LETTER

Security of Cloud-Based Revocable Identity-Based Proxy Re-Encryption Scheme

Seunghwan PARK[†a)], ***Nonmember and*** **Dong Hoon LEE**[†b)], ***Member***

SUMMARY Designing secure revocable storage systems for a large number of users in a cloud-based environment is important. Cloud storage systems should allow its users to dynamically join and leave the storage service. Further, the rights of the users to access the data should be changed accordingly. Recently, Liang et al. proposed a cloud-based revocable identity-based proxy re-encryption (CR-IB-PRE) scheme that supports user revocation and delegation of decryption rights. Moreover, to reduce the size of the key update token, they employed a public key broadcast encryption system as a building block. In this paper, we show that the CR-IB-PRE scheme with the reduced key update token size is not secure against collusion attacks.

key words: *revocable identity-based encryption, key revocation, cloud-based identity-based proxy re-encryption, ciphertext update*

1. Introduction

Nowadays, a cloud storage service is an important infrastructure service that enables clients to store and share data with other users. To prevent information leakage due to the storage of sensitive data in a cloud storage service, a variety of cryptographic primitives have been studied with regard to cloud-based environments. A cloud-based environment should allow its users to dynamically join and leave the service. Therefore, cryptographic schemes for cloud storage systems should be constructed considering the user revocation problem. In 2012, Sahai, Seyalioglu, and Waters [7] introduced the notion of revocable-storage attribute based encryption (RS-ABE). This attribute-based encryption (ABE) is the first to support user revocation and ciphertext update such that the original ciphertext from an earlier time period T can be updated to a new ciphertext at a new time period $T + 1$ without leaking any plaintext information. However, the RS-ABE system allows anyone to update the ciphertext. This raises the problem that justifiable access of the non-revoked user to the data in the cloud storage could be interrupted by indiscriminate ciphertext updating by a malicious user, irrespective of the designated update time. This implies that the system needs to enable a designated party who is given the right to update the ciphertext by the key generation center. Recently, Liang et al. [4] proposed a cloud-based revocable identity-based proxy re-encryption (CR-IB-PRE) scheme. This scheme allows only a valid administrator to update the ciphertext with a re-encryption key for the new time period. In contrast to the previous revocable identity-based encryption (RIBE) scheme, this scheme employs public key broadcast encryption as a building block instead of tree-based revocation encryption as a combined structure to reduce the complexity of the key update phase. Unfortunately, this approach makes it insecure against a collusion attack in a security model, as an adversary can obtain key materials to generate the decryption key of the target ciphertext.

Identity-based encryption (IBE) and ABE schemes for cloud storage have both been considered as ways to ensure data confidentiality. The IBE scheme is a new public key encryption paradigm where the public key can be the identity string of the user, for instance, an e-mail address. In IBE systems, providing an efficient revocation mechanism for a large number of users is very important because a user's private key can be revealed or a user's credentials may expire. Boneh and Franklin [2] introduced a revocation method that represents an identity as $ID||T$, where ID is the original identity and T is the current time. Unfortunately, this method is not practical because the center needs to connect with all users by individual secure channels. After Boldyreva et al. [1] proposed the first scalable RIBE scheme, many RIBE schemes were proposed to improve the security and efficiency of the RIBE scheme; these schemes combined a variety of IBE schemes and the tree-based revocation encryption of Naor, Naor, and Lotspiech [5], where the center's workload increases logarithmically with an increase in the number of users [8]. Park et al. [6] presented a RIBE scheme with constant-sized private and update keys by using multilinear maps. Even though many RIBE and RABE schemes have been presented by various researchers, the existing RIBE and RABE schemes that support only key revocation are limited with respect to their application to cloud storage systems. In RIBE, a user that is revoked at time T can decrypt the ciphertexts that were encrypted before T because the decryption key of the revoked user is still available to decrypt these ciphertexts. To solve this problem, Sahai, Seyalioglu, and Waters [7] proposed an RS-ABE system for cloud-based environments. Lee et al. [3] then proposed an improved RS-ABE system and a revocable storage predicate encryption system. In this paper, we analyze the security of Liang et al.'s CR-IB-PRE scheme with the reduced key update token size.

Manuscript received February 9, 2016.
Manuscript revised March 17, 2016.
Manuscript publicized March 30, 2016.

[†]The authors are with the Center for Information and Security Technologies, Korea University, Korea.
a) E-mail: sgusa@korea.ac.kr
b) E-mail: donghlee@korea.ac.kr (Corresponding author)

DOI: 10.1587/transinf.2016EDL8042

2. Review of Liang et al.'s CR-IB-PRE Scheme

2.1 Bilinear Maps

Definition 2.1 (Bilinear Maps): Let $\mathbb{G}$ and $\mathbb{G}_T$ be two (multiplicative) cyclic groups of prime order p. We assume that g is a generator of $\mathbb{G}$. Let $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a function that has the following properties:

- Bilinearity: The map e satisfies the following relation: $e(g^a, g^b) = g^{ab} : \forall a, b \in \mathbb{Z}_p$
- Non-degeneracy: $e(g, g) \neq 1$.

Thus, we say that $\mathbb{G}$ is a bilinear group and the map e is a bilinear pairing in $\mathbb{G}$. Note that $e(,)$ is symmetric since $e(g^a, g^b) = e(g, g)^{ab} = e(g^b, g^a)$.

2.2 Basic CR-IB-PRE Scheme of Liang et al.

Setup$(1^\lambda, N)$**:** This algorithm takes as input a security parameter 1^λ and the maximum number of the users N. It generates bilinear groups $\vec{\mathbb{G}} = (\mathbb{G}, \mathbb{G}_T)$ of prime order p. Let g be the generator of $\mathbb{G}$ and $(p, \vec{\mathbb{G}}, e)$ be the description of bilinear groups. It selects random elements $g_2, g_3, v_1, v_2 \in \mathbb{G}$, random exponent $\alpha, \beta \in \mathbb{Z}_p$, random n-length set $U = \{u_j | 0 \leq j \leq n\}$, and TCR hash function $TCR_1 : \mathbb{G}_T \to \mathbb{Z}_p^*$, where $u_j \in \mathbb{G}$, and computes $g_1 = g^\alpha$. It outputs a master key $MK = (g_2^\alpha, g_3^\beta)$, an empty revocation list RL, an empty state ST, and public parameters as $PP = (g, g_1, g_2, g_3, v_1, v_2, U, TCR_1)$.

KeyGen(ID, MK, PP)**:** This algorithm takes as input an identity $ID \in \mathcal{I}$, master key MK, and public parameters PP. It selects a random exponent $g_3^\beta, r_{ID} \in \mathbb{Z}_p$ and outputs a private key by implicitly including ID as $SK_{ID} = (K_1 = g_3^\beta (u_0 \prod_{j \in v_{ID}} u_j)^{r_{ID}},\ K_2 = g^{r_{ID}})$.

TokenUp(T_i, RL, MK, ST, PP)**:** This algorithm takes as input time T, revocation list RL, master key MK, state ST, and public parameters PP. It selects a random exponent $r_T \in \mathbb{Z}_p$ and outputs token τ_T by implicitly including T as $\tau_T = (\tau_{T,1} = (g_2^\alpha / g_3^\beta \cdot (v_1 v_2^T)^{r_T},\ \tau_{T,2} = g^{r_T})$.

DeKeyGen(SK_{ID}, τ_T, PP)**:** This algorithm takes as input private key SK_{ID}, token τ_T, and public parameters PP. It selects random exponents $r_1, r_2 \in \mathbb{Z}_p$ and outputs decryption key $DK_{ID|T}$ by implicitly including T as $DK_{ID|T} = (D_1 = K_1 \cdot \hat{\tau}_{T,1} \cdot (u_0 \prod_{j \in v_{id}} u_j)^{r_1} \cdot (v_1 v_2^T)^{r_2} = g_2^\alpha \cdot (u_0 \prod_{j \in v_{ID}} u_j)^{\hat{r}_1} \cdot (v_1 v_2^T)^{\hat{r}_2},\ D_2 = K_2 \cdot g^{r_1} = g^{\hat{r}_1},\ D_3 = \hat{\tau}_{T,2} \cdot g^{r_2} = g^{\hat{r}_2})$ where $\hat{r}_1 = r_{ID} + r_1$, $\hat{r}_2 = r_{T_i} + r_2$. Note that the user shares r_1, r_2 with PKG so that PKG can store $(ID|i, \hat{r}_1, \hat{r}_2)$ in list $List^{SK_{ID|i}}$ for further use.

ReKeyGen$(MK, DK_{id|T}, T, T')$**:** This algorithm takes as input master key MK, decryption key $DK_{ID|T}$, time T, and another time T'.

1. $ReKeyGen(MK, T, T')$: It first generates re-encryption key token $v_{T \to T'}$ as $v_{T \to T'} = (v_{T \to T'}^1 = (v_1 \cdot v_2^{T'})^{TCR_1(\theta)} / (v_1 \cdot v_2^T)^{\hat{r}_2},\ v_{T \to T'}^2 = (\hat{C}_0, \hat{C}_1, \hat{C}_2, \hat{C}_3) \leftarrow Enc(ID, T', \theta))$. where $\theta \in_R \mathbb{G}_T$, $\hat{r}_2$ is recovered from $(ID|T', \hat{r}_1, \hat{r}_2)$, which is stored in $List^{SK_{ID|T}}$.
2. $ReKey(SK_{ID|T}, v_{T \to T'})$: It selects a random exponent $r' \in \mathbb{Z}_p$ and outputs re-encryption key $RK_{T \to T'}$ as $RK_{T \to T'} = (rk_1 = D_1 \cdot v_{T \to T'}^1 \cdot (u_0 \prod_{j \in v_{ID}} u_j)^{r'},\ rk_2 = D_2 \cdot g^{r'},\ rk_3 = v_{T \to T'}^2)$.

Encrypt(ID, T, M, PP)**:** This algorithm takes as input identity ID, time T, message M, and public parameters PP. It first chooses a random exponent $t \in \mathbb{Z}_p$ and outputs a ciphertext by implicitly including ID and T as $CT_{ID,T} = (C_0 = e(g_1, g_2)^t \cdot M,\ C_1 = g^t,\ C_2 = (u_0 \prod_{j \in v_{ID}} u_j)^t,\ C_3 = (v_1 \cdot v_2^{T'})^t)$.

ReEnc$(RK_{T \to Ti'}, CT_{ID,T_i})$**:** This algorithm takes as input re-encryption key $RK_{T \to T'}$ as $(rk_1, rk_2, rk3)$, and ciphertext $CT_{ID,T}$ as (C_0, C_1, C_2, C_3). It first computes $C_4 = e(C_1, rk_1)/e(C_2, rk_2) = e(g^t, g_2^\alpha \cdot (v_1, \cdot v_2^{T'})^{TCR_1(\theta)})$ and sets the re-encrypted ciphertext $CT_{ID,T'}$ to (C_0, C_1, C_4, rk_3). Note that if $CT_{ID,T'}$ needs to be further re-encrypted for time T'', then the proxy parses rk_3 as $(\hat{C}_0, \hat{C}_1, \hat{C}_2, \hat{C}_3)$. It takes as input the re-encryption key $RK_{T' \to T''}$ as (rk'_1, rk'_2, rk'_3), and the proxy then computes $C'_4 = e(\hat{C}_1, rk'_1)/e(\hat{C}_2, rk'_2)$ and sets ciphertext $CT_{ID,T''}$ to $(C_0, C_1, C_4, \hat{C}_0, \hat{C}_1, C'_4, rk'_3)$.

Decrypt$(CT_{ID,T}, DK_{ID,T}, PP)$**:** This algorithm takes as input ciphertext $CT_{ID,T} = (C_0, C_1, C_2, C_3)$, decryption key $DK_{ID,T} = (D_1, D_2, D_3)$, and public parameters PP.

1. For the original ciphertext, it computes $e(C_1, D_1) / e(C_2, D_2) e(C_3, D_3) = e(g_1, g_2)^t$, and outputs message $C_0 / e(g_1, g_2)^t = M$.
2. For a re-encrypted ciphertext, it does the following:
   - If the re-encrypted ciphertext is re-encrypted only once, i.e., $CT_{ID,T} = (C_0, C_1, C_4, \hat{C}_0, \hat{C}_1, \hat{C}_2, \hat{C}_3)$, then it computes $\hat{C}_0 \cdot e(\hat{C}_2, DK_2) e(\hat{C}_3, DK_3) / e(\hat{C}_1, DK_1) = \sigma$ and outputs message $C_0 \cdot e(C_1, (v_1 v_2^T)^{TCR_1(\sigma)}) / C_4 = M$.
   - If the ciphertext is re-encrypted l times from time T_1 to T_{l+1}, i.e., $CT_{ID,T_{l+1}} = (C_0^{(1)}, C_1^{(1)}, C_4^{(1)}, \ldots, C_0^{(l)}, C_1^{(l)}, C_4^{(l)}, rk_3^{(l+1)})$, then it first computes $C_0^{(l+1)} \cdot e(C_2^{(l+1)}, D_2) \cdot e(C_3^{(l+1)}, D_3) / e(C_1^{(l+1)}, D_1) = \sigma^l$, then computes $C_0^{(i)} \cdot e(C_1^{(i)}, (v_1 v_2^{T_{i+1}})^{TCR_1(\sigma^{(i)})}) / C_4^{(i)} = \sigma^{i-1}$ (from $i = 2$ to l), and finally outputs message $C_0^{(1)} \cdot e(C_1^{(1)}, (v_1 v_2^{T_2})^{TCR_1(\sigma^{(1)})}) / C_4^{(1)} = M$.

Revoke(ID, T, RL, st)**:** This algorithm takes as input identity ID, revocation time T, revocation list RL, and state ST. If $(ID, -) \notin ST$, then it outputs $\perp$ because the private key of ID has not been generated. Otherwise, it adds (ID, T) to RL. It then outputs the updated revocation list RL.

2.3 Reduce the Complexity of Key Update

Let $SYM = (SYM.Enc, SYM.Dec)$ denote a one-time symmetric encryption system.

Setup$(1^\lambda, N)$**:** This algorithm additionally chooses $\gamma, \hat{\alpha} \in_R \mathbb{Z}_p$ and TCR hash function $TCT_2 : \mathbb{G}_T \rightarrow \{0,1\}^{poly(1^k)}$, and adds $v_0 = g^\gamma$ and TCR_2 to public parameters PP, and $(\gamma, \hat{\alpha})$ to master key MK.

KeyGen(ID, MK, PP)**:** This algorithm generates a new key component $K_3 = g_z^\gamma$ and sets additional public parameters $g_z = g^{\hat{\alpha}^z}, g_{z+1} = g^{\hat{\alpha}^{z+1}}, g_{N+2-z} = g^{\hat{\alpha}^{N+2-z}}, g_{N+2+z} = g^{\hat{\alpha}^{N+2+z}}$ for user ID, where z represents the index for identity ID.

TokenUp(T, RL, MK, ST, PP)**:** This algorithm first defines a non-revoked set S of user identities at time T from RL. After constructing token τ_T, it chooses $\hat{t} \in_R \mathbb{Z}_q^*, K \in_R \mathbb{G}_T$ and sets $\hat{\tau}_T = (\hat{\tau}_{T,1} = K_{SYM} \cdot e(g_{N+2}, g)^{\hat{t}}, \hat{\tau}_{T,2} = g^{\hat{t}}, \hat{\tau}_{T,3} = (v_0 \prod_{w \in S} g_{N+2-w})^{\hat{t}}, \hat{\tau}_{T,4} = SYM.Enc(TCR_2(K_{SYM}), \tau_{T,1} \| \tau_{T,2}))$.

DeKeyGen$(SK_{ID}, \hat{\tau}_T, PP)$**:** Before constructing the decryption key as in the *DeKeyGen* algorithm of the basic scheme, this *DeKeyGen* algorithm derives the token as $K_{SYM} = (\hat{\tau}_{T,1} \cdot e(\hat{\tau}_{T,3}, g_i) / e(K_3 \cdot \prod_{w \in R \setminus \{z\}} g_{N+2-w+z}, \hat{\tau}_{T,2}))$ and runs $\tau_{T,1} \| \tau_{T,2} = SYM.Dec\ (TCR_2(K_{SYM}), \hat{\tau}_{T,4})$.

The rest of the algorithms are the same as those of the basic scheme.

3. Our Security Analysis

3.1 Security Model of the CR-IB-PRE Scheme

The security model of CR-IB-PRE was introduced by Liang et al. [4]. In this security model, the update token query takes as input identity ID and time T. If $ID \neq ID^*$, the update token oracle outputs a valid update token. Otherwise, the update token oracle outputs $\perp$. However, this is not realistic in practice because the update tokens are generated regardless of identity ID in the real world. (The update tokens are generated by implicitly including time T and revocation list RL.) This condition restricts the adversary from obtaining more information about the private key and update token. In this paper, we propose a CR-IB-PRE security model that is a refined version of Liang et al.'s security model. The security of CR-IB-PRE is formally defined as follows:

Definition 3.1 (Security): The security of a CR-IB-PRE scheme under chosen plaintext attacks is defined in terms of the following experiment between a challenger $\mathcal{C}$ and probabilistic polynomial time (PPT) adversary $\mathcal{A}$:

1. **Setup**: $\mathcal{C}$ generates master key MK, revocation list RL, state ST, and public parameters PP by running **Setup**$(1^\lambda, N)$. It keeps MK, RL, ST to itself and gives PP to $\mathcal{A}$.
2. **Phase 1**: $\mathcal{A}$ adaptively requests a polynomial number of queries. These queries are processed as follows:
   - If this is a private key query for identity ID, then $\mathcal{C}$ gives the corresponding private key SK_{ID} to $\mathcal{A}$ by running **KeyGen**(ID, MK, ST, PP).
   - If this is an update token query for time T, then $\mathcal{C}$ gives the corresponding update key $UK_{T,R}$ to $\mathcal{A}$ by running **TokenUp**(T, RL, MK, ST, PP).
   - If this is a decryption key query for identity ID and time T, then $\mathcal{C}$ gives the corresponding decryption key $DK_{ID,T}$ to $\mathcal{A}$ by running **DeKeyGen**$(SK_{ID}, UK_{T,R}, PP)$.
   - If this is a re-encryption key query for identity ID, time T, and the next time T', then $\mathcal{C}$ gives the corresponding re-encryption key $RK_{T \rightarrow T'}$ to $\mathcal{A}$ by running **ReKeyGen**(MK, ID, T, T').
   - If this is a revocation query for an identity ID and a revocation time T, then $\mathcal{C}$ updates the revocation list RL by running **Revoke**(ID, T, RL, ST) with the restriction: The revocation query for a time T cannot be queried if the update key query for the time T was already requested.

   Note that $\mathcal{A}$ is allowed to request the update token query and the revocation query in a non-decreasing order of time, and an update key $UK_{T,R}$ implicitly includes a revoked identity set R derived from RL.
3. **Challenge**: $\mathcal{A}$ submits a challenge identity ID^*, challenge time T^*, and two challenge messages $M_0^*, M_1^* \in \mathcal{M}$ with equal length with the following restrictions:
   - If a private key query for an identity ID such that $ID = ID^*$ was requested, then identity ID^* should be revoked at some time T such that $T \leq T^*$.
   - The decryption key query for ID^* and T^* was not requested.

   $\mathcal{C}$ flips a random coin $b \in \{0,1\}$ and gives the challenge ciphertext CT^* to $\mathcal{A}$ by running **Encrypt**(ID^*, T^*, M_b^*, PP).
4. **Phase 2**: $\mathcal{A}$ may continue to request a polynomial number of private keys, update tokens, decryption keys, and re-encryption keys subject to the same restrictions as before.
5. **Guess**: Finally, $\mathcal{A}$ outputs guess $b' \in \{0,1\}$ and wins the game if $b = b'$.

The advantage of $\mathcal{A}$ is defined as $\mathbf{Adv}_{CR-IB-PRE,\mathcal{A}}^{IND\text{-}CPA}(\lambda) = \left| \Pr[b = b'] - \frac{1}{2} \right|$, where the probability is taken over the entire randomness of the experiment. A CR-IB-PRE scheme is secure in this security model against the chosen plaintext attacks if, for all PPT adversaries $\mathcal{A}$, the advantage of $\mathcal{A}$ for security parameter λ in the above experiment is negligible.

3.2 Analysis of Liang et al.'s CR-IB-PRE Scheme

The following lemma shows that there is a PPT algorithm that can distinguish challenge message M_b with non-negligible probability.

Lemma 3.2: There exists a PPT algorithm $\mathcal{A}$ that can distinguish whether challenge ciphertext CT^* is encrypted from message M_0 or M_1 if $\mathcal{A}$ generates the private key query for the non-revoked identity ID' at challenge time T^* after the revocation query for identity ID' at some time T' such that $T' > T^*$.

Proof 1: We prove that the collusion attack presented here is valid in the security model. Consider the security game between simulator $\mathcal{B}$ and adversary $\mathcal{A}$ for $IND-CPA$ security; the detailed process is as follows:

- In the setup, $\mathcal{B}$ first creates public parameters PP and gives PP to $\mathcal{A}$.
- In phase 1, $\mathcal{A}$ can adaptively request a polynomial number of private keys, update keys, decryption keys, and revocation queries. $SK_{ID^*} = (K_1^* = g_3^{\beta}(u_0 \prod_{j \in v_{ID}} u_j)^{r_{ID}},\ K_2^* = g^{r_{ID^*}},\ K_3^* = g_{z^*}^{\gamma})$.
- In the challenge step, $\mathcal{A}$ selects a challenge identity ID^* such that $(ID^*, \hat{T}) \in RL$ for any $\hat{T} \leq T^*$ and a pair of random messages (M_0, M_1), and gives (ID^*, T^*, M_0, M_1) to $\mathcal{B}$. That is, challenge identity ID^* is revoked before time T^* and $\mathcal{A}$ may query private key SK_{ID^*} for identity ID^*. Upon receiving the message from $\mathcal{A}$, $\mathcal{B}$ randomly picks M_b for $b \in \{0,1\}$, computes the challenge ciphertext $CT^* = \mathbf{Encrypt}(ID^*, T^*, M_b, PP)$, and sends it to $\mathcal{A}$. $CT^* = (C_0^* = e(g_1, g_2)^t \cdot M_b,\ C_1^* = g^t,\ C_2^* = (u_0 \prod_{j \in v_{ID^*}} u_j)^t,\ C_3^* = (v_1 \cdot v_2^{T^*})^t)$.
- In phase 2, $\mathcal{A}$ can continue to request a polynomial number of private keys, update keys, decryption keys, and revocation queries. We define the non-revoked set S^* of user identities at time T^* from RL. $\mathcal{A}$ first queries update token $\hat{\tau}_{T^*}$ or time T^* and non-revoked set S^* as $\tau_{T^*} = (\tau_{T^*,1} = (g_2^{\alpha}/g_3^{\beta} \cdot (v_1 v_2^{T^*})^{r_{T^*}},\ \tau_{T^*,2} = g^{r_{T^*}})$, $\hat{\tau}_{T^*} = (\hat{\tau}_{T^*,1} = K_{SYM} \cdot e(g_{N+2}, g)^{\hat{t}},\ \hat{\tau}_{T^*,2} = g^{\hat{t}},\ \hat{\tau}_{T^*,3} = (v_0 \prod_{w \in S^*} g_{N+2-w})^{\hat{t}},\ \hat{\tau}_{T^*,4} = SYM.Enc(TCR_2(K_{SYM}, \tau_{T^*,1} \| \tau_{T^*,2}))$. Next, $\mathcal{A}$ requests the revocation query for (ID', T') such that $ID' \in S^*$ and $T' > T^*$ and private key $SK_{ID'}$ for the identity ID' as $SK_{ID'} = (K_1' = g_3^{\beta}(u_0 \prod_{j \in v_{ID'}} u_j)^{r_{ID'}},\ K_2' = g^{r_{ID'}},\ K_3' = g_{z'}^{\gamma})$.
- In the guess step, $\mathcal{A}$ proceeds as follows:
  1. $\mathcal{A}$ parses the update token $\hat{\tau}_{T^*}$ under (T^*, S^*) as $(\hat{\tau}_{T^*,1}, \hat{\tau}_{T^*,2}, \hat{\tau}_{T^*,3}, \hat{\tau}_{T^*,4})$ and private key $SK_{ID'}$ under ID' as (K_1', K_2', K_3'). It then computes $K_{SYM} = (\hat{\tau}_{T^*,1} \cdot e(\hat{\tau}_{T^*,3}, g_i)/e(K_3' \cdot \prod_{w \in R \setminus \{z\}} g_{N+2-w+z}, \hat{\tau}_{T^*,2}))$.
  2. $\mathcal{A}$ runs $\tau_{T^*,1} \| \tau_{T^*,2} = SYM.Dec(TCR_2(K_{SYM}), \hat{\tau}_{T^*,4})$.
  3. $\mathcal{A}$ parses the private key SK_{ID^*} under ID^* as (K_1^*, K_2^*, K_3^*). It then computes decryption key $DK_{ID^*|T^*}$ as $DK_{ID^*|T^*} = (D_1^* = (K_1^* \cdot \hat{\tau}_{T^*,1} \cdot (u_0 \prod_{j \in v_{ID^*}} u_j)^{r_1} \cdot (v_1 v_2^{T^*})^{r_2} = g_2^{\alpha} \cdot (u_0 \prod_{j \in v_{ID^*}} u_j)^{\hat{r}_1} \cdot (v_1 v_2^{T^*})^{\hat{r}_2},\ D_2^* = K_2^* \cdot g^{r_1} = g^{\hat{r}_1},\ D_3^* = \hat{\tau}_{t^*,2} \cdot g^{r_2} = g^{\hat{r}_2})$
  4. $\mathcal{A}$ parses the ciphertext CT^* under (ID^*, T^*) as $(C_0^*, C_1^*, C_2^*, C_3^*)$, computes $e(C_1^*, D_1^*)/e(C_2^*, D_2^*)$ $e(C_3^*, D_3^*) = e(g_1, g_2)^t$, and outputs the message $C_0^*/e(g_1, g_2)^t = M_b$.
  5. Finally, $\mathcal{A}$ determines index b of obtained message M_b and outputs b.

Adversary $\mathcal{A}$'s behavior is valid. It is clear that $\mathcal{A}$ always wins the game.

4. Conclusion

In this paper, we showed that the CR-IB-PRE scheme [4] is vulnerable to collusion attacks whereby a non-revoked user's private key is revealed. How to construct the CR-IB-PRE scheme with a constant number of private key and update key elements remains an open problem.

Acknowledgements

This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIP) (No.R0126-16-1090,A study of a public-key authentication framework for internet entities with hierarchical identities). Dong Hoon Lee was supported by research grant funded by the Korea University.

References

[1] A. Boldyreva, V. Goyal, and V. Kumar, "Identity-based encryption with efficient revocation," In P. Ning, P.F. Syverson, and S. Jha, editors, ACM Conference on Computer and Communications Security, pp.417–426, ACM, 2008.

[2] D. Boneh and M.K. Franklin, "Identity-based encryption from the weil pairing," In J. Kilian, editor, CRYPTO 2001, volume 2139 of Lecture Notes in Computer Science, pp.213–229, Springer, 2001.

[3] K. Lee, S.G. Choi, D.H. Lee, J.H. Park, and M. Yung, "Self-updatable encryption: Time constrained access control with hidden attributes and better efficiency," In K. Sako and P. Sarkar, editors, ASIACRYPT 2013, volume 8269 of Lecture Notes in Computer Science, pp.235–254, Springer, 2013.

[4] K. Liang, J.K. Liu, D.S. Wong, and W. Susilo, "An efficient cloud-based revocable identity-based proxy re-encryption scheme for public clouds data sharing," In M. Kutylowski and J. Vaidya, editors, ESORICS 2014, volume 8712 of Lecture Notes in Computer Science, pp.257–272, Springer, 2014.

[5] D. Naor, M. Naor, and J. Lotspiech, "Revocation and tracing schemes for stateless receivers," In J. Kilian, editor, CRYPTO 2001, volume 2139 of Lecture Notes in Computer Science, pp.41–62, Springer, 2001.

[6] S. Park, K. Lee, and D.H. Lee, "New constructions of revocable identity-based encryption from multilinear maps," IEEE Trans. Inf. Forensics Security, vol.10, no.8, pp.1564–1577, 2015.

[7] A. Sahai, H. Seyalioglu, and B. Waters, "Dynamic credentials and ciphertext delegation for attribute-based encryption," In R. Safavi-Naini and R. Canetti, editors, CRYPTO 2012, volume 7417 of Lecture Notes in Computer Science, pp.199–217, Springer, 2012.

[8] J.H. Seo and K. Emura, "Revocable identity-based encryption revisited: Security model and construction," In K. Kurosawa and G. Hanaoka, editors, PKC 2013, volume 7778 of Lecture Notes in Computer Science, pp.216–234, Springer, 2013.