On the security of a Certificateless Proxy Re-Encryption Scheme without Pairing

Arinjita Paul, S. Sharmila Deva Selvi and C. Pandu Rangan

Theoretical Computer Science Lab,
Department of Computer Science and Engineering,
Indian Institute of Technology Madras, Chennai, India.
{arinjita,sharmila,prangan}@cse.iitm.ac.in

Abstract. Proxy re-encryption (PRE) is a cryptographic primitive introduced by Blaze, Bleumer and Strauss [2] to provide delegation of decryption rights. A semi-trusted proxy agent re-encrypts ciphertexts under the public key of Alice into ciphertexts under the public key of Bob, without learning anything about the underlying message. In IWSEC 2017, Kuchta *et al.* presented a pairing-free certificateless proxy re-encryption scheme, and claimed that their scheme is the first to provide the certificateless property without resorting to pairing. They proved their construction is CCA-secure in the random oracle model, under the Computational Diffie-Hellman assumption. In this work, we show that the recently proposed construction of Kuchta *et al.* is vulnerable to several attacks.

Keywords: Proxy Re-Encryption, Pairing-free, Public Key, Conditional, Unidirectional.

1 Analysis of the Certificateless PRE Scheme in IWSEC 2017

We first give an overview of the CL-PRE scheme due to Kuchta *et al.* and later describe our attacks against the confidentiality of their construction.

1.1 Review of the scheme

– **Setup**(1^λ): On input of a security parameter λ, the KGC chooses a cyclic group $\mathbb{G}$ of prime order q. It selects $s \in_R \mathbb{Z}_q^*$, sets the master secret key $msk = s$. It computes the master public key $mpk = y = g^s$, where $g \in \mathbb{G}$ is a generator of $\mathbb{G}$. It chooses the following cryptographic hash functions:

$$\begin{aligned}
H_1 &: \{0,1\}^* \times \mathbb{G} \rightarrow \mathbb{Z}_q^*,\\
H_2 &: \{0,1\}^* \times \mathbb{G}^3 \rightarrow \mathbb{Z}_q^*,\\
H_3 &: \mathbb{G}^4 \times \{0,1\}^* \rightarrow \mathbb{Z}_q^*,\\
H_4 &: \mathbb{G}^2 \times \{0,1\}^* \times \{0,1\}^* \rightarrow \mathbb{Z}_q^*,\\
H_5 &: \mathbb{G}^4 \rightarrow \mathbb{Z}_q^*,\\
H_6 &: \{0,1\}^m \times \{0,1\}^n \times \{0,1\}^* \times \mathbb{G}^2 \rightarrow \mathbb{Z}_q^*,\\
H_7 &: \mathbb{G} \rightarrow \{0,1\}^{m+n}.
\end{aligned}$$

Here, m and n are bit lengths. It return the public parameters $params = (\mathbb{G}, q, y, g, m, n, H_1, H_2, H_3, H_4, H_5, H_6, H_7)$ and the master secret key $msk = s$.

– **PartKeyExtr**$(params, msk, ID_A)$: On input of the public parameters $params = (\mathbb{G}, q, y, g, m, n, H_1, H_2, H_3, H_4, H_5, H_6, H_7)$, master secret key $msk = s$ and an identity ID_A of a user, the KGC generates the partial keys as follows:

- Pick $\alpha_{A,11}, \alpha_{A,12}, \beta \in \mathbb{Z}_q^*$.
- Compute $a_{A,11} = g^{\alpha_{A,11}}$, $a_{A,12} = g^{\alpha_{A,12}}$ and $a_2 = g^{\beta}$.
- Compute $x_{A,11} = \alpha_{A,11} + sH_1(ID_A, a_{A,11})$, $x_{A,12} = \alpha_{A,12} + sH_1(ID_A, a_{A,12})$ and $x_2 = \beta + sH_2(ID_A, a_{A,11}, a_{A,12}, a_2)$.
- Return the partial private key $psk_A = (x_{A,11}, x_{A,12})$ and the partial public key $ppk_A = (a_{A,11}, a_{A,12}, a_2, x_2)$.

– **KeyGen**$(params, ppk_A, ID_A)$: On input the public parameters $params$, partial public key $ppk_A = (a_{A,11}, a_{A,12}, a_2, x_2)$ and an identity ID_A, the KGC computes the user keys as below:
- Select $z_{A1}, z_{A2}, \gamma \in \mathbb{Z}_q^*$.
- Compute $u_{A1} = g^{z_{A1}}$, $u_{A2} = g^{z_{A2}}$, $a_3 = g^{\gamma}$, $t = \gamma + x_2 H_3(ID_A, u_{A1}, u_{A2}, a_2, a_3)$.
- Return the user secret key $sk_A = (z_{A1}, z_{A2})$ and user public key $pk_A = (u_{A1}, u_{A2}, t, a_3)$ to the user through a secure channel.

– **SetPrivatValue**$(params, sk_A, psk_A)$: On input the public parameters $params$, user secret key $sk_A = (z_{A1}, z_{A2})$ and partial secret key $psk_A = (x_{A,11}, x_{A,12})$, the user A sets its full secret key as below:

$$SK_A = (z_{A1}, z_{A2}, x_{A,11}, x_{A,12}).$$

– **SetPublicValue**$(params, pk_A, ppk_A)$: On input the public parameters $params$, user public key $pk_A = (u_{A1}, u_{A2}, t, a_3)$ and partial public key $ppk_A = (a_{A,11}, a_{A,12}, a_2, x_2)$, the user sets its full secret key as below:

$$PK_A = (u_{A1}, u_{A2}, a_{A,11}, a_{A,12}, a_2, x_2, t, a_3).$$

– **ReEncKey**$(params, ID_A, ID_B, SK_A, PK_B)$: On input of the parameters $params$, an identity ID_A of user A, and identity ID_B of user B, secret key SK_A of user A and public key PK_B of user B, the user A computes the re-encryption key as below:
- Compute $t_{Bi} = a_{B,1i} y^{H_1(ID_B, a_{B,1i})}$ for $i \in \{1, 2\}$.
- Compute $t_{AB} = H_4(t_{B1}^{z_{A1}}, u_{B1}^{x_{A,11}}, ID_A, ID_B)$.
- Compute $RK_{A \to B} = (x_{A,11} + z_{A1}) + (x_{A,12} + z_{A2}) H_5(u_{A1}, u_{A2}, a_{A,11}, a_{A,12})$.
- Return the re-encryption key $RK_{A \to B}$.

– **PubKeyVer**$(params, psk_A, PK_A)$: On input the public parameters $params$, partial secret key $psk_A = (x_{A,11}, x_{A,12})$ and full public key $PK_A = (u_{A1}, u_{A2}, a_{A,11}, a_{A,12}, a_2, x_2, t, a_3)$ of user A, the partial private keys, partial public keys and public keys are verified as below:

$$g^{x_{A,11}} \stackrel{?}{=} a_{A,11} y^{H_1(ID_A, a_{A,11})}, \; g^{x_{A,12}} \stackrel{?}{=} a_{A,12} y^{H_1(ID_A, a_{A,12})}, \tag{1}$$

$$g^{x_2} \stackrel{?}{=} a_2 y^{H_2(ID_A, a_{A,11}, a_{A,12}, a_2)}. \tag{2}$$

$$g^{t} \stackrel{?}{=} a_3 a_2^{H_3(ID_A, u_{A1}, u_{A2}, a_2, a_3)} y^{H_2(ID_A, a_{A,11}, a_{A,12}, a_2) H_3(ID_A, u_{A1}, u_{A2}, a_2, a_3)}. \tag{3}$$

– **Encrypt**$(params, ID_A, PK_A, m)$: On input the public parameter $params$, an identity ID_A, a public key PK_A and a message $m \in \{0,1\}^m$, compute the ciphertext as below:
- Pick $\sigma \in_R \{0,1\}^n$.
- Compute $r = H_6(M, \sigma, ID_A, u_{A1}, u_{A2})$.
- Compute $C_1 = g^r$.
- Compute $C_2 = (m || \sigma) \oplus H_7((t_{A1} u_{A1} t_{A2} u_{A2})^{H_5(u_{A1}, u_{A2}, a_{A,11}, a_{A,12})^r})$.
- Return the ciphertext $C = (C_1, C_2)$.

– **ReEncrypt**$(params, ID_A, ID_B, C, RK_{A\to B})$:On input of the public parameter $params$, identities ID_A and ID_B, a ciphertext C and a re-encryption key $RK_{A\to B}$, the proxy agent computes the re-encrypted ciphertext as below:
  - Compute $C_1' = C_1^{RK_{A\to B}}$.
  - Set $C_2' = C_2$.
  - Return the re-encrypted ciphertext $C' = (C_1', C_2')$.

– **Decrypt1**$(params, ID_A, C, SK_A)$: On input $params$, an identity ID_A, a first level ciphertext C and secret key $SK_A = (z_{A1}, z_{A2}, x_{A,11}, x_{A,12})$, decrypt the ciphertext by computing as below:

$$(m||\omega) = C_2 \oplus H_7(C_1^{(x_{A,11}+z_{A1})+(x_{A,12}+z_{A2})H_5(u_{A1},u_{A2},a_{A,11},a_{A,12})}).$$

Compute $r = H_6(m, \sigma, ID_A, u_{A1}, u_{A2})$ and check if $C_1 \stackrel{?}{=} g^r$. If the condition is satisfied, return the message m.

– **Decrypt2**$(params, ID_B, C', SK_B)$: On input $params$, an identity ID_B, a second level ciphertext C' and secret key $SK_B = (z_{B1}, z_{B2}, x_{B,11}, x_{B,12})$, decrypt the ciphertext by computing as below:

$$m||\sigma = C_2' \oplus H_7((C')^{\frac{1}{t_{BA}}}),$$

where $t_{BA} = H_4(u_{A1}^{x_{B,11}}, t_{A1}^{z_{B1}}, ID_A, ID_B)$.

1.2 Our Attacks

We enumerate attacks that imply that the scheme due to Kuchta *et al.* is not secure. The attacks are demonstrated below:

1. *Key-escrow*: From the definition of certificateless encryption scheme [1], we note that, the the task of key-generation is split between the two entities : a Key Generation Center (KGC) and the user himself, to prevent the KGC from having access to secret keys of the user. This technique addresses the key-escrow problem inherent in the IBE setting. However, in the certificateless scheme due to Kuchta *et al.*, both the partial keys (psk_i, ppk_i) and the user keys (pk_i, sk_i) are generated by the KGC alone and then transferred to the user through a secure channel. An unconditional trust placed on the KGC makes the scheme vulnerable to the key-escrow problem, where a malicious KGC possessing the secret keys of all users can decrypt any ciphertext of its choice. This clearly violates the concept of a certificateless system. The scheme is vulnerable to $IND\text{-}CLPRE\text{-}CCA$ attack by a Type-II adversary, who represents a KGC who has a knowledge of the master secret key msk.

2. *CCA attack*: Let $C_A^* = (C_1^*, C_2^*)$ be a first-level challenge ciphertext under a target identity ID_A^*. Due to the absence of public verification of ciphertexts, we can mount the following malleability attack:
   (a) Construct a first-level ciphertext $C_A = (C_1, C_2)$ from the challenge ciphertext C_A^*, by computing $C_1 = (C_1^*)^{r_1}$, where $r_1 \in \mathbb{Z}_q^*$ is chosen by the adversary, and $C_2 = C_2^*$. Note that, the first level ciphertext C_A is a valid construction of a ciphertext under the target identity ID_A, owing to the malleability of the challenge ciphertext C_A^*.
   (b) Send a re-encryption key generation query for a re-encryption key $RK_{A^*\to j}$, where ID_j is an honest user.
   (c) Construct a second-level ciphertext $C_j' = (C_1', C_2')$, where $C_1' = (C_1^{RK_{A^*\to j}})^{\frac{1}{r_1}}$ and $C_2' = C_2$.
   (d) Query the *Decryption*2 oracle for the decryption of C_j' under the identity ID_j. Note that this is permitted as per the security model, since C_A is no longer a challenge ciphertext.

(e) The adversary gets the message m_δ from the output of the *Decryption*2 oracles.
(f) Thus, the adversary can break the *CCA* security without having access to the secret keys or without solving any hard problem.

3. *CCA* attack: Let $C_A^* = (C_1^*, C_2^*)$ be a challenge ciphertext under a target identity ID_A^*. Due to the absence of public verification of ciphertexts, we can mount another malleability attack as below:
   (a) Construct a first-level ciphertext $C_A = (C_1, C_2)$ from the challenge ciphertext C_A^* by computing $C_1 = (C_1^*)^{r_1}$, where $r_1 \in \mathbb{Z}_q^*$ is chosen by the adversary, and $C_2 = C_2^*$. Note that, the first level ciphertext is a valid construction of a ciphertext under the target identity ID_A owing to the malleability of the challenge ciphertext C_A^*.
   (b) Send a re-encryption query from ID_{A^*} to ID_j with the first level ciphertext C_A as input, where ID_j is a corrupt user. The second-level ciphertext $C'_A = (C'_1, C'_2)$, where $C'_1 = C_1^{RK_{A^* \to j}}$ and $C'_2 = C_2$ is output by the re-encryption oracle.
   (c) Compute $C''_1 = {C'_1}^{\frac{1}{r_1}}$.
   (d) Decrypt the second-level ciphertext $C'_A = (C''_1, C'_2)$ using the secret keys of the corrupt identity ID_j known to the adversary. Note that this is permitted as per the security model, since C_A is no longer a challenge ciphertext.
   (e) The adversary decrypts C'_A using the secret keys of ID_j and gets the message m_δ. Thus,the adversary breaks the CCA security.

4. *CCA*-attack: We report a typo in the *Re-Encrypt* algorithm in the computation of the second level ciphertext component C'_1, which should be $C'_1 = C_1^{RK_{A \to B} \cdot t_{AB}}$. The *ReEncKey* algorithm must return the components $RK_{A \to B}$ and t_{AB} as the re-encryption keys for ID_A to ID_B. However, an adversary can mount the following CCA attack as below:
   (a) Let C_A^* be a first level challenge ciphertext under the target identity ID_A^*. The adversary queries for a re-encryption key $RK_{A^* \to j}$, where the identity ID_j is an honest user.
   (b) Re-encrypt the challenge ciphertext C_A^* into a second level ciphertext $C'_j = (C'_1, C'_2)$ under the identity ID_j using the re-key $RK_{A^* \to j}$.
   (c) Extract the message m_δ by computing $C'_2 \oplus (C_1)^{\frac{1}{t_{A^* j}}}$, where $t_{A^* j}$ is available with ID_j from the re-encryption key-generation query.
   Note that, this can be avoided by computing the re-encryption key from ID_A to ID_B as $RK_{A \to B} = ((x_{A,11} + z_{A1}) + (x_{A,12} + z_{A2}) H_5(u_{A1}, u_{A2}, a_{A,11}, a_{A,12})) \cdot t_{AB}$.

5. Another drawback of the scheme is that the public key verification algorithm $PubKeyVer(params, psk_A, PK_A)$ requires any user to possess the partial secret key psk_A of the user with ID_A inorder to verify the public keys. However, only the user himself and the KGC has knowledge of the partial secret keys. An adversary can replace the public keys with dummy keys of its choice, and a sender has no choice to verify the correctness of the public keys of identity ID_A for encrypting a message for ID_A. Abiding by the definitions of certificateless PRE, if we consider the user key generation algorithm $KeyGen(params, ppk, ID_A)$ to be run by the user himself, a Type-I adversary can mount a CCA attack on the scheme as follows. The adversary $\mathcal{A}$ selects an identity ID_A on which it wishes to attack. Let the public key of the identity ID_A be $(u_{A1}, u_{A2}, a_{A,11}, a_{A,12}, a_2, x_2, t, a_3)$.
   (a) The adversary $\mathcal{A}$ replaces the public keys of the identity ID_A with new public keys computed as follows:
      – Pick $r_1, r_2 \in \mathbb{Z}_q^*$.
      – Compute $u'_{A1} = g^{r_1} \cdot t_{A1}^{-1}$, where t_{A1} is defined as $a_{A,11} \cdot y^{H_1(ID_A, a_{A,11})}$.
      – Compute $u'_{A2} = g^{r_2} \cdot t_{A2}^{-1}$, where t_{A2} is defined as $a_{A,12} \cdot y^{H_1(ID_A, a_{A,12})}$.
      – Choose $\delta' \in_R \mathbb{Z}_q^*$. Compute $a'_3 = g^{\delta'}$ and $t' = \delta' + x_2 H_3(ID_A^*, u'_{A1}, u'_{A2}, a_2, a_3)$.
      – Replace the public keys of the identity ID_A^* by placing a public key replacement query with the new public key $(u'_{A1}, u'_{A2}, a_{A,11}, a_{A,12}, a_2, x_2, t', a'_3)$, where the key components $a_{A,11}, a_{A,12}, a_2$ and x_2 remain unchanged. Consequently, the new public key computed by

the adversary is valid as per the $PubKeyGen$ algorithm as it satisfies the equations (1), (2) and (3).

(b) The adversary outputs ID_A as the target identity in the challenge phase and two messages m_0, m_1 to the challenger. The challenger picks $\delta \in \{0,1\}$ uniformly at random, computes the challenge ciphertext C_δ encrypting m_δ and returns C_δ to the adversary.

(c) The adversary decrypts the challenge ciphertext to extract message m_δ by computing as follows:

$$\begin{aligned} m_\delta||\sigma &= C_2^* \oplus H_7\Big(C_1^{(x'_{A,11}+z'_{A1})+(x'_{A,12}+z'_{A2})H_5(u'_{A1},u'_{A2},a_{A,11},a_{A,12})}\Big) \\ &= (m_\delta||\sigma) \oplus H_7\Big(\big(t_{A1}u_{A1}(t_{A2}u_{A2})^{H_5(u'_{A1},u'_{A2},a_{A,11},a_{A,12})}\big)^r\Big) \oplus H_7\Big(C_1^{(r_1+r_2H_5(u'_{A1},u'_{A2},a_{A,11},a_{A,12}))}\Big) \\ &= (m_\delta||\sigma) \oplus H_7\Big(g^{r_1}g^{r_2H_5(u'_{A1},u'_{A2},a_{A,11},a_{A,12})}\Big)^r \oplus H_7\Big(C_1^{(r_1+r_2H_5(u'_{A1},u'_{A2},a_{A,11},a_{A,12}))}\Big) \\ &= m_\delta||\sigma. \end{aligned}$$

2 Conclusion

Although several certificateless PRE schemes have been proposed in the literature, to the best of our knowledge, two schemes have reported the certificateless property without pairing. One of the schemes is due to Kuchta *et al.* [3], which is vulnerable to several attacks as demonstrated in this report. We remark that the flaws in the scheme cannot be fixed trivially. Recently, Sharmila *et al.* [4] proposed a CLPRE scheme without resorting to bilinear pairing in the random oracle model. To the best of our knowledge, the scheme due to Sharmila *et al.* is the only certificateless PRE scheme that affirmatively resolves the problems faced by PKI-based and IB-based PRE schemes, and is efficient owing to its pairing-free property.

References

1. Sattam S. Al-Riyami and Kenneth G. Paterson. Certificateless public key cryptography. In *Advances in Cryptology - ASIACRYPT 2003, 9th International Conference on the Theory and Application of Cryptology and Information Security, Taiwan, November 30 - December 4, 2003, Proceedings*, pages 452–473, 2003.
2. Matt Blaze, Gerrit Bleumer, and Martin Strauss. Divertible protocols and atomic proxy cryptography. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 127–144. Springer, 1998.
3. Veronika Kuchta, Gaurav Sharma, Rajeev Anand Sahu, Tarunpreet Bhatia, and Olivier Markowitch. Secure certificateless proxy re-encryption without pairing. In *Advances in Information and Computer Security: 12th International Workshop on Security, IWSEC 2017, Hiroshima, Japan, August 30 – September 1, 2017, Proceedings*, pages 85–101. Springer International Publishing, 2017.
4. S.Sharmila Deva Selvi, Arinjita Paul, and C. Pandu Rangan. An efficient certificateless proxy re-encryption scheme without pairing. Cryptology ePrint Archive, Report 2017/768, 2017. `http://eprint.iacr.org/2017/768`.