Cryptanalysis of a pairing-free certificate-based proxy re-encryption scheme for secure data sharing in public clouds

S. Sharmila Deva Selvi, Irene Miriam Isaac, and C. Pandu Rangan

Theoretical Computer Science Lab,
Department of Computer Science and Engineering,
Indian Institute of Technology, Madras.
`sharmioshin@gmail.com,irenemisaac@gmail.com,rangan@cse.iitm.ac.in`

Abstract. Proxy re-encryption(PRE) is a primitive that is used to facilitate secure access delegation in the cloud. Proxy re-encryption allows a proxy server to transform ciphertexts encrypted under one user's public key to that under another user's public key without learning anything about the underlying message or the secret key. Over the years proxy re-encryption schemes have been proposed in different settings. In this paper we restrict our analysis to certificate based proxy re-encryption. The first CCA secure certificate based PRE without bilinear pairings was proposed by Lu and Li in Future Generation Computer Systems, 2016. In this paper we present a concrete attack on their scheme and prove that it is not CCA secure.

Keywords: Public cloud, Data sharing, Certificate-based proxy re-encryption, Bilinear pairing,Chosen-ciphertext security, Random oracle model

1 Introduction

Proxy re-encryption(PRE) is a mechanism in which a semi-trusted proxy can convert a ciphertext encrypted under a user Alice to a ciphertext encrypted under user Bob. Here Alice is the delegator while Bob is the delegatee. The main requirement of PRE is that the proxy should not obtain any information about the underlying message or the secret keys of the delegator. The delegator Alice constructs a re-encryption key using her secret key and the public parameters. The proxy uses this re-encryption key to transform a ciphertext under the public key of Alice to a ciphertext under the public key of Bob.

Proxy re-encryption finds applications in many fields, the most important ones being encrypted email forwarding, secure distributed file systems and outsourced filtering of encrypted spam. In the case of email forwarding, Alice can entrust a proxy to temporarily delegate her decryption rights to Bob in her absence.

PRE schemes can be divided into two based on the direction of re-encryption. Unidirectional schemes allows the ciphertext to be re-encrypted only in one direction i.e. either from Alice to Bob or from Bob to Alice. Bidirectional proxy re-encryption schemes on the other hand allows the ciphertext to be re-encrypted in both ways i.e. from Alice to Bob and from Bob to Alice. PRE schemes can also be classified with respect to their usability. Single-hop schemes imply that a given ciphertext can be re-encrypted only once while multi-hop schemes allow the ciphertext to be re-encrypted multiple times. The majority of proxy re-encryption schemes proposed are in the Identity Based, PKI (Public Key Infrastructure) based, attribute based or lattice based setting.

The first proxy re-encryption scheme was proposed by Blaze,Bleumer and Strauss [3] in 1998. However, the paper does not give any formal definition of PRE and the scheme is transitive and is not collusion resistant. The first unidirectional PRE scheme was designed by Dodis and Ivan[6] but it is not secure as the decryption key of delegatee Bob requires a part of the private key of the delegator Alice. Ateniese et. al [2] proposed the first unidirectional PRE using bilinear pairings. Though their scheme is non-transitive and collusion resistant, it provides chosen plaintext security only. Traditional PKI based systems require certificates from a trusted certificate authority (CA) to ensure the authenticity of the public keys. Thus PKI systems suffer from third party queries and certificate management problems. In 2007, Ateniese and Green [5] proposed the first Identity Based PRE in the random oracle model. Although ID based PRE schemes solve the issues with PKI systems, it suffers from key escrow problem. Since private keys have to be sent over secure channels, key distribution is also an issue. Al-Riyami and Patterson [1] introduced certificate-less PKC in 2003. Xu et al. [11] extended it to certificate-less PRE to solve the key escrow problem. Here the private key is generated by a user and a partially trusted Key Generation Center(KGC). It also suffers from key distribution problem as the partial private key has to be sent to the user securely.

The concept of certificate based PRE was introduced by Sur et al.[10] based on the certificate based encryption introduced by Gentry[4]. Li et al. [7] and Lu et al. [8] proposed CB-PRE schemes in the random oracle model. All the three schemes use the costly bilinear pairings. Lu et al.[9] proposed a CB-PRE scheme in the random oracle model without bilinear pairings. In CB-PRE the user generates a public key/private key pair similar to a PKI based system. The CA issues a certificate on a user's public key. The certificate is bound to the user's identity and acts as a partial decryption key. Decryption can be performed only if both the private key and the certificate are known. This eliminates the third party queries and solves the certificate revocation problem. There is no key escrow problem as the CA doesn't know the secret key. Since certificates are sent publicly there is no overhead associated with key distribution.

Paper Organisation In section 2 we present the definition and security model of CB-PRE. In section 3 we present a review of the scheme by Lu et al. [9]. In section 4 we propose an attack on the scheme.

2 Definition and Security model of CB-PRE

2.1 Definition

A CB-PRE scheme has the following algorithms:

Setup(κ): On input of security parameter κ, this algorithm will output the public parameters $params$ and a master secret key msk. This algorithm is performed by a CA. The CA then publishes the public parameters $params$ while the master secret key msk is kept secret.

UserKeyGen($params, ID_U$): Given the security parameters $params$, this algorithm outputs a private key SK_U and a partial public key PPK_U for a user with identity ID_U.

Certify$(params, msk, ID_U, PPK_U)$**:** This algorithm is performed by a CA. The algorithm produces a full public key PK_U a certificate $Cert_U$ for the user U. The algorithm takes as input the public parameters $params$, the master secret key msk, a user U's identity ID_U, and a partial public key PPK_U. PPK_U and $Cert_U$ are sent to the user via an open channel.

Encrypt$(params, M, ID_A, PK_A)$ The algorithm outputs the original ciphertext C_A on input of the public parameters $params$, a message M, a delegator A's identity ID_A and public key PK_A.

ReKeyGen$(params, ID_A, SK_A, Cert_A, ID_B, PK_B)$**:** On input of the public parameters $params$, a delegator A's identity ID_A, secret key SK_A, certificate $Cert_A$, a delegate B's identity ID_B and public key PK_B, the algorithm outputs the re-encryption key $RK_{A \to B}$.

ReEncrypt$(params, C_A, RK_{A \to B})$**:** The algorithm takes as input the public parameters $params$, the original ciphertext C_A and the re-encryption key $RK_{A \to B}$ and outputs the re-encrypted ciphertext C_B under a delegate B's identity ID_A and public key PK_B.

Decrypt1$(params, C_A, ID_A, SK_A, Cert_A)$**:** On input of the public parameters $params$, an original ciphertext C_A, a user A's identity ID_A, private key SK_A and certificate $Cert_A$ the algorithm outputs the message M if the decryption is successful else outputs $\perp$.

Decrypt2$(params, C_B, ID_B, SK_B, Cert_B, ID_A, PK_A)$**:** On input of the public parameters $params$, an re-encrypted ciphertext C_B, a delegate B's identity ID_B, private key SK_B, certificate $Cert_B$, the delegator A's identity ID_A and public key PK_A, the algorithm outputs the message M if the decryption is successful else outputs $\perp$.

A CB-PRE scheme as defined above is correct if for any message M the following two conditions hold:

1. Decrypt1$(params, Encrypt(params, M, ID_A, PK_A), ID_A, SK_A, Cert_A) = M$
2. Decrypt2$(params, ReEncrypt(params, Encrypt(params, M, ID_A, PK_A), RK_{A \to B}), ID_B, SK_B, Cert_B, ID_A, PK_A) = M$

2.2 Security Model

The security model of CB-PRE schemes is defined by two adversaries. Type-1 adversary is an uncertified user who doesn't know the master secret key msk and the target user's certificate $Cert_T$. Type-2 adversary on the other hand is an honest-but-curious CA who knows the master secret key msk and is responsible for generating the user's certificates. The indistinguishability against adaptive chosen ciphertext attacks (IND-CCA2) of CB-PRE schemes can be modeled as two different games between an adversary and a challenger. **Game-1** is played against a Type-1 adversary and **Game-2** against a Type-2 adversary. The security model makes use of six-oracles. The adversary can query these oracles adaptively. The oracles are defined as follows:

$O_{UserCreate}$: When the adversary queries the oracle with an identity ID_i, the challenger does the following:

1. If the identity ID_i does not exist, the challenger creates a public key private key pair PK_i, SK_i for ID_i and outputs PK_i. Here, the user with identity ID_i is created.
2. Else the challenger returns the PK_i associated with ID_i.

The other oracles only respond to queries on existing identities.

$O_{Corrupt}$: When the adversary queries the oracle with an identity ID_i the challenger returns the secret key SK_i of ID_i.

$O_{Certificate}$: When the adversary queries the oracle with an identity ID_i the challenger returns the certificate $Cert_i$ of ID_i.

$O_{ReKeyGen}$: When the adversary queries the oracle with two identities ID_i and ID_j the challenger returns the re-encryption key $RK_{A \to B}$ from ID_i to ID_j.

$O_{ReEncrypt}$: When the adversary queries the oracle with two identities ID_i, ID_j and the original ciphertext C_i the challenger returns the re-encrypted cipertext C_j.

$O_{Decrypt}$: When the adversary queries the oracle with an identity ID_i and the original or re-encrypted ciphertext C_i the challenger returns the decryption of C_i.

The **Game-1** between a Type-1 adversary (A_1) and a challenger proceeds as follows:

Setup: The challenger takes as input a security parameter κ and runs the algorithm $Setup(\kappa)$. The algorithm generates a master secret key msk and the public parameters $params$. The challenger returns the $params$ to the adversary and keeps msk to itself.

Phase-1: The adversary A_1 can adaptively query the oracles $O_{UserCreate}$, $O_{Corrupt}$, $O_{Certificate}$, $O_{ReKeyGen}$, $O_{ReEncrypt}$ and $O_{Decrypt}$.

Challenge: Once **Phase-1** is over A_1 outputs a challenge identity ID_T and two messages (M_0, M_1). The challenger chooses a bit $b \epsilon_R \{0, 1\}$ outputs the challenge ciphertext $C^* = Encrypt(params, M_b, ID_T, PK_T)$.

Phase-2: The adversary A_1 queries the oracles as in **Phase-1**. The adversary A_1 can not query the oracle $O_{Certificate}$ on the identity ID_T or the oracle $O_{Decrypt}$ on (ID_T, C^*) and its derivatives.

Guess: The adversary A_1 outputs a bit $b' \epsilon \{0, 1\}$. The adversary wins the game if $b = b'$. The advantage of A_1 can be defined as $Adv(A_1) = 2|Pr[b = b'] - 1/2|$.

The **Game-2** between a Type-2 adversary (A_2) and a challenger proceeds as follows:

Setup: The challenger takes as input a security parameter κ and runs the algorithm $Setup(\kappa)$. The algorithm generates a master secret key msk and the public parameters $params$. The challenger returns the $params$ and msk to the adversary.

Phase-1: The adversary A_2 can adaptively query the oracles $O_{UserCreate}$, $O_{Corrupt}$, $O_{ReKeyGen}$, $O_{ReEncrypt}$ and $O_{Decrypt}$.

Challenge: Once **Phase-1** is over A_2 outputs a challenge identity ID_T and two messages (M_0, M_1). The challenger chooses a bit $b \epsilon_R \{0, 1\}$ outputs the challenge ciphertext $C^* = Encrypt(params, M_b, ID_T, PK_T)$.

Phase-2: The adversary A_2 queries the oracles as in **Phase-1**. The adversary A_2 can not query the oracle $O_{Corrupt}$ on the identity ID_T or the oracle $O_{Decrypt}$ on (ID_T, C^*) and its derivatives.

Guess: The adversary A_2 outputs a bit $b' \epsilon \{0, 1\}$. The adversary wins the game if $b = b'$. The advantage of A_2 can be defined as $Adv(A_2) = 2|Pr[b = b'] - 1/2|$.

3 Review of Scheme

Setup(κ): The algorithm takes the security parameter κ as input, and generates the public parameters $params$ and the master secret key msk as follows:

1. Construct an additive cyclic group G of elliptic curve points with order q and generator P. q is a κ bit prime number.
2. Compute $P_{pub} = \alpha P$ where $\alpha \; \epsilon_R \; Z_q^*$.
3. Choose five cryptographic hash functions $H_1 : \{0,1\}^* \times G \times G \to Z_q^*, H_2 : \{0,1\}^n \times \{0,1\}^l \times \{0,1\}^* \times G \times G \to Z_q^*, H_3 : G \to \{0,1\}^{n+l}, H_4 : G \times \{0,1\}^{n+l} \times G \to Z_q^*$ and $H_5 : \{0,1\}^* \times \{0,1\}^* \times G \to Z_q^*$, where n and l denote the bit-length of a message and a random bit string respectively.
4. The public parameters $params = \{G, q, P, P_{pub}, n, l, H_1, H_2, H_3, H_4, H_5\}$ and master secret key $msk = \alpha$.

UserKeyGen($params$) : The algorithm takes the public parameters $params$ as input, and outputs an element $x_U \epsilon_R Z_q^*$ as a private key SK_U for a user U and a partial public key $PPK_U = x_U P$.

Certify($params, msk, ID_U, PPK_U$): The algorithm takes as input the public parameters $params$, the master secret key $msk = \alpha$, a user U's identity ID_U, a partial public key PPK_U and does the following:

1. Compute user U's full public key as $PK_U = (PK_{U1}, PK_{U2}) = (PPK_U, y_U P)$ where $y_U \epsilon_R Z_q^*$.
2. Certificate of user U is given as $Cert_U = y_U + \alpha H_1(ID_U, PK_U)$.

Encrypt($params, M, ID_A, PK_A$) The algorithm takes as input the public parameters $params$, a message $M \in \{0,1\}^n$, a delegator A's identity ID_A, public key $PK_A = (PK_{A1}, PK_{A2})$ and does the following:

1. Compute $r = H_2(M, \delta, ID_A, PK_A)$ where $\delta \epsilon_R \{0,1\}^l$.
2. Compute $Q_A = PK_{A1} + PK_{A2} + H_1(ID_A, PK_A) P_{pub}$ and set $X = rP$ and $Y = (M||\delta) \oplus H_3(rQ_A)$.
3. Set $Z = tP, \sigma = t + rH_4(X, Y, Z)$ where $t \epsilon_R Z_q^*$.
4. Send $C_A = (X, Y, Z, \sigma)$ as the original ciphertext of the message M.

ReKeyGen($params, ID_A, SK_A, Cert_A, ID_B, PK_B$): On input of the public parameters $params$, a delegator A's identity ID_A, secret key SK_A, certificate $Cert_A$, a delegate B's identity ID_B

and public key $PK_B = (PK_{B1}, PK_{B2})$, the algorithm sets $s = H_5(ID_A, ID_B, SK_A(PK_{B1} + PK_{B2} + H_1(ID_B, PK_B)P_{pub}))$ and outputs the re-encryption key $RK_{A\to B} = s^{-1}(SK_A + Cert_A)$. Note that $RK_{A\to B} = s^{-1}(SK_A + Cert_A) = H_5(ID_A, ID_B,$ $SK_A(PK_{B1} + PK_{B2} + H_1(ID_B, PK_B)P_{pub})^{-1}(SK_A + Cert_A)$.

ReEncrypt($params, C_A, RK_{A\to B}$**):** The algorithm takes as input the public parameters $params$, the original ciphertext $C_A = (X, Y, Z)$, the re-encryption key $RK_{A\to B}$ and does the following:

1. If $\sigma P = Z + H_4(X, Y, Z)X$ go to step 2, else output $\perp$.
2. Set $X' = RK_{A\to B}X$ and $Y' = Y$ where
   $X' = RK_{A\to B}X = s^{-1}(SK_A + Cert_A)rP$
   $= rs^{-1}(PK_{A1} + PK_{A2} + H_1(ID_A, PK_A)P_{pub})$.
3. Output $C_B = (ID_A, X', Y')$ is the re-encrypted ciphertext.

Decrypt1($params, C_A, ID_A, SK_A, Cert_A$**):** The algorithm takes as input the public parameters $params$, the original ciphertext $C_A = (X, Y, Z, \sigma)$, the delegator A's identity ID_A, secret key SK_A, certificate $Cert_A$ and does as follows:

1. If $\sigma P = Z + H_4(X, Y, Z)X$ go to step 2, else output $\perp$.
2. Compute $M'||\delta' = Y \oplus H_3(rQ_A)$.
3. Output M' if $X = r'P$ holds, where $r' = H_2(M', \delta', ID_A, PK_A)$. Else output $\perp$.

Decrypt2($params, C_B, ID_B, SK_B, Cert_B, ID_A, PK_A$**):** The algorithm takes as input the public parameters $params$, the re-encrypted ciphertext $C_B = (ID_A, X', Y')$, the delegate B's identity ID_B, secret key SK_B, certificate $Cert_B$ and does as follows:

1. Compute $s' = H_5(ID_A, ID_B, (SK_B + Cert_B)PK_{A1})$.
2. Retrieve $M'||\delta' = Y' \oplus H_3(s'X')$.
3. Output M' if $X' = s'^{-1}r'(PK_{A1} + PK_{A2} + H_1(ID_A, PK_A)P_{pub})$ where $r' = H_2(M', \delta', ID_A, PK_A)$. Else output $\perp$.

4 The Attack

Lu and Li [9] presented the first CB-PRE scheme without pairings. They proved the CCA security of the scheme in the random oracle model. We now present a CCA attack on the scheme with respect to the Type-1 adversary A_1. By the definition given in the security model, Adversary A_1 is an uncertified user without the knowledge of the master secret key msk or the target user's certificate $Cert_T$.

According to Lu et al. [9] security model the adversary A_1 is allowed to query the oracles $O_{UserCreate}$, $O_{Corrupt}$, $O_{Certificate}$, $O_{ReKeyGen}$, $O_{ReEncrypt}$ and $O_{Decrypt}$ adaptively with some restrictions. During the game C runs the $Setup$ algorithm on input κ and gives the public parameters $params$ to A_1, while keeping the master secret msk. During **Phase-1** of **Game-1**, the adversary adaptively queries the oracles $O_{UserCreate}$, $O_{Corrupt}$, $O_{Certificate}$, $O_{ReKeyGen}$, $O_{ReEncrypt}$ and $O_{Decrypt}$. After interacting with the challenger in **Phase-1**, the adversary chooses two messages (M_0, M_1) and the target identity ID_T. The challenger encrypts the message M_b , $b \epsilon_R \{0, 1\}$ using identity ID_T. C gives the challenge ciphertext C^* to A_1. The adversary A_1 on receiving C^*, adaptively queries $O_{UserCreate}$, $O_{Corrupt}$, $O_{ReKeyGen}$, $O_{Certificate}$, $O_{ReEncrypt}$ and $O_{Decrypt}$ with the following restrictions:

1. A_1 can not query the oracle $O_{Certificate}$ on ID_T.
2. A_1 can not query the oracle $O_{Decrypt}$ on (ID_T, C^*) and its derivatives.

The adversary A_1 can query the $O_{ReEncrypt}$ oracle on the challenge ciphertext C^* towards any identity ID_j whose secret key is not known. A_1 now queries the $O_{ReEncrypt}$ to re-encrypt the challenge ciphertext C^* under a new identity ID_j. This is query is allowed by the security model and the $O_{ReEncrypt}$ returns the re-encrypted ciphertext C_j under identity ID_j to A_1. A_1 now queries $O_{Corrupt}$ for the secret key of the challenge identity ID_T. As this is a valid query according to the security model, the $O_{Corrupt}$ returns SK_T to A_1. The re-encrypted ciphertext is constructed such that it can be decrypted with the knowledge of the secret key SK_T and without the knowledge of the Certificate $Cert_T$. The adversary now retrieves the encrypted message M_b and sends the bit b to the challenger C and wins **Game-1**. The attack can be demonstrated as follows:

1. Let ID_T be the challenge identity and $C^* = \langle X, Y, Z, \sigma \rangle$ be the challenge ciphertext given to A_1 by C during the challenge phase. C^* is the encryption of M_b, $b \epsilon_R \{0, 1\}$ where
   (a) $X = rP$, $r = H_2(M_b, \delta, ID_T, PK_T)$
   (b) $Y = (M_b || \delta) \oplus H_3(rQ_T)$,
   $Q_T = PK_{T1} + PK_{T2} + H_1(ID_T, PK_T)P_{pub}$
   (c) $Z = tP$ where $t \epsilon_R Z_q^*$

   (d) $\sigma = t + rH_4(X, Y, Z)$
2. On receiving C^* the adversary A_1 queries the re-encryption of C^* from ID_T to ID_j. Here, it should be noted that C does not know the secret key SK_j corresponding to ID_j, but it knows SK_T of ID_T. This is a legal query according to the security definition.
3. Let $D_j = \langle ID_j, X', Y' \rangle$ be the output of the re-encryption of $C*$ from ID_T to ID_j where
   (a) $X' = s^{-1}(SK_T + Cert_T)rP$
   given $s = H_5(ID_T, ID_j, SK_T(PK_{j1}+PK_{j2}+H_1(ID_j, PK_j)P_{pub}))$
   (b) $Y' = Y$
4. Adversary A_1 upon receiving D_j does the following computation:
   (a) Queries the $O_{Corrupt}$ oracle with ID_j and receives the secret key SK_j as output. This is allowed according to the security model.
   (b) Computes $s = H_5(ID_T, ID_j, SK_T(PK_{j1} + PK_{j2} + H_1(ID_j, PK_j)P_{pub}))$.
   (c) Computes $\Delta = s(X') = ss^{-1}((SK_T + Cert_T)rP) = rQ_T$.
   Correctness of Δ: $\Delta = sX' = ss^{-1}((SK_T + Cert_T)rP$
   $= r(PK_{T1} + PK_{T2} + H_1(ID_T, PK_T)P_{pub}) = rQ_T$
   (d) $(M_b||\delta) = Y \oplus H_3(rQ_T)$
5. Thus the adversary A_1 can get back the message M_b corresponding to C^* without knowing the full private key and adhering to the constraints given in **Game-1**.
6. Now A_1 returns b as guess to C by making use of the oracles provided by C and without the knowledge of the private key.
7. Hence the scheme by Lu et al. [9] is not CCA secure according to the model.

References

1. Al-Riyami, S.S., Paterson, K.G.: Certificateless public key cryptography. In: International conference on the theory and application of cryptology and information security. pp. 452–473. Springer (2003)
2. Ateniese, G., Fu, K., Green, M., Hohenberger, S.: Improved proxy re-encryption schemes with applications to secure distributed storage. ACM Transactions on Information and System Security (TISSEC) **9**(1), 1–30 (2006)
3. Blaze, M., Bleumer, G., Strauss, M.: Divertible protocols and atomic proxy cryptography. In: International Conference on the Theory and Applications of Cryptographic Techniques. pp. 127–144. Springer (1998)

4. Gentry, C.: Certificate-based encryption and the certificate revocation problem. In: International Conference on the Theory and Applications of Cryptographic Techniques. pp. 272–293. Springer (2003)
5. Green, M., Ateniese, G.: Identity-based proxy re-encryption. In: International Conference on Applied Cryptography and Network Security. pp. 288–306. Springer (2007)
6. Ivan, A.A., Dodis, Y.: Proxy cryptography revisited. In: NDSS (2003)
7. Li, J., Zhao, X., Zhang, Y.: Certificate-based conditional proxy re-encryption. In: International Conference on Network and System Security. pp. 299–310. Springer (2015)
8. Lu, Y.: Efficient certificate-based proxy re-encryption scheme for data sharing in public clouds. KSII Transactions on Internet and Information Systems (TIIS) **9**(7), 2703–2718 (2015)
9. Lu, Y., Li, J.: A pairing-free certificate-based proxy re-encryption scheme for secure data sharing in public clouds. Future Generation Computer Systems **62**, 140–147 (2016)
10. Sur, C., Park, Y., Shin, S.U., Rhee, K.H., Seo, C.: Certificate-based proxy re-encryption for public cloud storage. In: 2013 Seventh International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing. pp. 159–166. IEEE (2013)
11. Xu, L., Wu, X., Zhang, X.: Cl-pre: a certificateless proxy re-encryption scheme for secure data sharing with public cloud. In: Proceedings of the 7th ACM symposium on information, computer and communications security. pp. 87–88. ACM (2012)