemes fail to provide the *non-transferable property* in which the proxy and the delegatee can collude to further delegate the decryption right to anyone. This is the main open problem left for PRE schemes. Other schemes assume the existence of a fully trusted private key generator (PKG) to generate the re-encryption key to be used by the proxy for encrypting a given ciphertext for a target delegatee. But this poses two problems in PRE schemes if the PKG is malicious: the PKG in their schemes may decrypt both original ciphertexts and re-encrypted ciphertexts (referred as the *key escrow* problem); and the PKG can generate re-encryption key for arbitrary delegatees without permission from the delegator (we refer it as the *PKG despotism* problem). In this paper, we proposed the first non-transferable proxy re-encryption scheme which successfully achieves the non-transferable property. We also reduced the full trust in PKG, only a limited amount of trust is placed in the proxy and PKG. We show that the new scheme solved the PKG despotism problem and key escrow problem as well.

**Key words:** proxy re-encryption, identity based encryption, certificateless public key encryption, non-transferable property

## 1 Introduction

### 1.1 Background

In the life, you might have experienced at least one of the following situations: When you are on holiday, but you still want to check emails regularly in order to be aware of some emergency issues. You may think it is easy to solve by checking emails from the mobile phone, or bringing a notebook with you everywhere. But what will you do if you are in a place where you are not convenient to access network or telecommunication, or the network is too slow to login mail server? Now, you may ask a friend to check your emails instead of you. But is it safe to tell others your email password or other confidential information? Don't you

worry about your private information will be leaked to others? You might have also met another situation: You saved some encrypted photos or sensitive files on a file server to facilitate the sharing of them. But decryption keys distributing would be a big problem if you want to share files to a group of friends. Thus, systems such as Cepheus uses a trusted access control server to distribute keys. Group members must contact with access control server to obtain decryption keys for accessing files. However, this solution is not satisfactory, since the access control server model requires a great deal of trust in the server operator. If it is unworthy of this trust, the server operator could abuse the server's key material to decrypt any data stored on the file server. Furthermore, even if the access control server operator is trustworthy, placing so much critical key data in a single location makes for an inviting target.

Luckily, proxy re-encryption [1] is such a desired cryptographic scheme that can perfectly solve those problems mentioned above. It allows a third-party (the proxy) to re-encrypt a ciphertext which has been encrypted for one party without seeing the underlying plaintext so that it can be decrypted by another. This is illustrated in Figure 1 where the Sender encrypts a text for Alice; Alice sends a re-encryption key and the ciphertext to the proxy which performs the re-encryption and sends Bob the re-encrypted ciphertext which can be decrypted by Bob without knowing the secret key of Alice. Re-encryption may be not the only way to transfer the decryption power from a party to another party, but using PRE scheme brings three main advantages: (i) Plaintext is invisible to proxy though it is responsible for doing re-encryption. (ii) Delegator does not need to share his private key with the delegatee. (iii) Delegator does not need to pre-define a decryption key with delegatee. Delegatee just need to use his own private key to decrypt the re-encrypted files. The above scheme aroused much

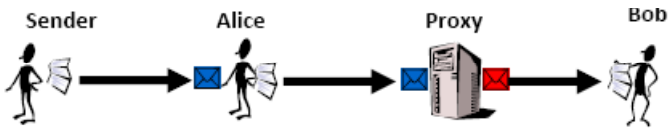


**Fig. 1.** Proxy Re-Encryption

interest in the encryption community [1-6,9-13] since it could be exploited in a number of applications for achieving better information security and privacy, such as:

– Email forwarding: Delegator wishes to delegate his email decryption right to a delegatee. The proxy can "forward" re-encrypted emails to a delegated recipient. The recipient then accesses the emails without needing to know the delegator's decryption key.

- Encrypted files distribution: The encrypted files are stored in a file server. Only the content owners can grant the access right of the files to the target users; even the file server operator has no right to access the files.
- Law-enforcement monitoring: The encrypted communication data is transferred via an Internet service provider (ISP). The ISP can require the content owners to provide the access right to the law enforcement officers to let them monitor the data being transferred to various users; however, the ISP operator cannot access the data.

### 1.2 Review of the Transferable Problem

However, existing PRE schemes (details of existing schemes will be given in the next section) suffer from the same problem of failing to provide the *non-transferable* property which was first introduced by Ateniese *et al.* in 2005 [2]. A proxy re-encryption scheme is said to be non-transferable if the proxy and a set of colluding delegatees cannot re-delegate decryption rights to other parties. On one hand, this is a very desirable property. For example, user A saves some encrypted private confidential files on the file server. If A delegates B the decryption right for accessing those files, A may need some guarantee that his personal information "goes no further". It requires that the delegatee B plus the proxy cannot re-delegate decryption right to others. On the other hand, researchers [2,4] are not sure that transferability can be preventable since the delegatee B can always decrypt and forward the plaintext to another party. However, this approach requires that the delegatee remains an active, online participant. What we want to prevent is the delegatee (plus the proxy) providing other parties with a secret value that it can be used offline to decrypt A's ciphertexts. Again, the delegatee can always send its secret key to another party. But in doing so, the delegatee put itself in a risky situation. Therefore, achieving a non-transferable PRE scheme, in the sense that the only way for delegatee to transfer decryption capabilities to another party is to expose his own secret key, seems to be the main open problem left for PRE.

### 1.3 Limitations of Existing Solutions

Libert and Vergnaud [4] indicated that it is quite difficult to prevent the proxy and delegatees from colluding to do re-delegation and that discouraging collusion rather than preventing illegitimate re-delegation is an easier approach. Thus, they try to trace the malicious proxy after its collusion with one or more delegatees. No doubt that it works to deter collusion from happening. However, it is more desirable to have a better way to prevent collusion, not just discourage collision. Some identity-based PRE schemes assume the existence of a fully trusted private key generator (PKG) which helps to generate the re-encryption key to be used by the proxy for encrypting a given ciphertext for a target delegatee. Since the re-encryption key is generated using the master key of the PKG, the proxy and the delegatee(s) cannot further delegate the decryption right to others without the help of the PKG. However, this creates two problems

in PRE schemes. First, there is another key escrow problem for which the PKG in their schemes may be able to decrypt both original and re-encrypted ciphertexts. And the PKG despotism problem, in which the PKG has the power of generating re-encryption key for arbitrary delegatees. Thus those PRE schemes involving PKG just transform "delegatee-proxy-collusion transferable problem" to "PKG alone transferable problem". So it is fair to say that they did not solve the open problem related to the non-transferability issue.

### 1.4 Our Contributions

To tackle the transferable problem as well as the key escrow problem and PKG despotism problem, a new PRE model is built based on certificateless encryption. We still borrow the idea of using PKG to generate a re-encryption key, but our new non-transferable re-encryption scheme successfully solved the problems in previous PKG-based works. The characteristics of our proposed scheme are summarized as follow.

- The proposed scheme has the non-transferable property. The re-encryption key is generated by a key generating centre (PKG); Delegator participants actively to help generating decryption key for delegatee using part of his private key. Thus delegatee and proxy cannot collude to re-delegate decryption rights since they do not have knowledge of PKG's master secret and the delegator's private key.
- Without the participation of the delegator, PKG is unable to generate any useful re-encryption key for delegating decryption right, thus completely resolves the PKG despotism problem.
- PKG cannot decrypt the original ciphertext and re-encrypted ciphertexts as well, thus solving the key escrow problem.

## 2 Related Work

Blaze, Bleumer and Strauss [1] proposed the first proxy re-encryption scheme, which is based on ELGamal encryption. But this scheme is bi-directional, that is, when the proxy is allowed to re-encrypt Alice's messages under Bob's key, it can also re-encrypt Bob's messages under Alice's key. Bob may not like this. Another weakness is that if the proxy colludes with Alice, they can easily learn Bob's secret key $SK_B$. Likewise, the proxy and Bob may collude to learn Alice's secret key. Furthermore, in order to compute the re-encryption key from $A$ to $B$, denoted as $rk_{A->B}$, one party must share his or her secret key with the other or they must rely on a trusted third party. The other drawback is that the scheme is transitive in the following sense. Suppose that the proxy is allowed to generate two re-encryption keys $rk_{A->B}$ and $rk_{B->C}$; then the proxy can derive an additional re-encryption key $rk_{A->C}$ for delegation from $A$ to $C$.

Later, Ivan and Dodis [3] proposed three unidirectional proxy re-encryption schemes based on ElGamal, RSA, and IBE (ID-based encryption) respectively.

Their main contribution is that they solved (i) the bi-directional problem and (ii) the transitive problem in [1]. But in their schemes, Alice's private key is split into two parts $DK_1$ and $DK_2$, with $DK_1$ distributed to proxy and $DK_2$ distributed to Bob. Thus when the proxy colludes with Bob, they can derive Alice's private key.

In 2005, Ateniese *et al.* [2] presented three proxy re-encryption schemes which are considered to be more secure than other approaches. Their major advantages are the following. The schemes are unidirectional and the delegator's private key is protected from being disclosed by the collusion of proxy and a delegatee. They implemented one of their proposed schemes in a secure distributed file system to show that the scheme can work efficiently in practice. They summarized nine important properties of proxy re-encryption schemes, which include the non-transferable property. Lacking the non-transferable property in all existing schemes was considered an open problem of the contemporary PRE schemes.

This open problem was first addressed in 2008 by Libert and Vergnaud [4]. They indicated that it is quite difficult to prevent the proxy and delegatees from colluding to do re-delegation and that discouraging collusion rather than preventing illegitimate re-delegation is an easier approach. Thus, they proposed, instead of preventing the collusion of proxy and delegatee, tracing the malicious proxy after its collusion with one or more delegatees. It is the first attempt to address the open problem. However, it still cannot prevent re-delegation from happening.

Matsuo's PRE schemes [5] use the PKG to help generating re-encryption key for the delegator and the delegatee. Based on this approach, they proposed two PRE schemes: one for the decryption right delegation from a user of PKI-based public key encryption system to IBE system users, and the other for the delegation among IBE system users. This is the first set of schemes that use PKG to generate re-encryption key. However, the PKG in the schemes can decrypt all re-encrypted ciphertexts; so, there is a potential security problem as long as PKG is untrusted or malicious.

In 2008, Wang *et al.*[6] extended the idea of Matsuo's scheme by allowing PKG to generate re-encryption keys based on its master secret key. They proposed several proxy re-encryption schemes:(i) PRE from IBE to Certificate Based Public Key Encryption; (ii) PRE based on a variant of the first system of Selective identity secure IBE [7]; (iii) PRE based on the second system of Selective identity secure IBE [7];and (iv) PRE based on Sakai-Kasahara IBE scheme [8]. Based on this work, Wang *et al.* proposed five other schemes [9-13] to address different problems of proxy re-encryption schemes. However, there are still some issues not yet addressed in each one of them. In [9], the proxy can re-encrypt on its own the ciphertext for the delegator into ciphertext for any delegatee; this is not a desired property of PRE. In [10], it seems that they solved the open problem related to the non-transferable issue, since proxy and delegate cannot collude to re-delegate decryption right; however, in the scheme, the PKG alone can delegate arbitrarily to anyone as it can generate a re-encryption key for any delegatee. In [11,13], the PKG can also delegate arbitrarily as what it could

do in [10]. Among the five schemes, [10] seems to be the best in solving the non-transferable issue, we will compare our scheme with [10] in Section 5.2.

## 3 Preliminaries

### 3.1 Bilinear Map

Let $G$ and $G_T$ be multiplicative cyclic groups of prime order $p$, and $g$ be generator of $G$. We say that $G_T$ has an admissible bilinear map $e$: $G \times G \rightarrow G_T$, if the following conditions hold.

- $e(g^a, g^b) = e(g, g)^{ab}$ for all $a$, $b$.
- $e(g, g) \neq 1$.
- There is an efficient algorithm to compute $e(g^a, g^b)$ for all $a$, $b$ and $g$.

### 3.2 Assumption

The security of our concrete construction is based on a complexity assumption, called "Truncated Decision Augmented Bilinear Diffie-Hellman Exponent Assumption (Truncated *q-ABDHE*)"proposed in [14].

Let $e : G \times G \rightarrow G_T$ be a bilinear map, where $G$ and $G_T$ are cyclic groups of large prime order $p$. Given a vector of $q$+3 elements:

$$(g', g'^{(\alpha^{q+2})}, g, g^{\alpha}, \ldots, g^{(\alpha^q)}) \in G^{q+3}$$

and an element $Z \in G_T$ as input, output 0 if $Z = e(g^{(\alpha^{q+1})}, g')$ and output 1 otherwise.

An algorithm $\mathcal{B}$ has advantage $\varepsilon$ in solving the truncated *q-ABDHE* if:

$$|\Pr[\mathcal{B}(g', g'^{(\alpha^{q+2})}, g, g^{\alpha}, \ldots, g^{(\alpha^q)}, e(g^{(\alpha^{q+1})}, g')) = 0]$$

$$-\Pr[\mathcal{B}(g', g'^{(\alpha^{q+2})}, g, g^{\alpha}, \ldots, g^{(\alpha^q)}, Z) = 0]| \geq \varepsilon$$

where the probability is over the random choice of generators $g$, $g'$ in $G$, the random choice of $\alpha$ in $Z_p$, the random choice of $Z \in G_T$, and the random bits consumed by $\mathcal{B}$.

*Definition 1 [14].We say that the truncated (decision) (t, ε, q)-ABDHE assumption holds in G if no t-time algorithm has advantage at least ε in solving the truncated (decision) q-ABDHE problem in G.*

## 4 Our Non-Transferable PRE Scheme

### 4.1 Non-Transferable PRE Model

Our Non-Transferable PRE scheme is based on certificateless encryption. It is composed of nine algorithms:

- Setup. On input a security parameter $1^k$, the public parameters $mpk$ and master secret key $msk$ are generated.
- Key Generation.
  •Set-Secret-Value. algorithm generates a secret value which is only known to user himself.
  •Partial-Private-Key-Extract. On input a user's identity $ID$, $msk$, algorithm generates partial private key for user.
  •Set-Private-Key. On input the partial private key and the secret value, algorithm outputs the whole private key for user.
  •Set-Public-Key. On input a user's identity $ID$ and secret value, algorithm generates public key.
- Private key Correctness Check. Algorithm checks the correctness of the private key.
- Encryption. The encryption algorithm takes public key $upk_i$ of delegator $i$ and message $m$ as input, outputs a ciphertext $C_i$ encrypted under $upk_i$.
- Decryption(delegator). The decryption algorithm takes private key $usk_i$ of delegator $i$ and ciphertext $C_i$ as input, outputs message $m$. This algorithm actually is not necessary for PRE scheme. We put it here just for indicating that delegator has the ability to decrypt the original ciphertext $C_i$.
- Re-Encryption Key Generation. Algorithm verifies the delegator $i$'s signature, and extracts delegatee $j$'s $ID$ from signature. The re-encryption key generation algorithm outputs a re-encryption key $rk_{i\to j}$ and other relational values.
- Partial-Decryption-Key Generation. Algorithm checks the correctness of the re-encryption key, and generates a partial decryption key.
- Re-Encryption. The re-encryption algorithm takes re-encryption key $rk_{i\to j}$ and ciphertext $C_i$ as input, outputs a re-encrypted ciphertext $C_j$ under $upk_j$.
- Decryption(delegatee). The decryption algorithm takes private key $usk_j$ of delegatee $j$, partial decryption key and ciphertext $C_j$ as input, outputs message $m$.

### 4.2 Non-Transferable PRE Scheme Construction

We construct the Non-Transferable PRE scheme based on the basic IBE system proposed in [15]. However, the IBE system in [15] cannot fully satisfy our security requirement. We transformed this IBE system into a certificateless public key encryption system [16], so that our PRE scheme based on this new certificateless encryption system can successfully solve the transferable problem in existing PRE schemes. The main ideas of the scheme are as follow: Before delegation, delegator will send delegatee's identity to PKG. PKG is responsible for generating the re-encryption key, and sending this key and some other information to delegator. Delegator checks the correctness of the re-encryption key, and generates a partial decryption key making use of the information received from PKG. Then, delegator sends the re-encryption key to the proxy, and the partial decryption key to delegatee. The proxy re-encrypts the original ciphertext from delegator, and sends the re-encrypted ciphertext to delegatee. The delegatee can decrypt

the ciphertext using his private key and the partial decryption key received from delegator.

In the following sections, we let Alice ($A$) be the delegator, and Bob ($B$) be the delegatee.

***Setup***:

Let $G$ and $G_T$ be groups of order $p$ such that $p$ is a $k$-bit prime, and let $e : G \times G \to G_T$ be the bilinear map. $H_I$:$\{0,1\}^* \to \mathrm{Z}_p$, $H$:$\{0,1\}^* \to \mathrm{Z}_p$ are secure hash functions. The PKG selects four random generators $h_1$, $h_2$, $h_3$, $g \in G$ and randomly chooses $\alpha \in Z_p$. It sets $g_1 = g^\alpha$. Define the message space $\mathcal{M} = G_T$. The public parameters *mpk* and master secret key *msk* are given by

$$mpk = (g, g_1, h_1, h_2, h_3, H_I, H, \mathcal{M}), msk = (\alpha)$$

***Key Generation***:

This is a protocol through which a user $U$ with an identity $ID$ can securely get his partial private key from PKG.

On input the public key/master secret key pair (*mpk*, *msk*) and an identity $ID_A \in \{0,1\}^k$ of entity $A$, the PKG computes $id_A = H_I(ID_A)$. If $id_A = \alpha$ , it aborts. Otherwise, the protocol proceeds as follow:

- Set-Secret-Value. Entity $A$ selects $r_A \in Z_p$ at random. $r_A$ is $A$'s secret value.
- Partial-Private-Key-Extract.
  1. $A$ sends $R = {h_1}^{r_A}$ to PKG, and gives PKG the following zero-knowledge proof of knowledge:
  $$PK\{r_A : R = {h_1}^{r_A}\}$$
  2. PKG randomly selects $r'_A$, $r_{\mathrm{A},2}$, $r_{\mathrm{A},3} \in Z_p$ and computes $h'_A = (Rg^{-r'_A})^{1/(\alpha - id_A)}$, $h_{A,2} = (h_2 g^{-r_{A,2}})^{1/(\alpha - id_A)}$, $h_{A,3} = (h_3 g^{-r_{A,3}})^{1/(\alpha - id_A)}$ and sends $A$'s partial private key ($r'_A$, $h'_A$, $r_{A,2}$, $h_{A,2}$, $r_{A,3}$, $h_{A,3}$) to $A$.
- Set-Private-Key. $A$ computes

  $$r_{A,1} = r'_A / r_A, h_{A,1} = (h'_A)^{1/r_A} = (h_1 g^{-r_{A,1}})^{1/(\alpha - id)}$$

  Then, A's private key can be denoted as

  $$usk_A = (r_A, r_{A,1} = r'_A/r_A, h_{A,1} = (h_1 g^{-r_{A,1}})^{1/(\alpha - id_A)}, r_{A,2}, h_{A,2}, r_{A,3}, h_{A,3})$$

  and the delegatee $B$'s private key is denoted as

  $$usk_B = (r_B, r_{B,1} = r'_B/r_B, h_{B,1} = (h_1 g^{-r_{B,1}})^{1/(\alpha - id_B)}, r_{B,2}, h_{B,2}, r_{B,3}, h_{B,3})$$

- Set-Public-Key. $A$ publishes her public key $upk_A = (p_{A,1}, p_{A,2})$, where $p_{A,1} = {g_1}^{r_A}$, and $p_{A,2} = (g^{r_A})^{id_A}$.

***Private key Correctness Check***:

On input $(mpk, usk_{ID})$ and an identity $ID \in \{0,1\}^k$, A computes $id_A = H_I(ID_A)$ and checks whether $e(h_{A,i}, g_1/g^{id_A}) = e(h_1 g^{-r_{\mathrm{A},i}}, g)$ for $i$=1,2,3. If correct, output 1. Otherwise, output 0.

***Encryption***:

To encrypt a message $m \in G_T$ using public key, sender checks that the equality $e(g^{id_A}, p_{A,1})$=$e(g_1, p_{A,2})$ holds. If not, output $\perp$ and abort encryption. Otherwise, sender generates a unique randomly-selected secret parameter $s \in Z_p$, and computes $id_A = H_I(ID_A)$. Finally, sender outputs the ciphertext $C$ where:

$$C = (C_1, C_2, C_3, C_4) = ({p_{A,1}}^s {p_{A,2}}^{-s}, e(g,g)^s, m \cdot e(g,h_1)^{-s}, e(g,h_2)^s e(g,h_3)^{s\beta})$$

We set $\beta = H(C_1, C_2, C_3)$.

***Decryption(delegator)***:

To decrypt a ciphertext $C = (C_1, C_2, C_3, C_4)$ using secret key $usk_A$, delegator Alice computes $\beta = H(C_1, C_2, C_3)$ and tests whether

$$C_4 = e(C_1, h_{A,2} {h_{A,3}}^{\beta})^{1/r_A} \cdot {C_2}^{r_{A,2} + r_{A,3}\beta}$$

If it is not equal, output $\perp$. Else output

$$m = C_3 \cdot e\left(C_1, h_{A,1}\right)^{1/r_A} \cdot {C_2}^{r_{A,1}}$$

The following Re-Encryption process is done through an interactive protocol among Alice, Bob, PKG and Proxy, which is shown in Figure 2.

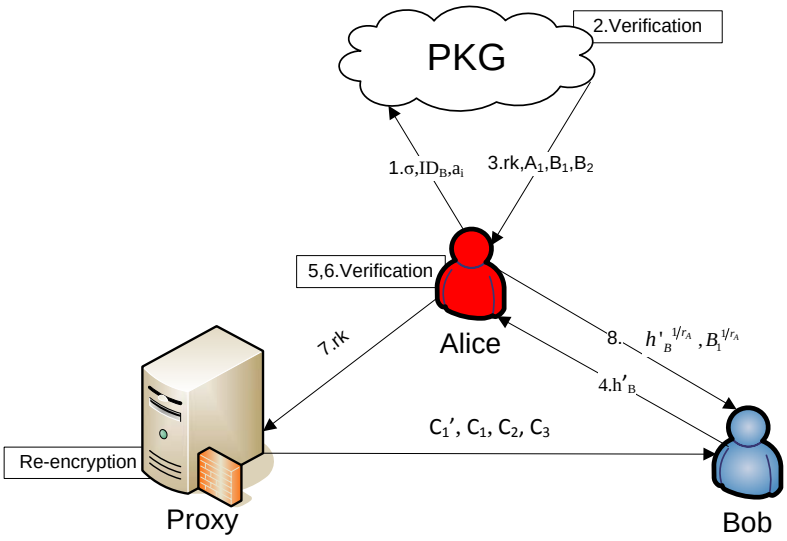


**Fig. 2.** Proposed Non-Transferable Proxy Re-encryption framework

***Re-Encryption Key Generation***:

1. The delegator Alice generates a random value $a_i \in Z_p$ for each time period $i$, where $i \geq 1$. $a_i$ will be invalid after the period $i$. Alice signs Bob's identity $ID_B$, and sends the signature $\sigma, ID_B, a_i$ to PKG via a secure channel.
   Delegator Sign:
   - Choose $z \in Z_p$, and compute $U = g^z$.
   - Compute $V = H_I(ID_B, U)$.
   - Compute $W = g^{\alpha r_A + V}$.
   - The signature on $ID_B$ is $\sigma = (U, W)$.
2. PKG verifies Alice to identify the identity of the delegator.
   PKG Verify:
   - Compute $V = H_I(ID_B, U)$.
   - Accept the signature iff $e(h_1, W) = e({h_1}^{r_A}, g^{\alpha}) e(h_1, g)^V$.
3. If verification passes, PKG generates a unique randomly-selected secret parameter $y \in Z_p$, and computes re-encryption key $rk_{A\to B} = (\frac{\alpha - id_B}{\alpha - id_A} + a_i y) \bmod p, A_1 = \left({h_1}^{r_A} g^{-r'_A}\right)^y, B_1 = \left({h_1}^{r_B} g^{-r'_B}\right)^{a_i y/(\alpha - id_B)}, B_2 = {h_1}^{a_i y}$ and sends $rk_{A\to B}$, $A_1$, $B_1$, $B_2$ to Alice.

***Partial-Decryption-Key Generation:***

4. Delegatee Bob sends $h'_B$ to Alice via a secure and authenticated channel.
5. Alice checks whether
$$e(h_1, B_1) = e(B_2, h'_B)$$
to ensure $B_1$ is a valid value which will help delegatee for decryption later. If correct, output 1, otherwise, output 0.
6. Alice checks whether
$${h'_A}^{(id_A - id_B)} \cdot {A_1}^{a_i} \cdot \left({h_1}^{r_A} g^{-r'_A}\right) = \left({h_1}^{r_A} g^{-r'_A}\right)^{rk_{A\to B}}$$
to ensure that $rk_{A\to B}$ is a re-encryption key generated properly for delegation from her to Bob.
7. Alice sends the re-encryption key $rk_{A\to B}$ to Proxy via an authenticated channel.
8. Alice computes ${h'_B}^{1/r_A}$ and ${B_1}^{1/r_A}$, and sends them to Bob as partial decryption key.

***Re-Encryption:***

Proxy computes ${C_1}' = {C_1}^{rk_{A\to B}} = g^{r_A s(\alpha - id_A)(\frac{\alpha - id_B}{\alpha - id_A} + a_i y)}$, and sends $({C_1}', C_1, C_2, C_3)$ to Bob.

***Decryption (delegatee)***:

Bob computes

$$C_3 \frac{e({C_1}', {h'_B}^{(1/r_A)(1/r_B)}) {C_2}^{r_{B,1}}}{e(C_1, {B_1}^{(1/r_A)(1/r_B)})}$$
$$= m \cdot e(g, h_1)^{-s} \frac{e(g^{r_A s(\alpha - id_A)(\frac{\alpha - id_B}{\alpha - id_A} + a_i y)}, (h_1 g^{-r_{B,1}})^{\frac{1}{(\alpha - id_B) r_A}}) (e(g,g)^s)^{r_{B,1}}}{e(g^{r_A s(\alpha - id_A)}, (h_1 g^{-r_{B,1}})^{\frac{a_i y}{(\alpha - id_B) r_A}})}$$

$= m \cdot e(g, h_1)^{-s} e(g^{s(\alpha - id_A)(\frac{\alpha - id_B}{\alpha - id_A})}, (h_1 g^{-r_{B,1}})^{\frac{1}{(\alpha - id_B)}}) e(g, g)^{s * r_{B,1}}$
$= m \cdot e(g, h_1)^{-s} e(g^{s(\alpha - id_B)}, (h_1 g^{-r_{B,1}})^{\frac{1}{(\alpha - id_B)}}) e(g, g)^{s r_{B,1}}$
$= m$

## 5 Security Analysis

The main advantage of our scheme is: It achieved Non-transferable property, Non-Key-escrow property and Non-PKG-despotism property, in which Non-Key-escrow property and Non-PKG-despotism property are defined by us especially for estimating security of a PKG involved PRE schemes. To prove that our scheme is able to achieve Non-transferable property, we construct a possible attack, and demonstrate how our scheme can resist to this attack.

- Non-transferable: In PRE, the proxy and a set of colluding delegatees cannot re-delegate decryption rights. For example, from $rk_{A \to B}$, $sk_B$ and $pk_C$, they cannot produce $rk_{A \to C}$. Now go back to our scheme for a concrete discussion. After one delegation, the proxy holds $rk_{A \to B}$, and the delegatee Bob holds $(r_B, r_{B,1})$ and $(DK_0 = {h'_B}^{1/r_A}, DK_1 = B_1^{1/r_A})$. If proxy and Bob want to collude to re-delegate the decryption right to others, Bob may compute $DK_2 = DK_0^{1/r_B}$ and $DK_3 = DK_1^{1/r_B}$ by himself, then the PKG exponentiate the $DK_2$ by $rk_{A \to B}$. Given $(C_1, C_2, C_3)$ which is the original ciphertext for the delegator, anyone who holds $(DK_2, DK_3, r_{B,1})$ can decrypt by $C_3 * e(C_1, DK_2) * C_2^{r_{B,1}} / e(C_1, DK_3)$. However, notice that whatever method (2-party computation, or oblivious computation) the proxy and delegatee used to compute the $DK_2, DK_3$, this re-delegation will success only when Bob wishes to send his secret key $r_{B,1}$ explicitly to other parties, because $r_{B,1}$ must be used to exponentiate $C_2$ for decrypting the ciphertext. Further, $C_2$ is changable in each delegation due to the random number $s$, so $C_2^{r_{B,1}}$ cannot be pre-computed, and $r_{B,1}$ must be sent explicitly to other parties. But by doing so, he may put himself in danger, because his private key would be known to PKG once other parties report $r_{B,1}$ to PKG. Thus we achieved the purpose of preventing delegatee from colluding with proxy to re-delegate the decryption right. Non-transferable property is achieved in our scheme.
- *Non-Key-escrow*: In PRE, PKG should not be allowed to decrypt both original ciphertext and re-encrypted ciphertext for anyone, this is called Non-Key-escrow. Most of PKG-based PRE schemes do not achieve this property. However, in our proposed scheme, the original ciphertext is decrypted by $m = C_3 \cdot e\left(C_1, h_{A,1}\right)^{1/r_A} \cdot {C_2}^{r_{A,1}}$. $r_A$ is needed for decryption. Moreover, the re-encrypted ciphertext is decrypted by $C_3 \frac{e(C_1', {h'_B}^{(1/r_A)(1/r_B)}) {C_2}^{r_{B,1}}}{e(C_1, {B_1}^{(1/r_A)(1/r_B)})}$. $r_B$ is needed for decryption. Notice that $r_A$ is the secret value of delegator, and $r_B$ is the secret value of delegatee. PKG holds neither $r_A$ nor $r_B$. Thus only users themselves can decrypt the ciphertext, not the PKG. Non-Key-escrow property is achieved in our scheme.

- *Non-PKG-despotism*: In PRE, PKG is not allowed to generate a proper re-encryption key arbitrarily for delegating decryption right without permission from delegator, this is called Non-PKG-despotism. In our proposed scheme, delegator participants actively to help PKG generating re-encryption key by sending the random value $a_i$, and the validity of the re-encryption key can be verified by delegator by checking if ${h'_A}^{(id_A - id_B)} \cdot {A_1}^{a_i} \cdot \left({h_1}^{r_A} g^{-r'_A}\right) = \left({h_1}^{r_A} g^{-r'_A}\right)^{rk_{A \to B}}$. Further, delegator is responsible for generating a partial decryption key $({h'_B}^{1/r_A}, B_1^{1/r_A})$ for delegatee using his own secret value $r_A$. Without the participation of the delegator, delegatee is unable to decrypt the re-encrypted ciphertext. In another word, the re-encryption key generated by PKG alone is useless unless delegator is willing to help. Thus completely resolves the PKG despotism problem.

To compare some existing proxy re-encryption schemes with our proposed scheme as fully as possible, we also analyze below some important properties defined in [2]. The comparison results are presented in Table 1.

- *Unidirectional*: Delegation from $A \to B$ does not allow re-encryption from $B \to A$.
- *Non-interactive*: Re-encryption keys can be generated by Alice using Bob's public key; no trusted third party or interaction is required. In our proposed scheme, PKG is employed to generate re-encryption keys, so delegator needs to interact with PKG to generate the keys.
- *Proxy transparent*: This is an important feature possessed by the original BBS scheme [1]. The proxy in the BBS scheme is *transparent* in the sense that neither the sender of an encrypted message nor any of the delegatees has to be aware of the existence of the proxy. In BBS scheme, this property is achieved at the price of allowing transitive delegation and recovery of the master secrets of the delegator. In Ateniese's scheme, only a weaker form of proxy transparency, called *proxy invisibility* can be achieved, because the sender needs to know the existence of proxy, in order to decide whether to generate first-level encryption or second-level encryption. In our proposed scheme, proxy is transparent. Both sender and delegatees do not have to know the proxy, since there is only one form of encryption.
- *Original-access*: Alice can decrypt re-encrypted ciphertexts that were originally sent to her.
- *Key optimal*: The size of Bob's secret storage remains constant, regardless of how many delegations he accepts. Like Ateniese's scheme, delegatee is allowed to decrypt re-encrypted ciphertext during some specific time period $i$. Thus information received from PKG for decryption only need to exist temporarily in delegatee's side. After a time period $i$, the information would be invalid. Delegatee can delete the information immediately. Thus in the long run, our scheme is still key optimal.
- *Collusion-"safe"*: Bob and the proxy's collusion cannot recover Alice's secret key. In our proposed scheme, secrecy of Alice's secret key depends on a

random value $r_A$. It is chosen by Alice, and is not used in re-encryption key. Although Bob and proxy collude, they cannot recover it.

- *Temporary*: Bob is only able to decrypt messages intended for Alice that were authored during some specific time period $i$. In our scheme, to achieve temporary proxy re-encryption, for each time period $i \geq 1$, Alice generates a random value $a_i \in Z_p$. Because $a_i$ will be invalid after time period $i$, the re-encryption key's life cycle is also period $i$. We remark that in all existing schemes including our scheme, the temporary property is achieved based on the assumption that the proxy will update the re-encryption key after each period expires.
- *Non-transitive*: Based on the re-encryption keys, $rk_{A\to B}$ and $rk_{B\to C}$, the proxy cannot produce $rk_{A\to C}$. In our proposed scheme, the re-encryption key is generated using the master secret key of the PKG, proxy cannot generate $rk_{A\to C}$ without knowing the master secret key. And the delegatee's identity is included in the re-encryption key, the proxy is unable to replace the delegatee with another party. So even with the keys $rk_{A\to B}$ and $rk_{B\to C}$, the proxy cannot produce $rk_{A\to C}$.

**Table 1.** Comparison of existing PRE schemes and our proposed scheme

| Property | BBS [1] | ID [3] | Ateniese[2] | Wang[10] | Our Scheme |
|---|---|---|---|---|---|
| Unidirectional | No | Yes | Yes | Yes | Yes |
| Non-interactive | No | Yes | Yes | No | No |
| Proxy transparent | Yes | No | Yes$^{\#}$ | Yes | Yes |
| Original-access | Yes | Yes | Yes | No | Yes |
| Key optimal | Yes | No | Yes | Yes | Yes |
| Collusion-safe | No | No | Yes | Yes | Yes |
| Temporary | Yes | Yes | Yes | No | Yes |
| Non-transitive | No | Yes | Yes | Yes | Yes |
| Non-transferable | No | No | No | No* | Yes |
| Non-Key escrow | −− | No | −− | No | Yes |
| Non-PKG despotism | −− | No | −− | No | Yes |

(∗) PKG alone can transfer
($^{\#}$) can only achieve proxy invisibility which is a weaker form of proxy transparent

## 6 Conclusions

In this paper, we attempt to solve the open problem pointed out in 2005, in proposing a non-transferable proxy re-encryption scheme. With the proposed PRE scheme, the proxy and a delegatee cannot collude to transfer decryption rights. We also introduced two important properties, namely *Non-Key-escrow* and *Non-PKG-despotism*, into the proposed PRE scheme. The principle behind

our solution is that instead of 'prohibiting' a party to propagate information, we punish the party who illegitimately propagates information by exposing the important secrets of the party. This method is feasible due to the fact that nobody would run the risk of exposing its own secrets to do illegal decryption right transfer. Thus, our 'punish' method is more practicable and effective than the 'tracing' method in [4], because it can strongly prevent illegal decryption right transfer from happening, but not just tracing the malicious proxy after the illegal decryption right transfer. To the best of our knowledge, our paper is the first paper which completely solves the transferable problem.

**Acknowledgments.** We would like to show my deepest gratitude to Sherman S.M. Chow, for all his kindness and help. Without his valuable comment, we could not have solved the difficult part of this paper.

## References

1. M. Blaze, G. Bleumer, and M. Strauss.: Divertible protocols and atomic proxy cryptography. In: EUROCRYPT 1998, volume 1403 of LNCS, pp. 127-144, (1998)
2. G. Ateniese, K. Fu, M. Green, S. Hohenberger.: Improved Proxy Re-encryption Schemes with Applications to Secure Distributed Storage. In: 12th Annual Network and Distributed Systems Security Symposium, San Diego, California (2005)
3. A. Ivan and Y. Dodis.: Proxy cryptography revisited. In: 10th Annual Network and Distributed Systems Security Symposium, (2003)
4. Benoit Libert, Damien Vergnaud.: Tracing Malicious Proxies in Proxy Re-Encryption. In: 2nd international conference on Pairing-Based Cryptography, Egham, UK (2008)
5. T. Matsuo.: Proxy Re-encryption Systems for Identity-Based Encryption. In: 1st International Conference on Pairing-Based Cryptography - Pairing 2007, LNCS 4575, pp. 247-267. Springer-Verlag (2007)
6. X.A. Wang, X.Y. Yang, F.G. Li.: On the Role of PKG for Proxy Re-encryption in Identity Based Setting. In: Cryptology ePrint Archive, Report 2008/410. (2008)
7. D. Boneh, X.Boyen.: Efficient Selective-id Secure Identity Based Encryption without Random Oracles. In: Advances in Cryptology-EUROCRYPT 2004. LNCS 3027, pp. 223-238. Springer, (2004)
8. R. Sakai, M. Kasahara.: ID based cryptosystems with pairing on elliptic curve. Cryptology ePrint Archive, Report 2003/054. (2003)
9. X.A. Wang, X.Y. Yang.: Identity based broadcast encryption based on one to many identity based proxy re-encryption. In 2nd IEEE International Conference on Computer Science and Information Technology, pp.47-50, (2009)
10. X.A. Wang, X.Y. Yang.: Proxy re-encryption scheme based on BB2 identity based encryption. In: 2nd IEEE International Conference on Computer Science and Information Technology, pp.134-137, (2009)
11. X.A. Wang, X.Y. Yang.: Proxy Re-encryption Scheme Based on SK Identity Based Encryption. In: 5th International Conference on Information Assurance and Security. pp.657-660, (2009)
12. K. Niu, X.A. Wang, M.Q. Zhang.: How to Solve Key Escrow Problem in Proxy Re-encryption from CBE to IBE. In: 1st International Workshop on Database Technology and Applications. pp.95-98, (2009)

13. X.A. Wang, X.Y. Yang, M.Q. Zhang.: Proxy Re-encryption Scheme from IBE to CBE," 1st International Workshop on Database Technology and Applications. pp.99-102, (2009)
14. Gentry, C.: Practical identity-based encryption without random oracles. In: EUROCRYPT 2006. LNCS, vol. 4004, pp. 445-464. Springer, Heidelberg (2006)
15. Goyal, V.: Reducing trust in the PKG in identity based cryptosystems. In: Menezes, A. CRYPTO 2007. LNCS, vol. 4622, pp. 430-448. Springer, Heidelberg (2007)
16. Sattam S. Al-Riyami and Kenneth G.: Paterson, Certificateless Public Key Cryptography. In: Lecture Notes in Computer Science, pp. 452 - 473, (2003)