Inter-domain Identity-Based Proxy Re-encryption

Qiang Tang[1], Pieter Hartel[1], and Willem Jonker[1,2]

[1] Faculty of EWI, University of Twente, the Netherlands
[2] Philips Research, The Netherlands

Abstract. Proxy re-encryption is a cryptographic primitive developed to delegate the decryption right from one party (the delegator) to another (the delegatee). So far, research efforts have only been devoted to the intra-domain setting, where the delegator and the delegatee are registered in the same domain. In this paper, we investigate the proxy re-encryption in the inter-domain setting, where the delegator and the delegatee are from different domains, and focus on the identity-based case. We analyze the trust relationships and possible threats to the plaintext privacy, and provide rigorous security definitions. We propose a new inter-domain identity-based proxy re-encryption scheme and prove its security in our security model. An interesting property of the proposed scheme is that, to achieve the chosen plaintext security for the delegator, the delegatee's IBE only needs to be one-way.

1 Introduction

Mambo and Okamoto [10] firstly propose the concept of delegation of decryption right in the context of speeding up decryption operations. Blaze, Bleumer and Strauss [2] introduce the concept of atomic proxy cryptography which is proxy re-encryption. In a proxy re-encryption scheme, a delegator (say, Alice) and a delegatee (say, Bob) generate a proxy key that allows a semi-trusted third party (say, the proxy) to convert ciphertexts encrypted for Alice into ciphertexts which can be decrypted by Bob. Blaze, Bleumer and Strauss present a proxy re-encryption scheme based on Elgamal [6]. In their scheme, the proxy is also capable of converting ciphertexts encrypted for Bob into ciphertexts which can be decrypted by Alice. Jakobsson [9] and Zhou *et al.* [16] simultaneously propose quorum-based protocols, which divide the proxy into many components. Dodis and Ivan [8] propose generic constructions of proxy re-encryption schemes by using double-encryption. Ateniese *et al.* [1] propose an Elgamal-based scheme and show its application in securing file systems. In addition, Ateniese *et al.* also point out a number of desirable properties for proxy re-encryption schemes. Note that these papers mainly focus on the traditional public-key encryption schemes.

Since Shamir [12] firstly propose the concept, Identity-Based Encryption (IBE) has become a powerful tool in both theoretical cryptography and practical applications, especially after the work by Boneh and Franklin [4]. Considering

M. Yung, P. Liu, and D. Lin (Eds.): Inscrypt 2008, LNCS 5487, pp. 332–347, 2009.
© Springer-Verlag Berlin Heidelberg 2009

their usefulness, it is interesting to extend the concept of proxy re-encryption into the identity-based setting, i.e. ID-based proxy re-encryption. Until now, apart from the generic construction of Dodis and Ivan [8], there are two identity-based proxy re-encryption schemes, in which the delegator and the delegatee are registered in the same domain. One is proposed by Green and Ateniese [7] and the other is proposed by Matsuo [11]. In both schemes, the delegator and the delegatee are assumed to be registered at the same domain (or, the same Key Generation Center (KGC)).

1.1 Motivation and Contribution

Proxy re-encryption has many promising applications including access control in file storage [1], email forwarding [15], and law enforcement [8]. For many cases of these applications, it would be more reasonable to assume an inter-domain setting where the delegator and the delegatee are from different domains than a intra-domain setting where all parties are from the same domain.

For example, Alice from university A (Alice's domain) might want to exploit an ID-based proxy re-encryption scheme so that messages encrypted under her identifier can be "automatically" converted into ciphertexts for her friend Bob from company B (Bob's domain). In this example, it is unrealistic to assume that university A and company B share the same KGC.

In the inter-domain setting, existing ID-based proxy re-encryption schemes (e.g. those in [7,11]) cannot be used because the delegator and the delegatee are required to be registered at the same domain. To our knowledge, no particular research efforts have been devoted to proxy re-encryption in the inter-domain ID-based setting.

In this paper, we analyze the trust relationships and possible threats to the plaintexts of both the delegator and the delegatee for proxy re-encryption in the inter-domain setting, and provide rigorous security definitions. Compared with the intra-domain formulations [7,11], our formulation of inter-domain proxy re-encryption has the following differences.

1. In our case, the delegator and the delegatee are from different domains, while they are assumed to be from the same domain in previous formulations in [7,11].
2. In our model, the proxy key can be generated by either the delegator himself (the case in [7]) or the delegator together with the delegatee and even the KGCs. We believe this general assumption is more realistic in practice than that in [7].
3. As a result of the above assumption, the proxy key might leak some information about the delegatee's private key, hence, we have also taken into account the semantic security for the delegatee. This security formulation is necessary because the delegatee's IBE key might also be used for normal IBE services.

4. With respect to the definition of CPA security for the delegator, we have taken into account an ignored fact by previous works, i.e. a curious delegatee naturally has access to the plaintexts which have been re-encrypted by the proxy.

We propose a new inter-domain identity-based proxy re-encryption scheme by extending the concept of the Green-Ateniese proxy re-encryption scheme IBP1 [7]. Given that the delegatee's IBE is IND-CPA secure, our scheme is secure against a chosen plaintext attack for the delegatee (IND-CPA secure). Given that the delegatee's IBE is one-way, we show that our scheme is secure against a chosen plaintext attack for the delegator (IND-CPA secure) based on the decision BDH assumption in the random oracle model. Interestingly, to achieve the chosen plaintext security for the delegator, the delegatee's IBE does not need to be IND-CPA secure.

1.2 Organization

The rest of the paper is organized as follows. In Section 2 we provide some preliminary knowledge on pairing and IBE. In Section 3 we present the security model for inter-domain ID-based proxy re-encryption. In Section 4 we present our new inter-domain ID-based proxy re-encryption scheme and analyze its security. In Section 5 we conclude the paper.

2 Preliminary

We first review the necessary knowledge about pairing and the related assumptions. More detailed information can be found in the seminal paper [4]. A pairing (or, bilinear map) satisfies the following properties:

1. $\mathbb{G}$ and $\mathbb{G}_1$ are two multiplicative groups of prime order p;
2. g is a generator of $\mathbb{G}$;
3. $\hat{e} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_1$ is an efficiently-computable bilinear map with the following properties:
   - Bilinear: for all $u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p^*$, we have $\hat{e}(u^a, v^b) = \hat{e}(u, v)^{ab}$.
   - Non-degenerate: $\hat{e}(g, g) \neq 1$.

As defined in [4], $\mathbb{G}$ is said to be a bilinear group if the group action in $\mathbb{G}$ can be computed efficiently and there exists a group $\mathbb{G}_1$ and an efficiently-computable bilinear map $\hat{e}$ as above.

The Bilinear Diffie-Hellman (BDH) problem in $\mathbb{G}$ is as follows: given a tuple $g, g^a, g^b, g^c \in \mathbb{G}$ as input, output $\hat{e}(g, g)^{abc} \in \mathbb{G}_1$. An algorithm $\mathcal{A}$ has advantage ϵ in solving BDH in $\mathbb{G}$ if

$$\Pr[\mathcal{A}(g, g^a, g^b, g^c) = \hat{e}(g, g)^{abc}] \geq \epsilon.$$

Similarly, we say that an algorithm $\mathcal{A}$ has advantage ϵ in solving the decision BDH problem in $\mathbb{G}$ if

$$|\Pr[\mathcal{A}(g, g^a, g^b, g^c, \hat{e}(g,g)^{abc}) = 0] - \Pr[\mathcal{A}(g, g^a, g^b, g^c, T) = 0]| \geq \epsilon.$$

where the probability is over the random choice of $a, b, c \in \mathbb{Z}_p^*$, the random choice of $T \in \mathbb{G}_1$, and the random bits of $\mathcal{A}$.

Definition 1. *We say that the (decision) (t, ϵ)-BDH assumption holds in $\mathbb{G}$ if no t-time algorithm has advantage at least ϵ in solving the (decision) BDH problem in $\mathbb{G}$.*

Given a security parameter k, a problem (say, BDH) is said to be intractable if any adversary has only negligible advantage in reasonable time. We usually define a scheme to be secure if any adversary has only a negligible advantage in the underlying security model. The time parameter is usually ignored.

Definition 2. *The function $P(k) : \mathbb{Z} \to \mathbb{R}$ is said to be negligible if, for every polynomial $f(k)$, there exists an integer N_f such that $P(k) \leq \frac{1}{f(k)}$ for all $k \geq N_f$.*

In IBE, we assume a Trusted Key Generation Center (KGC) will generate the public system parameter and dynamically issue private keys for users. An IBE scheme consists of four algorithms ($\mathsf{Setup}, \mathsf{Extract}, \mathsf{Encrypt}, \mathsf{Decrypt}$).

- $\mathsf{Setup}(k)$: Run by the KGC, this algorithm takes a security parameter k as input and generates the public parameter is *params* and a master key mk. The public parameter *params* is an implicit input for other algorithms and we omit it in the description for simplicity.
- $\mathsf{Extract}(id, mk)$: Run by the KGC, this algorithm takes an identifier id and the master key mk as input, and outputs the private key sk_{id} corresponding to id.
- $\mathsf{Encrypt}(m, id)$: Run by the message sender, this algorithm takes a message m and an identifier id as input, and outputs a ciphertext c encrypted under the public key corresponding to id. Suppose that the plaintext space is $\mathcal{M}$.
- $\mathsf{Decrypt}(c, sk_{id})$: Run by the user with identifier id, this algorithm takes a ciphertext c and the private key sk_{id} as input, and outputs the message m.

The semantic security against an adaptive chosen plaintext attack (IND-CPA) is modelled by an IND-CPA game between a challenger and an adversary, where the challenger simulates the protocol execution and answers the queries from the adversary. Similarly, we can also define the one-wayness for IBE. Both attack games are depicted in Figure 1, and detailed explanations can be found in [4].

Note that, in both games, the adversary is not allowed to issue a query to the $\mathsf{Extract}$ oracle with the input id^*. We assume the parameter *params* contains the state information generated during the experiment.

Definition 3. *An IBE scheme is said to be semantically secure against an adaptive chosen plaintext attack (IND-CPA) if any polynomial time adversary's advantage is negligible in the IND-CPA game, where the advantage is defined to be $|\Pr[b' = b] - \frac{1}{2}|$.*

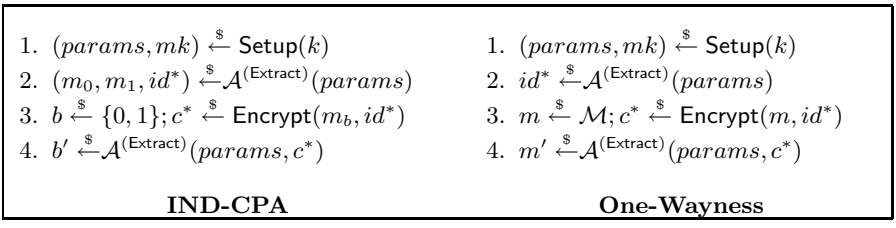

Fig. 1. Security Definitions for IBE

Definition 4. *An IBE scheme is said to be one-way if any polynomial time adversary's advantage is negligible in the One-Wayness game, where the advantage is defined to be* $\Pr[m' = m]$.

3 Inter-domain ID-Based Proxy Re-encryption

Analogous to the traditional proxy re-encryption schemes (e.g. [1,2]), an inter-domain ID-based proxy re-encryption scheme allows a proxy to convert ciphertexts for an IBE user into ciphertexts for another IBE user, where the IBE users are from two different domains. In practice, there might be multiple different parties who play the role of proxy. For example, Alice may choose a party to delegate her decryption right to Bob and Eve may choose a different party to delegate his decryption right to Charlie, while these two proxy parties have no relationship. For the simplicity of description, we only assume one proxy in our security analysis and this proxy is given all the proxy keys.

Suppose that the delegator is registered at KGC_1 with an IBE scheme

$$(\mathsf{Setup}_1, \mathsf{Extract}_1, \mathsf{Encrypt}_1, \mathsf{Decrypt}_1)$$

and the delegatee is registered at KGC_2 with another IBE scheme

$$(\mathsf{Setup}_2, \mathsf{Extract}_2, \mathsf{Encrypt}_2, \mathsf{Decrypt}_2).$$

As a result, there are five types of parties involved in the system: KGC_1, the delegator (and IBE users in the delegator's domain), the proxy, KGC_2, and the delegatee (and IBE users in the delegatee's domain). Apart from the IBE algorithms, an inter-domain IBE proxy re-encryption scheme consists of the following two new algorithms:

- $\mathsf{Pextract}(id, id', sk_{id}, \{sk_{id'}, mk_1, mk_2\})$: This algorithm takes the delegator's identifier id, the delegatee's identifier id', the delegator's private key sk_{id}, and possibly also $\{sk_{id'}, mk_1, mk_2\}$ as input and outputs the proxy key $rk_{id \to id'}$ to the proxy. This algorithm will be run by the delegator and possibly with other parties, such as the delegatee and KGCs.
- $\mathsf{Preenc}(c, rk_{id \to id'})$: Run by the proxy, this algorithm takes a ciphertext c for the delegator and the proxy key $rk_{id \to id'}$ as input, and outputs a new ciphertext c' for the delegatee.

Compared with that of Green and Ateniese [7], we have made the definition of Pextract a more general one. This definition has made the semantic security definition for the delegatee necessary, because the proxy key may leak some information on the delegatee's private key. The definition given by Matsuo [11] might be as general as ours, but the semantic security for the delegatee has been ignored. In the Appendix A, we show that a scheme, which has proven secure under the definition of Matsuo [11], may be insecure in practice.

3.1 Threat Model

We assume that both KGC_1 and KGC_2 are fully trusted. As mentioned in [5], the key escrow problem of IBE can be avoided by applying some standard techniques (such as secret sharing) to the underlying scheme, hence, we skip a formal discussion of this problem in this paper. We identify the following security requirements with respect to plaintext privacy.

1. The involved proxy is assumed to be curious in the following sense: it will honestly convert the delegator's ciphertexts using the proxy key; however, it might be curious to obtain some information about the plaintexts of the delegator and the delegatee. Ideally, the proxy should not obtain any information about the plaintexts of either the delegator or the delegatee.
2. The delegatee should be able to decrypt all the appropriate type of plaintexts of the delegator after the re-encryption by the proxy. However, the delegatee alone should not obtain any information about the plaintexts before the re-encrypted by the proxy. This is essential when we want the proxy to be a policy enforcer.
3. Besides the re-encrypted ciphertexts from the delegator, a delegatee might also receive messages which are encrypted directly using his public key. The delegator and the proxy should not obtain any information about these messages.

In our formal definitions, the first and second requirements lead to the IND-CPA security for the delegator, and the third requirement leads to the IND-CPA security for the delegatee.

3.2 Formal Semantic Security Definitions

Semantic security for the delegator. In standard CPA security formulation for IBE (e.g. that in Section 2), the adversary is restricted from issuing any decryption query while it is allowed to obtain the ciphertext for any plaintext query (by running the encryption function). For the CPA security formulation for inter-domain proxy re-encryption, we want to apply the same restriction so that the adversary (either a curious proxy or a curious delegatee) is restricted from issuing any $\mathsf{Decrypt}_1$, $\mathsf{Decrypt}_2$, and Preenc query. Note that, for a malicious delegatee, a Preenc query is equivalent to a $\mathsf{Decrypt}_1$ query. In our case, the adversary is allowed to issue $\mathsf{Preenc}^{\dagger}$ query to obtain the re-encrypted ciphertext

for any plaintext. This oracle query models the situation that a curious delegatee naturally has access to the plaintexts which have been re-encrypted by the proxy. This issue has been ignored in the CPA security formulation in [7].

As a standard practice, the security is evaluated by an attack game played between a challenger and an adversary, where the challenger simulates the protocol execution and answers the queries from the adversary. Note that the allowed queries for the adversary reflects the adversary's capability in practice.

Definition 5. *An inter-domain ID-based proxy re-encryption scheme is said to be IND-CPA secure for the delegator if any polynomial time adversary has only a negligible advantage in the IND-CPA game, where the advantage is defined to be* $|\Pr[b' = b] - \frac{1}{2}|$.

1. $(params_1, mk_1) \xleftarrow{\$} \mathsf{Setup}_1(k); (params_2, mk_2) \xleftarrow{\$} \mathsf{Setup}_2(k)$
2. $(m_0, m_1, id^*) \xleftarrow{\$} \mathcal{A}^{(\mathsf{Extract}_1, \mathsf{Extract}_2, \mathsf{Pextract}, \mathsf{Preenc}^\dagger)}(params_1, params_2)$
3. $b \xleftarrow{\$} \{0,1\}; c^* \xleftarrow{\$} \mathsf{Encrypt}_1(m_b, id^*)$
4. $b' \xleftarrow{\$} \mathcal{A}^{(\mathsf{Extract}_1, \mathsf{Extract}_2, \mathsf{Pextract}, \mathsf{Preenc}^\dagger)}(params_1, params_2, c^*)$

Fig. 2. Semantic security for the delegator

As depicted in Figure 2, the IND-CPA game is as follows.

1. Game setup: The challenger takes a security parameter k as input, runs the Setup_1 algorithm to generate the public system parameter $params_1$ and the master key mk_1, and runs the Setup_2 algorithm to generate the public system parameter $params_2$ and the master key mk_2.
2. Phase 1: The adversary takes $params_1$ and $params_2$ as input, and is allowed to issue the following types of oracle queries:
   (a) $\mathsf{Extract}_1$ query with any identifier id: The challenger returns the private key sk_{id} corresponding to id.
   (b) $\mathsf{Extract}_2$ query with any identifier id': The challenger returns the private key $sk_{id'}$ corresponding to id'.
   (c) $\mathsf{Pextract}$ query with (id, id'): The challenger returns the delegation key $rk_{id \to id'}$.
   (d) $\mathsf{Preenc}^\dagger$ query with (m, id, id'): The challenger computes $c = \mathsf{Encrypt}_1(m, id)$ and then returns a new ciphertext $c' = \mathsf{Preenc}(c, rk_{id \to id'})$.

   Once the adversary decides that Phase 1 is over, it outputs two equal length plaintexts m_0, m_1 and an identifier id^* on which it wishes to be challenged. There are two constraints here. The first one is that id^* has not been the input to any $\mathsf{Extract}_1$ query. The second one is that, at the end of Phase 1, for any id', if (id^*, id') has been the input to a $\mathsf{Pextract}$ query then id' should not have been the input to any $\mathsf{Extract}_2$ query.
3. Challenge: The challenger picks a random bit $b \in \{0,1\}$ and return $c^* = \mathsf{Encrypt}_1(m_b, id^*)$ as the challenge to the adversary.

4. Phase 2: The adversary is allowed to continue issuing the same types of queries as in Phase 1 but with the following constraints. The adversary is not allowed to issue any $\mathsf{Extract}_1$ query with id^*. At the end of Phase 2, for any id', if (id^*, id') has been the input to a $\mathsf{Pextract}$ query then id' should not be the input to any $\mathsf{Extract}_2$ query.
5. Guess (game ending): The adversary outputs a guess $b' \in \{0, 1\}$.

In this attack game for CPA security, the adversary (either a curious proxy or a curious delegatee) has been given the maximum privilege under the condition that it should not trivially win the game. If the adversary acts as a malicious proxy, the adversary is allowed to obtain any proxy keys and IBE keys from both domains, except for the trivial cases: obtain the private key sk_{id^*} from KGC_1's domain or a delegatee's private key for which the adversary knows the proxy key. If the adversary acts as a malicious delegatee, the adversary is allowed to obtain any proxy keys and IBE keys from both domain and access to the $\mathsf{Preenc}^\dagger$ oracle, except for the trivial cases: obtain the private key sk_{id^*} from KGC_1's domain or the proxy key for which the adversary knows the IBE private key.

Semantic security for the delegatee. Similarly, we can define the chosen plaintext security for the delegatee and the corresponding IND-CPA game is depicted in Figure 3.

Definition 6. *An inter-domain ID-based proxy re-encryption scheme is said to be IND-CPA secure for the delegatee if any polynomial time adversary has only a negligible advantage in the IND-CPA game, where the advantage is defined to be* $|\Pr[b' = b] - \frac{1}{2}|$.

1. $(params_1, mk_1) \xleftarrow{\$} \mathsf{Setup}_1(k); (params_2, mk_2) \xleftarrow{\$} \mathsf{Setup}_2(k)$
2. $(m_0, m_1, id^*) \xleftarrow{\$} \mathcal{A}^{(\mathsf{Extract}_1, \mathsf{Extract}_2, \mathsf{Pextract})}(params_1, params_2)$
3. $b \xleftarrow{\$} \{0, 1\}; c^* \xleftarrow{\$} \mathsf{Encrypt}_2(m_b, id^*)$
4. $b' \xleftarrow{\$} \mathcal{A}^{(\mathsf{Extract}_1, \mathsf{Extract}_2, \mathsf{Pextract})}(params_1, params_2, c^*)$

Fig. 3. Semantic security for the delegatee

Note that, in this attack game, the adversary is not allowed to issue a query to the $\mathsf{Extract}_2$ oracle with the input id^*. In this attack game, the adversary has been given the maximum privilege under the condition that it should not trivially win the game because the adversary is allowed to obtain any proxy keys and IBE keys from both domain, except for the trivial cases: obtain the private key sk_{id^*} from KGC_2's domain.

According to our definition, if an inter-domain ID-based proxy re-encryption scheme achieves IND-CPA security for the delegatee, then it is uni-directional according to the definition in [1], i.e. delegation from the delegator to the delegatee does not allow re-encryption (using the same proxy key) from the delegatee to the delegator.

4 An Inter-domain ID-Based Proxy Re-encryption Scheme

In this section we propose a new inter-domain ID-based proxy re-encryption scheme by extending the concept of the Green-Ateniese proxy re-encryption scheme IBP1 [7]. We then prove its security in our security model.

4.1 Description of Our Scheme

The delegator uses a variant of the Boneh-Franklin IBE scheme [4]. Similar modifications are also made in [7] and they are essential for us to construct proxy re-encryption schemes.

1. $\mathsf{Setup}_1(k)$: Run by the KGC, this algorithm takes a security parameter k as input and generates two cyclic groups $\mathbb{G}$ and $\mathbb{G}_1$ of prime order p, a generator g of $\mathbb{G}$, a bilinear map $\hat{e} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_1$, a master secret key $\alpha \in \mathbb{Z}_p^*$, and a hash function $\mathsf{H}_1 : \{0,1\}^* \rightarrow \mathbb{G}$. The public parameter is $params = (\mathbb{G}, \mathbb{G}_1, p, g, \mathsf{H}_1, \hat{e}, pk)$, where $pk = g^\alpha$ is the public key of the KGC.
   In the original Boneh-Franklin scheme, the plaintext space is $\{0,1\}^n$ where n is an integer and there is an additional hash function $\mathsf{H}_2 : \mathbb{G}_1 \rightarrow \{0,1\}^n$.
2. $\mathsf{Extract}_1(id)$: Run by the KGC, this algorithm takes an identifier $id \in \{0,1\}^*$ and the master key mk as input, and outputs the private key $sk_{id} = pk_{id}^\alpha$, where $pk_{id} = \mathsf{H}_1(id)$.
3. $\mathsf{Encrypt}_1(m, id)$: Run by the message sender, this algorithm takes a message $m \in \mathbb{G}_1$ and an identifier $id \in \{0,1\}^*$ as input, and outputs the ciphertext $c = (c_1, c_2)$ where $r \in \mathbb{Z}_p^*$, $c_1 = g^r$, and $c_2 = m \cdot \hat{e}(pk_{id}, pk)^r$.
   In the original Boneh-Franklin scheme, $c_2 = m \oplus \mathsf{H}_2(\hat{e}(pk_{id}, pk)^r)$.
4. $\mathsf{Decrypt}_1(c, sk_{id})$: Run by the receiver with identifier id, this algorithm takes a ciphertext $c = (c_1, c_2)$ and the private key sk_{id} as input, and outputs the message $m = \frac{c_2}{\hat{e}(sk_{id}, c_1)}$.
   In the original Boneh-Franklin scheme, $m = c_2 \oplus \mathsf{H}_2(\hat{e}(sk_{id}, c_1))$.

Suppose that the delegator is registered at KGC_1 with the above IBE scheme, and possesses identifier and private key pair (id, sk_{id}). KGC_1 publishes another hash function $\mathsf{H}_2 : \{0,1\}^* \rightarrow \mathbb{G}$. Suppose that the delegatee is registered at KGC_2 with another IBE scheme $(\mathsf{Setup}_2, \mathsf{Extract}_2, \mathsf{Encrypt}_2, \mathsf{Decrypt}_2)$, and possesses identifier and private key pair $(id', sk_{id'})$. Suppose also that this IBE scheme has message space $\mathcal{M}_2$.

If the delegator wants to delegate his decryption right to the delegatee, the algorithms are as follows.

- $\mathsf{Pextract}(id, id', sk_{id})$: Run by the delegator, this algorithm outputs the proxy key $rk_{id \rightarrow id'} = (sk_{id}^{-1} \cdot \mathsf{H}_2(X), e_{id \rightarrow id'})$, where $X \in_R \mathcal{M}_2$ and $e_{id \rightarrow id'} = \mathsf{Encrypt}_2(X, id')$.

– $\mathsf{Preenc}(c, rk_{id\to id'})$: Run by the proxy, this algorithm takes a ciphertext $c = (c_1, c_2)$, where $c_1 = g^r$ and $c_2 = m \cdot \hat{e}(pk_{id}, pk)^r$, and $rk_{id\to id'} = (sk_{id}^{-1} \cdot \mathsf{H}_2(X), e_{id\to id'})$ as input, and outputs a new ciphertext $c' = (c'_1, c'_2, c'_3)$ for the delegatee, where

$$c'_1 = \mathsf{Encrypt}_2(c_1, id'), \;\; c'_3 = e_{id\to id'},$$

$$\begin{aligned} c'_2 &= c_2 \cdot \hat{e}(c_1, sk_{id}^{-1} \cdot \mathsf{H}_2(X) \cdot \mathsf{H}_2(c_1)) \\ &= m \cdot \hat{e}(pk_{id}, pk)^r \cdot \hat{e}(c_1, sk_{id}^{-1} \cdot \mathsf{H}_2(X) \cdot \mathsf{H}_2(c_1)) \\ &= m \cdot \hat{e}(c_1, \mathsf{H}_2(X) \cdot \mathsf{H}_2(c_1)). \end{aligned}$$

Given a re-encrypted ciphertext c', the delegatee can obtain the plaintext m by computing:

$$\begin{aligned} m' &= \frac{c'_2}{\hat{e}(\mathsf{Decrypt}_2(c'_1, sk_{id'}), \mathsf{H}_2(\mathsf{Decrypt}_2(c'_3, sk_{id'})) \cdot \mathsf{H}_2(\mathsf{Decrypt}_2(c'_1, sk_{id'})))} \\ &= \frac{m \cdot \hat{e}(c_1, \mathsf{H}_2(X) \cdot \mathsf{H}_2(c_1))}{\hat{e}(c_1, \mathsf{H}_2(X) \cdot \mathsf{H}_2(c_1))} \\ &= m. \end{aligned}$$

Note that, to decrypt a re-encrypted ciphertext c', the delegatee needs to obtain in advance KGC_1's public parameter $(\mathbb{G}, \mathbb{G}_1, g, p, \hat{e}, \mathsf{H}_1, \mathsf{H}_2)$.

The proposed inter-domain ID-based proxy re-encryption scheme differs from the Green-Ateniese proxy re-encryption scheme IBP1 [7] in the following aspects:

1. In our scheme, the delegator and the delegatee are from two different domains, while they are required to be the same domain in the IBP1.
2. In the Green-Ateniese proxy re-encryption scheme IBP1, the algorithm Preenc outputs $c'_1 = c_1$ and $c'_2 = c_2 \cdot \hat{e}(c_1, sk_{id}^{-1} \cdot \mathsf{H}_2(X))$. Our modifications in the above scheme are essential for us to prove the IND-CPA security. Note that under our security definition the adversary is allowed access to the $\mathsf{Preenc}^\dagger$ oracle, which is different from that in [7]. Without the modifications, we cannot prove our result.

4.2 Analysis of the General Construction

Since the delegator generates the proxy key on his own, therefore, from Definition 3 and Definition 6, the following result is straightforward.

Lemma 1. *Given that* $(\mathsf{Setup}_2, \mathsf{Extract}_2, \mathsf{Encrypt}_2, \mathsf{Decrypt}_2)$ *is IND-CPA secure, the proposed inter-domain ID-based proxy re-encryption scheme is IND-CPA secure for the delegatee.*

Lemma 2. *Given that* $(\mathsf{Setup}_2, \mathsf{Extract}_2, \mathsf{Encrypt}_2, \mathsf{Decrypt}_2)$ *is one-way, the proposed inter-domain ID-based proxy re-encryption scheme is IND-CPA secure for the delegator based on the decision BDH assumption in the random oracle model.*

Proof sketch. We suppose that the total number of queries issued to H_1 and H_2 is bounded by integer q_1 and q_2, respectively[1]. Suppose an adversary $\mathcal{A}$ acting as a malicious delegatee has the non-negligible advantage ϵ in the IND-CPA game. The security proof is done through a sequence of games [13].

Game_0: In this game, the challenger faithfully simulates the protocol execution and answers the oracle queries from $\mathcal{A}$. the challenger simulates the random oracle H_1 as follows: the challenger maintains a list of vectors, each of them containing a request message, an element of $\mathbb{G}$ (the hash-code for this message), and an element of $\mathbb{Z}_p^*$. After receiving a request message, the challenger first checks its list to see whether the request message is already in the list. If the check succeeds, the challenger returns the stored element of $\mathbb{G}$; otherwise, the challenger returns g^y, where y a randomly chosen element of $\mathbb{Z}_p^*$, and stores the new vector in the list. the challenger simulates the random oracle H_2 as follows: the challenger maintains a list of vectors, each of them containing a request message and an element of $\mathbb{G}$ (the hash-code for this message). After receiving a request message, the challenger first checks its list to see whether the request message is already in the list. If the check succeeds, the challenger returns the stored element of $\mathbb{G}$; otherwise, the challenger returns u which is a randomly chosen element of $\mathbb{G}$, and stores the new vector in the list.

Let $\delta_0 = \Pr[b' = b]$, as we assumed at the beginning, $|\delta_0 - \frac{1}{2}| = \epsilon$.

Game_1: In this game, the challenger performs as follows.

1. Game setup: the challenger faithfully simulates the setup phase.
2. Phase 1: the challenger randomly selects $j \in \{1, 2, \cdots, q_1 + 1\}$. If $j = q_1 + 1$, the challenger faithfully answers the oracle queries from $\mathcal{A}$. If $1 \leq j \leq q_1$, we assume the j-th input to H_1 is $\tilde{id}$ and the challenger answers the oracle queries from $\mathcal{A}$ as follows: Answer $\mathsf{Extract}_1$, $\mathsf{Extract}_2$, $\mathsf{Pextract}$, and $\mathsf{Preenc}^\dagger$ faithfully, except that the challenger aborts as a failure when $\tilde{id}$ is the input to a $\mathsf{Extract}_1$ query.
3. Challenge: After receiving (m_0, m_1, id^*) from the adversary, if one of the following events occurs, the challenger aborts as a failure.
   (a) id^* has been issued to H_1 as the i-th query and $i \neq j$,
   (b) id^* has not been issued to H_1 and $1 \leq j \leq q_1$.
   Note that, if the adversary does not abort then either $1 \leq j \leq q_1$ and $id^* = \tilde{id}$ is the input to j-th H_1 query or $j = q_1 + 1$ and id^* has not been the input to any H_1 query. the challenger faithfully returns the challenge.
4. Phase 2: the challenger answers the oracle queries faithfully.
5. Guess (game ending): the adversary outputs a guess $b' \in \{0, 1\}$.

The probability that the challenger successfully ends is $\frac{1}{q_1+1}$, i.e. the probability that the challenger does not abort in its execution is $\frac{1}{q_1+1}$. Let $\delta_1 = \Pr[b' = b]$ when the challenger successfully ends, in which case $|\delta_1 = \delta_0|$. Let θ_1 be the probability that the challenger successfully ends and $b' = b$. We have $\theta_1 = \frac{\delta_1}{q_1+1}$.

Game_2: In this game, the challenger simulates the protocol execution and answers the oracle queries from $\mathcal{A}$ in the following way.

[1] For simplicity of description, it is reasonable to assume that the total number is counted for queries with different inputs.

1. Game setup: the challenger faithfully simulates the setup phase.
2. Phase 1: the challenger randomly selects $j \in \{1, 2, \cdots, q_1 + 1\}$. If $j = q_1 + 1$, the challenger faithfully answers the oracle queries from $\mathcal{A}$. If $1 \leq j \leq q_1$, the challenger answers j-th query to H_1 with g^β where $\beta \in_R \mathbb{Z}_p^*$, and answers the oracle queries from $\mathcal{A}$ as follows. Suppose the input of the j-th query to H_1 is $\tilde{id}$. the challenger answers queries to $\mathsf{Extract}_1$, $\mathsf{Extract}_2$, $\mathsf{Pextract}$, and $\mathsf{Preenc}^\dagger$ in the same way as in Game_1, except for the following. the challenger keeps a list of vector $(id', rk_{\tilde{id}\to id'}, X_{id'}, X'_{id'}, g_{id'}, h_{id'})$.
   (a) $\mathsf{Pextract}$ query with the input $(\tilde{id}, id')$: If $rk_{\tilde{id}\to id'}$ exists in one of the lists, the challenger returns the value. Otherwise, the challenger returns the proxy key $rk_{\tilde{id}\to id'}$, where

$$X_{id'}, X'_{id'} \in_R \mathcal{M}_2,\ g_{id'} \in_R \mathbb{G},\ e_{\tilde{id}\to id'} = \mathsf{Encrypt}_2(X'_{id'}, id'),$$

$$\mathsf{H}_2(X_{id'}) = g_{id'},\ rk_{\tilde{id}\to id'} = (sk_{\tilde{id}}^{-1} \cdot g_{id'}, e_{\tilde{id}\to id'}).$$

   the challenger adds the vector $(id', rk_{\tilde{id}\to id'}, X_{id'}, X'_{id'}, g_{id'}, h_{id'})$ to the list, where $h_{id'}$ is set to be a special symbol $\perp$.
   (b) $\mathsf{Preenc}^\dagger$ query with the input $(m, \tilde{id}, id')$: the challenger performs according to the following rules.
   – If $rk_{\tilde{id}\to id'}$ does not exist, the challenger generates

$$(id', rk_{\tilde{id}\to id'}, X_{id'}, X'_{id'}, g_{id'}, h_{id'}),$$

   where

$$X_{id'}, X'_{id'} \in_R \mathcal{M}_2,\ g_{id'} \in_R \mathbb{G},\ e_{\tilde{id}\to id'} = \mathsf{Encrypt}_2(X'_{id'}, id'),$$

$$\mathsf{H}_2(X_{id'}) = g_{id'},\ rk_{\tilde{id}\to id'} = (sk_{\tilde{id}}^{-1} \cdot g_{id'}, e_{\tilde{id}\to id'}).$$

   the challenger then returns a new ciphertext $c' = (c'_1, c'_2, c'_3)$, where

$$t_1, c_1, g_{c_1} \in_R \mathbb{G},\ c'_1 = \mathsf{Encrypt}_2(c_1, id'),$$

$$c'_2 = m \cdot \hat{e}(t_1, g) \cdot \hat{e}(c_1, g_{c_1}),\ c'_3 = e_{\tilde{id}\to id'},\ h_{id'} = g_{id'} \cdot c_1^{-1} \cdot t_1,$$

   and adds the new vector to the list.
   – If $rk_{\tilde{id}\to id'}$ exists in the list but $h_{id'} = \perp$ (which means a $\mathsf{Pextract}$ query with the input $(\tilde{id}, id')$ has been issued), the challenger then returns a new ciphertext $c' = (c'_1, c'_2, c'_3)$ faithfully.
   – If $rk_{\tilde{id}\to id'}$ exists in the list and $h_{id'} \neq \perp$, the challenger then returns a new ciphertext $c' = (c'_1, c'_2, c'_3)$, where

$$c_1, g_{c_1} \in_R \mathbb{G},\ c'_1 = \mathsf{Encrypt}_2(c_1, id'),$$

$$c'_2 = m \cdot \hat{e}(c_1 \cdot h_{id'}) \cdot \hat{e}(c_1, g_{c_1}),\ c'_3 = e_{\tilde{id}\to id'}.$$

   After a $\mathsf{Preenc}^\dagger$ query with the input $(m, \tilde{id}, id')$ and a $\mathsf{Extract}_2$ query with the input id' have been issued, the challenger returns $h_{id'}$ if $X'_{id'}$ is issued to H_2 and returns g_{c_1} if c_1 is issued to H_2.

3. Challenge: the challenger performs in the same way as in Game_1, except that answers $\mathsf{Pextract}$ query with the input (id^*, id') and $\mathsf{Preenc}^\dagger$ query with the input (m, id^*, id') as in Phase 1.
4. Phase 2: the challenger answers the oracle queries from $\mathcal{A}$ as in Phase 1.
5. Guess (game ending): the adversary outputs a guess $b' \in \{0, 1\}$.

Let θ_2 be the probability that the challenger successfully ends and $b' = b$. Let E_1 be the event that, for some id' and m, the adversary issues a H_2 query with the input $X'_{id'}$ query but there is no $\mathsf{Extract}_2$ query with the input id', or the adversary issues a H_2 query with the input $X'_{id'}$ query before any $\mathsf{Preenc}^\dagger$ query with the input $(m, \tilde{id}, id')$, or the adversary issues a H_2 query with the input c_1 query but there is no $\mathsf{Extract}_2$ query with the input id'. Compared with Game_1, Game_2 differs when E_1 occurs. From the difference lemma [13], we have $|\theta_2 - \theta_1| \leq \epsilon_2 = \Pr[E_1]$ which is negligible based on the one-wayness of $(\mathsf{Setup}_2, \mathsf{Extract}_2, \mathsf{Encrypt}_2, \mathsf{Decrypt}_2)$ in the random oracle model.

Game_3: In this game, the challenger simulates the protocol execution and answers the oracle queries from $\mathcal{A}$ in the same way as in Game_2, except for the following. For a $\mathsf{Pextract}$ query with the input (id^*, id'), the challenger returns the proxy key $rk_{id^* \to id'}$, where $T_{id'}, X'_{id'} \in_R \mathbb{G}_1$ and

$$rk_{id^* \to id'} = (T_{id'}, \mathsf{Encrypt}_2(X'_{id'}, id')).$$

Let θ_3 be the probability that the challenger successfully ends and $b' = b$. Compared with Game_2, we only change the notation of random value $sk_{\tilde{id}}^{-1} \cdot g_{id'}$ with $T_{id'}$. As a result, we have $|\theta_3 - \theta_2| = \epsilon_3 = 0$.

Game_4: In this game, the challenger simulates the protocol execution and answers the oracle queries from $\mathcal{A}$ in the same way as in Game_3, except for the following. In the challenge phase the challenger returns $c^* = (c_1^*, c_2^*)$ as the challenge, where

$$b \in_R \{0, 1\},\ r \in_R \mathbb{Z}_p^*,\ T \in_R \mathbb{G}_1,\ c_1^* = g^r,\ c_2^* = m_b \cdot T.$$

Let θ_4 be the probability that the challenger successfully ends and $b' = b$. We have $\theta_4 = \frac{1}{2(q_1+1)}$ since $T \in_R \mathbb{G}_1$. Compared with Game_3, the only difference in Game_4 is that $\hat{e}(g, g)^{\alpha \cdot \beta \cdot r}$ is replaced with $T \in_R \mathbb{G}_1$ in the challenge phase. Using the interpolation method [13], we have $|\theta_4 - \theta_3| \leq \epsilon_4$ which is negligible based on the decision BDH assumption.

From $|\theta_2 - \theta_1| \leq \epsilon_2$, $|\theta_3 - \theta_2| \leq \epsilon_3$, $|\theta_4 - \theta_3| \leq \epsilon_4$, and $\theta_4 = \frac{1}{2(q_1+1)}$, we have $|\frac{1}{2(q_1+1)} - \theta_1| \leq \epsilon_2 + \epsilon_3 + \epsilon_4$. In addition, from $|\delta_0 - \frac{1}{2}| = \epsilon$, $|\delta_1 - \delta_0| \leq \epsilon_1$ and $\theta_1 = \frac{\delta_1}{q_1+1}$, we have $\frac{\epsilon}{q_1+1} \leq \frac{\epsilon_1}{q_1+1} + \epsilon_2 + \epsilon_3 + \epsilon_4$. Because ϵ_i $(1 \leq i \leq 4)$ are negligible and ϵ is assumed to be non-negligible, we get a contradiction. As a result, the proposed inter-domain ID-based proxy re-encryption scheme is IND-CPA secure based on the decision BDH assumption in the random oracle model, given that $(\mathsf{Setup}_2, \mathsf{Extract}_2, \mathsf{Encrypt}_2, \mathsf{Decrypt}_2)$ is one-way. □

5 Conclusion

In this paper, we have examined the concept of inter-domain ID-based proxy re-encryption and proposed a chosen plaintext security security definitions. We have also proposed an inter-domain ID-based proxy re-encryption scheme which has the interesting property that, to achieve the chosen plaintext security for the delegator, the delegatee's IBE only needs to be one-way. In our security formulation, only chosen plaintext security has been defined, however this definition can be extended to model chosen ciphertext security by appropriately allowing $\mathsf{Decrypt}_1$, $\mathsf{Decrypt}_2$, and Preenc queries to the adversary. It is an interesting future work to construct inter-domain ID-based proxy re-encryption schemes with chosen ciphertext security. It is also interesting to further investigate the application of inter-domain proxy re-encryption in emerging fields such as Personal Health Record (PHR) protection [14].

References

1. Ateniese, G., Fu, K., Green, M., Hohenberger, S.: Improved proxy re-encryption schemes with applications to secure distributed storage. ACM Trans. Inf. Syst. Secur. 9(1), 1–30 (2006)
2. Blaze, M., Bleumer, G., Strauss, M.J.: Divertible protocols and atomic proxy cryptography. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 127–144. Springer, Heidelberg (1998)
3. Boneh, D., Boyen, X.: Efficient selective-ID secure identity-based encryption without random oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 223–238. Springer, Heidelberg (2004)
4. Boneh, D., Franklin, M.: Identity-based encryption from the weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
5. Chen, L.: An interpretation of identity-based cryptography. In: Aldini, A., Gorrieri, R. (eds.) FOSAD 2007. LNCS, vol. 4677, pp. 183–208. Springer, Heidelberg (2007)
6. El Gamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 10–18. Springer, Heidelberg (1985)
7. Green, M., Ateniese, G.: Identity-based proxy re-encryption. In: Katz, J., Yung, M. (eds.) ACNS 2007. LNCS, vol. 4521, pp. 288–306. Springer, Heidelberg (2007)
8. Ivan, A., Dodis, Y.: Proxy cryptography revisited. In: Proceedings of the Network and Distributed System Security Symposium, NDSS 2003. The Internet Society (2003)
9. Jakobsson, M.: On quorum controlled asymmetric proxy re-encryption. In: Imai, H., Zheng, Y. (eds.) PKC 1999. LNCS, vol. 1560, pp. 112–121. Springer, Heidelberg (1999)
10. Mambo, M., Okamoto, E.: Proxy cryptosystems: Delegation of the power to decrypt ciphertexts. IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences E80-A(1), 54–63 (1997)
11. Matsuo, T.: Proxy re-encryption systems for identity-based encryption. In: Takagi, T., Okamoto, T., Okamoto, E., Okamoto, T. (eds.) Pairing 2007. LNCS, vol. 4575, pp. 247–267. Springer, Heidelberg (2007)

12. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)
13. Shoup, V.: Sequences of games: a tool for taming complexity in security proofs (2006), http://shoup.net/papers/
14. The US Department of Health and Human Services. Summary of the HIPAA Privacy Rule (2003)
15. Wang, L., Cao, Z., Okamoto, T., Miao, Y., Okamoto, E.: Authorization-Limited Transformation-Free Proxy Cryptosystems and Their Security Analyses. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences (1), 106–114 (2006)
16. Zhou, L., Marsh, M.A., Schneider, F.B., Redz, A.: Distributed blinding for distributed elgamal re-encryption. In: ICDCS 2005: Proceedings of the 25th IEEE International Conference on Distributed Computing Systems, p. 824. IEEE Computer Society Press, Los Alamitos (2005)

Appendix A: An Observation on the Matsuo Scheme

The Matsuo proxy re-encryption scheme assumes the delegator and deletee use the Boneh-Boyen Hierarchical IBE scheme [3] where the identity dimension is set to be 1. The algorithms are as follows:

- Setup, which takes the security parameter ℓ as input, and outputs the public parameter $(g,g_1,g_2,h,p,\mathbb{G},\mathbb{G}_1,\hat{e})$ and the master key mk. Here, $(g,p,\mathbb{G},\mathbb{G}_1,\hat{e})$ is the basic bilinear map parameter, g_2 and h are randomly chosen from $\mathbb{G}$, $mk=\alpha$ and $g_1=g^\alpha$ where α is randomly chosen from $\mathbb{Z}_p^*$. Note that the public parameter is an implicit input to all other algorithms, and we omit it in the description for simplicity.
- Extract, which takes mk and an identifier $ID_t \in \mathbb{Z}_p^*$ as input, and outputs the corresponding private key sk_t, where β_t is randomly chosen from $\mathbb{Z}_p^*$ and

$$\begin{aligned} sk_t &= (g_2^{mk} \cdot (g_1^{ID_t} h)^{\beta_t}, g^{\beta_t}) \\ &= (g_2^{\alpha} (g_1^{ID_t} h)^{\beta_t}, g^{\beta_t}) \\ &= (d_{t0}, d_{t1}). \end{aligned}$$

  Note that t is an integer used to index users in the system.
- Encrypt, which takes a message $m \in \mathbb{G}_1$ and an identifier $ID_t \in \mathbb{Z}_p^*$ as input, and outputs a ciphertext c_t, where r is randomly chosen from $\mathbb{Z}_p^*$ and

$$\begin{aligned} c_t &= (g^r, (g_1^{ID_t} h)^r, m\hat{e}(g_1,g_2)^r) \\ &= (c_{t1}, c_{t2}, c_{t3}). \end{aligned}$$

- Decrypt, which takes a ciphertext c_t and the private key sk_t as input, and outputs a message m by computing

$$m = \frac{c_{t3}\hat{e}(d_{t1}, c_{t2})}{\hat{e}(d_{t0}, c_{t1})}.$$

Suppose Alice and Bob register under the same KGC, and possess identifier/private key pair (ID_i, sk_i) and (ID_j, sk_j), respectively. If Alice wants to delegate her decryption right to Bob, the algorithms of the Matsuo proxy re-encryption scheme [11] are as follows.

- Pextract, which takes (mk, d_{j1}) as input, and outputs the delegation key $rk_{ij} = d_{j1}^{\alpha}$.
- Preenc, which takes rk_{ij} and c_i as input, and outputs a new ciphertext c_j, where

$$\begin{aligned} c_j &= (c_{i1}, c_{i2}, c_{i3}\hat{e}(c_{i1}^{ID_j - ID_i}, rk_{ij})) \\ &= (c_{j1}, c_{j2}, c_{j3}). \end{aligned}$$

Given a re-encrypted ciphertext c_j, Bob can obtain the plaintext m by running the IBE decryption algorithm.

$$\begin{aligned} m' &= \frac{c_{j3}\hat{e}(d_{j1}, c_{j2})}{\hat{e}(d_{j0}, c_{j1})} \\ &= \frac{m\hat{e}(g_1, g_2)^r \hat{e}(g^{r(ID_j - ID_i)}, g^{\alpha\beta_j})\hat{e}(g^{\beta_j}, (g_1^{ID_i}h)^r)}{\hat{e}(g_2^{\alpha}(g_1^{ID_j}h)^{\beta_j}, g^r)} \\ &= \frac{m\hat{e}(g^{\alpha r}, g_2)\hat{e}(g^{\alpha r(ID_j - ID_i)}, g^{\beta_j})\hat{e}(g^{\beta_j}, g^{\alpha r ID_i}h^r)}{\hat{e}(g_2^{\alpha}(g^{\alpha ID_j}h)^{\beta_j}, g^r)} \\ &= \frac{m\hat{e}(g^{\alpha r}, g_2)\hat{e}(g^{\alpha r ID_j}, g^{\beta_j})\hat{e}(g^{\beta_j}, h^r)}{\hat{e}(g_2^r, g^{\alpha})\hat{e}((g^{\alpha ID_j}h)^{\beta_j}, g^r)} \\ &= m. \end{aligned}$$

In the definition of Pextract, Alice is not required to be involved in generation of rk_{ij}, while the master key and Bob's private key is required in the generation of rk_{ij}. It is easy to verify that, the delegation key rk_{ij} is capable of transforming any ciphertext, intended for any user registered at the KGC, into a decrytable ciphertext for Bob. As a result, once Alice has delegated her decryption right to Bob, then implicitly all users in the system have delegated their decryption right to Bob at the same time.