Collusion-Resistant Identity-Based Proxy Re-Encryption Without Random Oracles

JunJie Qiu\(^1,2\), JungBok Jo\(^1\) and HoonJae Lee\(^1\)

\(^1\)Department of Ubiquitous IT, Graduate School of Dongseo University, Busan, Korea
\(^2\)Weifang University of Science & Technology, Weifang, China
qjunj2000@163.com, {jobok, hjlee}@gdsu.dongseo.ac.kr
*Corresponding author: HoonJae Lee

Abstract

In an identity-based proxy re-encryption scheme, a semi-trusted proxy can convert a ciphertext under Alice’s public identity into a ciphertext for Bob. The proxy does not know the secret key of Alice or Bob, and also does not know the plaintext during the conversion. In identity-based proxy re-encryption, the collusion of the proxy and a delegatee may result in the decryption of ciphertext for delegator. In this paper, we present a new scheme which can avoid the collusion of proxy and delegatee. Our scheme improves the scheme of Chu and Tzeng while inheriting all useful properties such as unidirectionality and non-interactivity. In our scheme, we get the security by using added secret parameter and change the secret key and re-encryption key. Our sheme is secure against Chosen-Ciphertext Attack (CCA) and collusion attack in the standard model.

Keywords: Identity-based proxy re-encryption, Collusion attack, CCA, Standard model

1. Introduction

The concept of proxy re-encryption (PRE) is introduced by Blaze, et al. [1] in 1998. PRE schemes allow Alice (the delegator) to create a re-encryption key (or proxy key) and give this key to a semi-trusted proxy. The proxy uses this re-encryption key to transform the ciphertext under Alice’s public key into the ciphertext for Bob (the delegatee). In PRE scheme, the proxy does not know the secret key of Alice or Bob, and also does not know the plaintext during the conversion. PRE schemes have many applications such as digital rights management (DRM), distributed storage, encrypted email forwarding, and law enforcement [1-5].

Green and Ateniese proposed the first identity-based PRE (IB-PRE) [6] in 2007. Their scheme allows a proxy to translate the ciphertext under Alice’s identity into the ciphertext under Bob’s identity. It is based on Boneh-Franklin’s identity-based encryption (IBE) scheme [7], which was shown to be secure in the random oracle model. But many researchers have expressed doubts about the random oracle model. For example, some cryptosystems, which are secure in the random oracle model but are insecure in any real world implementation [8]. Fortunately, Chu and Tzeng proposed an IB-PRE against chosen cipher-text attack (CCA) without random oracles [9]. In [9], Chu et al. proposed two identity-based proxy re-encryption schemes without random oracles. The schemes both satisfy the properties of unidirectionality, non-interactivity and multi-use. But it is vulnerable to collusion attack.

In IB-PRE process, we must carefully consider the proxy’s security properties. Obviously the collusion of the proxy and a delegatee may result in the decryption of ciphertext for a delegator if the ciphertext are re-encrypted. How to effectively avoid collusion attack in the IB-PRE scheme has a very important effect.
Our contributions.

We present a new IB-PRE scheme. This scheme follows CT07 paradigm [9]. The main novelty is that our scheme can avoid the collusion of proxy and delegatee. The scheme improves the scheme of [9] while inheriting all useful properties such as unidirectionality and non-interactivity. In [9], it is easy to get the delegator’s secret key if the proxy and delegatee conspire. In our scheme, we get the security by using added secret parameter and change the secret key and re-encryption key. Our scheme is secure against Chosen-Ciphertext Attack (CCA) and collusion attack in the standard model.

Organizations.

In the rest of this paper, we first give some preliminaries we will use later, include bilinear groups and the definition of IB-PRE. Then we present a collusion-resistant IB-PRE scheme and give its security proof. Finally, we give conclusion of this paper.

2. Preliminaries

In this section, we will give some preliminaries which will use later.

2.1 Bilinear groups

Let $G$ and $G_T$ be two cyclic groups of prime order $p$. And $e$ is a function $e: G \times G \rightarrow G_T$ satisfying the following properties:

- Bilinear: For all $u, v \in G$ and $a, b \in \mathbb{Z}$, $e(u^a, v^b) = e(u, v)^{ab}$.
- Non-degenerate: For some $g \in G$, $e(g, g) \neq 1$.
- Computable: $e$ is efficiently computable.

We say that $G$ is a bilinear group and $e$ is a bilinear pairing in $G$.

2.2 The Identity-based Proxy Re-encryption

An IB-PRE scheme consists of 6 algorithms as follows:

- **Setup($\lambda$)**: Takes security parameter $\lambda$ as input, the algorithm outputs the master secret key $msk$ and public parameters $params$ which is distributed to users.

- **KeyGen(params, msk, id)**: Takes $params$, $msk$, and identity $id \in \{0,1\}^*$ as input, this algorithm generates a decryption key $sk_{id}$ corresponding to the identity $id$.

- **Encrypt(params, id, m)**: Takes $params$, $id$, and a plaintext $m$ as input, outputs the first-level (original) ciphertext $c_{id}$, the encryption of $m$ under identity $id$.

- **RKGen(params, sk_{id1}, id_{id1})**: Takes $params$, $sk_{id1}$ (generated by KeyGen algorithm), identity $id_{id1} \in \{0,1\}^*$ and $id_{id1} \in \{0,1\}^*$ as input, this algorithm generates a re-encryption key $rk_{id_{id1} \rightarrow id_{id2}}$.

- **Reencrypt(params, c_{id1}, rk_{id_{id1} \rightarrow id_{id2}})**: Takes $params$, the first-level ciphertext $c_{id1}$ (generated by Encrypt algorithm), the re-encryption key $rk_{id_{id1} \rightarrow id_{id2}}$ (generated by RKGen algorithm) as input, outputs the second-level (re-encrypted) ciphertext $c_{id2}$.

- **Decrypt(params, sk_{id}, c_{id})**: 


Takes \(params\), secret key \(sk_{id}\), and ciphertext \(c_{id}\) as input, this algorithm outputs the plaintext \(m\) or error symbol \(\bot\).

**Correctness.**

Suppose \((params, msk)\) properly generated by \(Setup(l^{'})\), for all identity \(id\), \(sk_{id} \leftarrow KeyGen(params, msk, id)\), \(r_{id}\) is a random element of \(\mathbb{Z}_p\). Let \(c_{id}\) be the ciphertext output from \(Encrypt(params, id, m)\) or \(Reencrypt(params, c_{id}, r_{id}, m')\). Then the following equations hold:

\[
m = Decrypt(params, sk_{id}, c_{id})
\]
\[
m = Decrypt(params, sk_{id}, c_{id})
\]

**2.3 Decisional Bilinear Diffie-hellman Assumption**

Our scheme is based on Decisional Bilinear Diffie- Hellman assumption ($DBDH$) in \(G, G_\ell\). The definition of $DBDH$ problem as follows:

Let \((G, G_\ell)\) be a pair of bilinear groups, and \(g\) is a random generator of \(G\). Given \(g, g^x, g^y, g^z \in G\), \(T \in G_\ell\), where \(a, b, c \in \mathbb{Z}_p\). Determine if \(T = e(g, g)^{ab}\) or if \(T\) is a random element of \(G_\ell\).

**3. New Identity-based Proxy Re-encryption**

**3.1 Analysis of Collusion Attack in [9]**

We first simply review the scheme of [9]. In [9], Setup algorithm generates the public parameter \(\mu = (g, g_1, g_2, F_1(), F_2(), (G, Sign, Vrfy))\) and master secret key \(msk = g_2^\alpha\), where \(g\) is generator of \(G\), \(g_1 = g^\alpha\), \(g_2\) random. KeyGen algorithm generates the private decryption key \(d_i = (g_2^\alpha F_1(v_1), g_2^\alpha)\), where \(v_1\) is identity and \(r \in \mathbb{Z}_p\). RKGen algorithm generates the re-encryption key \(d_i = (g_2^\alpha F_1(v_1), g_2^\alpha, K = E_k(k) \in G\) (\(k\) randomly), \(R \leftarrow Encrypt(\mu, v_1, k)\).

Suppose proxy and delegatee conspire. For the proxy, the value of re-encryption key \(d_{i\rightarrow j}\) is known. \(R\) is the ciphertext encrypted \(k\) under delegatee’s public key. So the value of \(k\) can be easily obtained, just decrypt \(R\) using delegatee’s secret key. Then \(K\) can be computed. So if proxy and delegatee conspire, \(d_i\) can be obtained. Finally the delegatee and proxy can get the secret key of delegator. If they get delegator’s secret key, they can decrypt any other ciphertext under this secret key.

**3.2 New IB-PRE Algorithms**

The new IB-PRE also consists of 6 algorithms as follows:

- **Setup**: Let \(l^{'}, l\) be security parameter. Let \(G \times G \rightarrow G_\ell\) be a bilinear map, where \(G\) and \(G_\ell\) have prime order \(p\) and \(g\) is a generator of \(G\). Let \(l \leq |p| = 2\) and \(E_i: \{0,1\}^l \rightarrow G_\ell\), \(E_i: \{0,1\}^l \rightarrow G\) be two encodings. Randomly choose secret \(\alpha, \beta \in \mathbb{Z}_p\). Set \(g_i = g^\alpha \times g^\beta\) and randomly choose \(g_2 \in G\). Let \(v, w\) are two \(n\) -bit strings and \(V, W\) are the set of all \(i\) for which the \(i\)-th bit of \(v\) and \(w\) is one. Define two functions \(F_1(v) = u'_1 \prod_{i=1}^n u_{i_1}\) and \(F_2(w) = u'_2 \prod_{i=1}^n u_{i_2}\), where \(u'_i, u_{i_1}\),
, ..., \( u_{n,1} \) and \( u_{n,2} \) are randomly chosen from \( G \). Let \((G, \text{Sign}, \text{Vrfy})\) as a one-time signature scheme. Then output master secret key \( \text{msk} = (g_r^e, g_\mu^e) \) and public parameter \( \mu = (g, g_1, g_2), F_1(\cdot), F_2(\cdot), (G, \text{Sign}, \text{Vrfy}) \).

- **KeyGen**: To extract a decryption key for identity \( id \in (0, 1)^n \), output \( sk_{id} = (g_r^e F_1(id)^t, g_\mu^e F_1(id)^t, g^t) \), where \( r \in Z_p \).

- **Encrypt**: Perform \( G(t^v) \) to get verification key \((vk, sk)\). To encrypt \( m \in (0, 1) \) under identity \( id \),

  1. Compute \( \bar{C} = M \cdot e(g, g_2), g^t, F_1(id)^t, F_2(vk)^t \), where \( t \in Z_p \) and \( M = E_t(m \| 0) \).

  2. Then compute \( \sigma = \text{Sign}_a(\bar{C}) \).

  3. Output \( C_a = (\bar{C}, vk, \sigma) \) as the first-level (original) ciphertext.

- **RKGen**: Let \( sk_{a1} = (sk_1, sk_2, sk_3, sk_4) \). Compute a re-encryption key from \( id_1 \rightarrow id_2 \) as:

\[
\text{RKGen} = (sk_{a1} \rightarrow sk_{a2}) = ((sk_1 \times sk_2) K^{-1}, sk_3, R),
\]

where \( K = E_t(k) \in G \) and \( R \leftarrow \text{Encrypt}‘(\mu, id_2, k), k \in \{0, 1\} \). The difference of Encrypt’ and Encrypt is that \( k \) appends ‘1’ and \( m \) appends ‘0’.

- **Reencrypt**: Let \( \text{RKGen} = (sk_{a1} \rightarrow sk_{a2}) \) and \( C_{a1} = (c_1, c_2, c_3, c_4, vk, \sigma) \). Check if \( \text{Vrfy}_{a1} = (c_1, c_2, c_3, c_4, \sigma) \) is 1. If not, output \( \perp \). Otherwise, compute \( C_{a2} = (C_{a1}, R, sk_4 F_2(\bar{C}), sk_5, g^t) \) as the second-level ciphertext, where \( r' \in Z_p \).

- **Decrypt**:

  1. We can use \( sk_{id} \) and \( C_a \) to decrypt the first-level ciphertext. Let \( sk_{id} = (sk_1, sk_2, sk_3) \) and \( C_a = (c_1, c_2, c_3, c_4, vk, \sigma) \). Check if \( \text{Vrfy}_{a1} = (c_1, c_2, c_3, c_4, \sigma) \) is 1. If not, output \( \perp \). Otherwise, compute \( sk_2 = (sk_1 \times sk_2) F_1(\bar{C}) \), \( sk_1 = g^t \) then compute

\[
M = c_1 \cdot \frac{e^t(sk_1, c_3 \times c_3)}{e^t(sk_1^*, c_2)}
\]

  2. To decrypt the second-level ciphertext, let \( C_{a2} = (C_{a1}, R, sk_4, sk_5, sk_4) \).

Check if \( \text{Vrfy}_{a2} = (c_1, c_2, c_3, c_4, \sigma) \) is 1 and check if \( e^t(sk_4, k, g) \) is equal to \( e^t(g, g_1) \cdot F_1(id_1), sk_1 \). If not, output \( \perp \). Otherwise, compute

\[
M = c_1 \cdot \frac{e^t(sk_4, c_3 \times c_3)}{e^t(sk_4, c_2)}
\]

where \( K = E_t(k) \) and \( k \leftarrow \text{Decrypt}‘(\mu, sk_{a2}, R) \).

Compute \( E_t^{-1}(M) \rightarrow m \| b \), output \( m \) while \( b = 0 \) and output \( \perp \) while \( b = 1 \).
3.3 Correctness

First we demonstrate the first-level ciphertext. Suppose

\[ M' = e_{sk_1, c3} e_{sk_1', c4} \]

Then

\[ M' = M \cdot e(g_1, g_2) \] \[ \frac{e(g', F_1(id'), F_1(id'), e(g', F_2(vk')))}{e(g_1^2, e_{sk_1, c3} e_{sk_1', c4} \cdot K, g')} \]

The proof technique of second-level ciphertext is similar to the proof technique of first-level ciphertext. Suppose \( M' = e_{sk_1, c3} e_{sk_1', c4} \)

Then

\[ M' = M \cdot e(g_1, g_2) \] \[ \frac{e(g', F_1(id'), F_1(id'), e(g', F_2(vk')))}{e\left(g_1^2, e_{sk_1, c3} e_{sk_1', c4} \cdot K, g'\right)} \]

3.4 Security

We now prove the security of our scheme.
Theorem 1. We say that an IB-PRE scheme is \((t,q_k,\delta)\)-IND-IBPRE-CCA secure, if for any \(t\) -time IND-IBPRE-CCA adversary \(A\) that makes at most \(q_k\) key generation queries, \(A\) has advantage \(\delta\) against the game. Then there is an algorithm \(B\) has 
\[ \text{Adv}^{\text{IND-IBPRE-CCA}} \left( \{ t \} \right) \geq \delta \text{ with } h \text{ of } e(1+q_k), \] 
where \(h\) denotes the base of the natural logarithm.

Setup. Algorithm \(B\) performs \(\text{Setup}(t')\) to get \((\mu,\text{mask})\) and give \(\mu\) to adversary \(A\).

Phase 1. \(A\) makes the following queries.

a) Key generation query \(O_k\). B maintains a table \((\beta, id, id, sk)\), where \(sk\) is re-encryption key. B first generates a random coin \(\beta\) so that \(\Pr[\beta=1]=\delta\). If \(\beta=0\), or \((0, id, \ast, \ast)\) or \((0, \ast, id, \ast)\) already exists on the table, \(B\) outputs a random bit and aborts. Otherwise, \(B\) issues \(\text{KeyGen}\) to get \(sk_\mu\) and returns it to \(A\). Then adds \((1, id, id, \ast)\) on the table.

b) Re-encryption key generation query \(O_{sk}\). B chooses a random coin \(\beta\), if \(\beta=1\), or \((1, id, id, \ast)\) or \((1, id, id, \ast)\) already exists on the table, \(B\) issues \(O_{sk}\) for \(id\), and computes the re-encryption key \(rk_{1^{2}2}\), then returns it to \(A\). Otherwise, \(B\) returns a random re-encryption key \(rk_{1^{2}2}=\left( x, y, \text{Encrypt}^{\prime}(\mu, id, z) \right)\), where \(x, y \in G\) and \(z \in \{ 0, 1 \} \). Finally, \(B\) records the tuple \((\beta, id, id, id, rk_{1^{2}2})\) on the table.

c) Re-encryption query \(O_{r}\). Let \(C_{id}=\left( c_{1}, c_{2}, c_{3}, c_{4}, vk, \sigma \right)\). If \((\ast, id, id, \ast)\) does not exist on the table, \(B\) issues \(O_{r}\) to get the re-encryption key \(rk_{1^{2}2}\). Then \(B\) uses \(rk_{1^{2}2}\) to re-encrypt the original ciphertext.

d) Decryption query \(O_{d}\). If \(C_{id}\) is a first-level ciphertext, \(B\) issues \(O_{d}\) to get the plaintext and return it to \(A\) if and only if it ends with ‘0’. If \(C_{id}\) is a second-level ciphertext, let \(C_{id}=\left( C_{id}, R, sk^{\ast}, sk, sk \right)\). If \(0, id, id, sk\) exists on the table for any \(id\) and \(sk=\langle sk_{1}, sk_{2}, sk_{3} \rangle\), \(B\) checks whether
\[ e(s_{k_{1}}^{\ast}, g) = e(s_{k_{1}}^{\ast}, g) \cdot e(F_{2}(vk), sk^{\ast}) \] and \(R=sk_{1}\).

If the equations hold, \(B\) sends \(C_{id}\) to the decryption query for \(id\), and returns the plaintext to \(A\) if and only if it ends with ‘0’. Otherwise, \(B\) sends \(R\) to the decryption query for \(id\) to get \(k=1\) and computes
\[ M \equiv c_{1} \cdot e(s_{k_{1}}, c_{3}^{\ast}c_{3})e(s_{k_{1}}^{\ast}, c_{4}) \]
where \(K=E_{2}(k)\). If \(E_{2}(M)=m=0\) for some \(m\), \(B\) returns \(m\) to \(A\). Otherwise, \(B\) returns \(\bot\).

Challenge. Once \(A\) decides phase 1 is over. It outputs a target identity \(id^\prime\), and two equal-length message \(m_{0}, m_{1}\) to \(B\). If \(1, id^\prime, id, k\) exists on the table for any \(id\) and \(k\), \(B\) outputs a random bit and aborts. Otherwise, \(B\) issues the challenge \((id^\prime, m_{0}, m_{1}=0)\) to get \(C'\) and returns it to \(A\).

Phase 2. The adversary \(A\) continues to issue queries as in phase 1, except the following restrictions.

a) \(A\) can not issue the key generation query (\(id^\prime\)) to obtain the target private key \(sk_{id^\prime}\).

b) \(A\) can not issue the re-encryption key generation query (\(id^\prime, id\)) if \(id^\prime\) appears in a previous key generation query.
c) A can not issue re-encryption query \((id',id',C')\) if \(id'\) appears in a previous key generation query.

d) A can issue decryption queries on neither \((C',id)\) nor \((id-Reencrypt(id',id',C'))\).

In this phase, Key generation query \(O_k\), Re-encryption query \(O_r\), and Decryption query \(O_d\) are the same as in phase 1.

In Re-encryption key generation query \(O_{rk}\). For all \(v_i \neq v'_i\), B issues \(O_k\) for \(v_i\), then computes the re-encryption key and return it to A. If \(v_i = v'_i\), B returns a random re-encryption key \(rk_{alt-sel}\) where \(x,y \in G\) and \(z \in G\). Finally, B records the tuple \((0,0,1,1)\) on the table.

**Guess.** Finally, A outputs a guess \(b' \in \{0,1\}\), and wins the game if \(b = b'\).

We can see that if B does not abort during the game, the view of A is identical to the real attack. So we only need to calculate the probability that B aborts during the game. Suppose A issues \(q_k\) key generation queries. The probability that B does not abort in phase 1 or phase 2 is \(\delta^{q_k}\). The probability that B does not abort during the challenge is \(1 - \delta\). The probability that B does not abort during the game is \(\delta^{q_k} (1 - \delta)\). The value is maximum while \(\delta = 1 - 1/(1 + q_k)\). Using the maximum value, the probability that B does not abort is at least \(1/\epsilon (1 + q_k)\). So B’s advantage is at least \(\epsilon (1 + q_k)\).

3.5 Comparison

Our IB-PRE scheme based on CT07 techniques. In our scheme, we get the security by using added secret parameter and change the secret key and re-encryption key. The structure and calculation of encryption, re-encryption, and decryption are the same to CT07, our scheme can get the same computational and communicational costs.

4. Conclusions

Our scheme has many applications, such as cloud computing, e-healthcare. In cloud computing, the server is not completely trusted. There will have some malicious users in cloud computing. Once the proxy and malicious user conspire, the user’s information will be reveal.

By using the CT07 techniques, we use bilinear pairing to construct a collusion-resistant IB-PRE scheme. We add the secret parameter, use this secret parameter construct new master secret key and decryption secret key. Our scheme can effectively prevent the collusion attack of proxy and delegatee. And our scheme is secure against CCA in the standard model.

Acknowledgement

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF2014) funded by the Ministry of Education, Science and Technology. And it also supported by the BB21 project of Busan Metropolitan City.

References


Authors

Junjie Qiu, he received his B.S. in education technology from Qufu Normal University, Qufu, China, in 2000, and his master’s degree in Computer Application Technology from Ocean University of China, Qingdao, China, in 2010. He is currently pursuing his PhD at Dongseo University, Busan, Korea. His research interests include public key cryptography and network security.

Hoon Jae Lee, he received the B.S., M.S. and Ph.D. degree in Electrical Engineering from Kyungpook national university in 1985, 1987 and 1998, respectively. He had been engaged in the research on cryptography and network security at Agency for Defense Development from 1987 to 1998. Since 2002 he has been working for Department of Computer Engineering of Dongseo University as an associate professor, and now he is a full professor. His current research interests are in security communication system, side-channel attack, USN & RFID security. He is a member of the Korea institute of Information security and cryptography, IEEE Computer Society, IEEE Information Theory Society and etc.