

A Proxy Re-Encryption Scheme Based on Elliptic Curve Group

Hang Li¹, Shou-Lin Yin^{1,*}, Chu Zhao¹, and Lin Teng¹

¹Software College

Shenyang Normal University

No.253, HuangHe Bei Street, HuangGu District, Shenyang, P.C 110034 - China

1451541@qq.com;352720214@qq.com;910675024@qq.com;1532554069@qq.com;

*Corresponding author:352720214@qq.com

Received August, 2016; revised November, 2016

ABSTRACT. *Due to the low efficiency of original proxy re-encryption scheme, we propose a proxy re-encryption scheme based on elliptic curve group. Under the computational Diffie-Hellman problem, we demonstrate the new scheme satisfying indistinguishable against adaptively chosen-ciphertext attacks in random oracle model. Proxy re-encryption scheme based on elliptic curve group does not need to calculate bilinear pairings. Finally, we give security analysis and make comparison to other schemes to verify the performance of our scheme.*

Keywords: Proxy re-encryption, Elliptic curve group, Adaptively chosen-ciphertext attacks, Bilinear pairings

1. **Introduction.** Cloud computing[1-2] is a hot topic in recent years, its aim is to integrate multiple computational entities with low costs into a perfect system. This system is composed of management of virtual computing, distributed file system, resource scheduling management and safety management control. Building high scalability, large scale, high availability and low cost cloud computing platform has become the direction of the current informatization construction. As the most widely used the cloud computing, cloud storage brings convenience to the users, it also has created the problem of data ownership and management separation at the same time. When a user uses the cloud computing environment, it needs to store the data into cloud and rely on the cloud service provider to process data, which makes the data separate from the user's control[3-4]. Due to privacy sensitive information in user data, so it needs proper mechanism to prevent the data in the cloud from unauthorized access. Ciphertext access control[5] is an important method to realize user data confidentiality and access control, this method requires that data are encrypted and stored in the cloud. In order to solve the problem of encrypted data distributing or sharing, some ciphertext access control methods use key distribution way, namely the data owner will send encrypted data to authorized users. In this case, it uses different keys to encrypt data for different access control units, obviously which can cause data repeated encryption and waste a lot of computation and storage resources. Another solution requires that the data owner retrieves encrypted data from cloud. After decrypting the data, it uses the public key of authorized users to re-encrypt data and sends it to authorized users. Obviously this will bring serious computation and communication overhead to the data owner, so it reduces the overall system efficiency. In cloud

computing environment, therefore, how to realize the efficient sharing of the encrypted data is a problem.

Proxy re-encryption technology[6-7] can effectively solve the problem of encrypted data sharing in cloud computing environment. Data owner will send the encryption key of specify users to the cloud provider, cloud provider will transfer data ciphertext stored in the cloud as data ciphertext of specified users, then the specified user uses his own private key to decrypt the re-encryption ciphertext. In recent years, the proxy re-encryption method is subsequently proposed. Also there are many proxy re-encryption schemes based on elliptic curve group.

Zhou[8] proposed a new proxy re-encryption pattern, referred to as an identity-based proxy re-encryption version 2. It allowed an authorized proxy to convert a ciphertext of an identity-based broadcast encryption scheme into a ciphertext of an identity-based encryption scheme. Deepa[9] described a key generation mechanism using Elliptic Curves. The generated key could be used as a symmetric key. The key was constituted by the contribution from all the legitimate users so that the revocation mechanism could be simplified, but at the expense of communication overhead. Chavan[10] presented a new intermediary proxy re-encryption technique. He foresaw that quick and secure reencryption would turn out to be progressively famous as a technique for overseeing scrambled document frameworks and he displayed new re-encryption conspires that understood a more grounded thought of security, and for adding access control to a protected record system. NU Amin[11] designed and analyzed a proxy promised signcryption scheme based on elliptic curve cryptosystem. However, these proxy re-encryption schemes depend on bilinear pairings computation, so they have a low efficiency.

So we propose a proxy re-encryption scheme based on elliptic curve group without calculating bilinear pairings. The new system scenario includes eight polynomial time algorithms. We use two adversary games to explain its security model. Under the computational Diffie-Hellman problem, we demonstrate the new scheme satisfying indistinguishable against adaptively chosen-ciphertext attacks in random oracle model. We also make comparisons with other latest proxy re-encryption schemes. The followings are the structures of this paper. In section2, we give some preliminaries. Section3 detailed introduces the new scheme. We give the security analysis in section4. There is a conclusion in section5.

2. Preliminaries.

2.1. Ellipse curve. In this paper, our scheme is based on ellipse curve group[12-13], and its security is based on computational Diffie-Hellman problem.

Supposing that F_p is a p element finite field. There are two elements a and b in F_p satisfying discriminant $\Delta = 4a^3 + 27b^2 \neq 0$. So ellipse curve can be written as $E(F_p)$ and it denotes the set of all the points (x, y) and infinity point O meeting Weierstrass equation $y^2 = x^3 + ax + b$. Namely $E(F_p) = ((x, y) | x, y \in F_p \text{ and } y^2 = x^3 + ax + b) \cup O$. Obviously, all the points in ellipse curve $E(F_p)$ consist of commutative group.

3. New scheme and its secure model.

3.1. Definition of new scheme. Proxy re-encryption scheme based on elliptic curve group is composed of the eight algorithms.

1. Setup algorithm. Input security parameter k , this algorithm generates and outputs main key msk of original ciphertext CA and system parameters set sps . CA publishes sps and keeps msk .

2. User-key-generation algorithm. Input sps , this algorithm generates and outputs private key SK_U of user U and public key pk_U .
3. Certificate generation algorithm. Input sps , msk , identity ID_U of user U and pk_U , this algorithm generates and outputs certificate $Cert_U$ of user U and public key pk_U . Generally, CA executes this algorithm.
4. Encryption algorithm. Input sps , plaintext M , identity ID_A of data publisher A and public key pk_A , this algorithm generates and outputs original ciphertext C_A of message A . Data publisher executes this algorithm.
5. Re-encryption key generation algorithm. Input sps , identity ID_A of data publisher A , private key SK_A , $Cert_A$, identity ID_A of authorized user and public key pk_B , this algorithm generates and outputs re-encryption key $RK_{A \rightarrow B}$. Data publisher A executes this algorithm.
6. Re-encryption algorithm. Input sps , original ciphertext C_A , re-encryption key $RK_{A \rightarrow B}$, this algorithm generates and outputs re-encryption ciphertext C_B or invalid flag. Cloud proxy executes this algorithm.
7. Decryption algorithm of original ciphertext. Input sps , original ciphertext C_A , ID_A , SK_A and $Cert_A$, this algorithm generates and outputs re-encryption plaintext M or invalid flag. Data publisher executes this algorithm.
8. Decryption algorithm of re-encryption ciphertext. Input sps , re-encryption ciphertext C_B , ID_A , PK_A , ID_B , SK_B and $Cert_B$, this algorithm generates and outputs re-encryption plaintext M or invalid flag. Authorized user B executes this algorithm.

The above algorithms satisfy the following correctness constraints. If $C_A = Encryption(sps, M, ID_A, PK_A)$, then $M = Decryption1(sps, M, C_A, ID_A, SK_A, Cert_A)$. If $RK_{A \rightarrow B} = ReKeyGen(sps, ID_A, SK_A, Cert_A, ID_B, PK_B)$ and $C_B = ReEncryption(sps, C_A, RK_{A \rightarrow B})$, then $M = Decryption2(sps, C_B, ID_A, PK_A, ID_B, SK_B, Cert_B)$.

3.2. Security model. Security model of proxy re-encryption scheme based on elliptic curve group includes two different adversary A_1 and A_2 . Adversary A_1 simulates uncertified user, he does not know the main key of system, but he can inquiry any user's privacy key and certification. Adversary A_2 simulates user with main key, he can generate any user's certification and inquiry private key of users except target user. Secure against chosen ciphertext attack[14,15] based on elliptic curve group can be defined by two adversary games: IND-CCA1 and IND-CCA2.

1. IND-CCA1 game.
 - (a) Setting system parameters. Challenger simulates algorithm $Setup(k)$ to generate system main key msk and parameters set sps . Challenger keeps msk and sends sps to adversary A_1 .
 - (b) The first stage inquiry. In this stage, adversary A_1 does the following inquiry.
 - User generation inquiry. Challenger maintains a list L_{user} to record user's private key, public key and certification. Initial value of list L_{user} is zero. Adversary A_1 inputs identity ID_U . If list already has corresponding record, then challenger directly outputs public key PK_U of corresponding identity ID_U . Otherwise, challenger produces corresponding private key SK_U of ID_U , public key PK_U and certification $Cert_U$, then it stores the results into list L_{user} and outputs public key PK_U . For any identity ID_U , adversary must first inquiry ID_U , then he can does other oracle inquiries.
 - Private key inquiry. Adversary A_1 inputs identity ID_U . Challenger acquires corresponding private key SK_U of identity ID_U from list L_{user} and sends it to adversary A_1 .

- Certification inquiry. Adversary A_1 inputs identity ID_U . Challenger acquires corresponding certification $Cert_U$ of identity ID_U from list L_{user} and sends it to adversary A_1 .
 - Re-encryption key inquiry. Adversary A_1 inputs identity ID_A and ID_B . Challenger generates a re-encryption key $RK_{A \rightarrow B}$ and sends it to adversary A_1 .
 - Re-encryption inquiry. Adversary A_1 inputs identity ID_A , ID_B and original ciphertext C_A . Challenger generates a re-encryption ciphertext C_B and sends it to adversary A_1 .
 - Decryption inquiry. Adversary A_1 inputs identity ID_A and a ciphertext C_U . Challenger decrypts ciphertext C_U and sends the decryption result to adversary A_1 .
- (c) Challenge stage. Adversary A_1 outputs identity ID_{ch} and two same length plaintexts (M_0, M_1) to start challenge. The limit is that adversary A_1 never inquires corresponding certification of ID_{ch} in first stage. Challenger randomly selects $\gamma \in \{0, 1\}$ and runs encryption algorithm $Encryption(sps, M_\lambda, ID_{ch}, PK_{ch})$ to generate original ciphertext C_{ch} . At last, C_{ch} as challenge ciphertext is sent to adversary A_1 .
- (d) The second stage inquiry. It is similar to stage 1. The limit is that adversary A_1 cannot inquire certification of challenge identity ID_{ch} . For any identity, $ID_U \neq ID_{ch}$. Adversary A_1 cannot do re-encryption key inquiry for (ID_{ch}, ID_U) and decryption inquiry for (ID_{ch}, C_{ch}) and (ID_{der}, C_{der}) . Where C_{der} is the output of re-encryption inquiry $(ID_{ch}, ID_{der}, C_{ch})$.
- (e) Guess. Adversary A_1 outputs the guess γ' of γ . If $\gamma = \gamma'$, then adversary A_1 wins this game. The advantage of adversary A_1 winning game is:

$$Adv_{A_1}^{IND-CCA1} = 2|Pr[\gamma = \gamma'] - 1/2|. \quad (1)$$

2. IND-CCA2 game.

- (a) Setting system parameters. Challenger simulates algorithm $Setup(k)$ to generate system main key msk and parameters set sps . Challenger keeps msk and sends sps to adversary A_2 .
- (b) The first stage inquiry. In this stage, adversary A_2 does the following inquiry.
- User generation inquiry. Challenger maintains a list L_{user} to record user's private key, public key and certification. Initial value of list L_{user} is zero. Adversary A_2 inputs identity ID_U . If list already has corresponding record, then challenger directly outputs public key PK_U of corresponding identity ID_U . Otherwise, challenger produces corresponding private key SK_U of ID_U , public key PK_U and certification $Cert_U$, then it stores the results into list L_{user} and outputs public key PK_U .
 - Private key inquiry. Adversary A_2 inputs identity ID_U . Challenger acquires corresponding private key SK_U of identity ID_U from list L_{user} and sends it to adversary A_2 .
 - Re-encryption key inquiry. Adversary A_2 inputs identity ID_A and ID_B . Challenger generates a re-encryption key $RK_{A \rightarrow B}$ and sends it to adversary A_2 .
 - Re-encryption inquiry. Adversary A_2 inputs identity ID_A , ID_B and original ciphertext C_A . Challenger generates a re-encryption ciphertext C_B and sends it to adversary A_2 .

- Decryption inquiry. Adversary A_2 inputs identity ID_A and a ciphertext C_U . Challenger decrypts ciphertext C_U and sends the decryption result to adversary A_2 .
- (c) Challenge stage. Adversary A_2 outputs identity ID_{ch} and two same length plaintexts (M_0, M_1) to start challenge. The limit is that adversary A_2 never inquiries corresponding certification of ID_{ch} in first stage. Challenger randomly selects $\gamma \in 0, 1$ and runs encryption algorithm $Encryption(sps, M_\lambda, ID_{ch}, PK_{ch})$ to generate original ciphertext C_{ch} of M_γ . At last, C_{ch} as challenge ciphertext is sent to adversary A_2 .
- (d) The second stage inquiry. It is similar to stage1. The limit is that adversary A_2 cannot inquiry private key of challenge identity ID_{ch} . For any identity, $ID_U \neq ID_{ch}$. Adversary A_2 cannot do re-encryption key inquiry for (ID_{ch}, ID_U) and decryption inquiry for (ID_{ch}, C_{ch}) and (ID_{der}, C_{der}) . Where C_{der} is the output of re-encryption inquiry $(ID_{ch}, ID_{der}, C_{ch})$.
- (e) Guess. Adversary A_2 outputs the guess γ' of γ . If $\gamma = \gamma'$, then adversary A_2 wins this game. The advantage of adversary A_2 winning game is:

$$Adv_{A_2}^{IND-CCA2} = 2|Pr[\gamma = \gamma'] - 1/2|. \quad (2)$$

Definition. Chosen-ciphertext security. If there is no any polynomial time adversary with a non-negligible advantage winning IND-CCA1 game and IND-CCA2 game. Then this new scheme meets chosen-ciphertext security.

4. Detailed description of proxy re-encryption scheme based on elliptic curve group.

1. Setup algorithm. Input secure parameter $k \in Z^+$, this algorithm generates q order elliptic curve group G , generator is g . Randomly select $\alpha \in Z_q^*$. Select seven hashing function $H_1 : 0, 1^* \times G^2 \rightarrow Z_q^*$, $H_2 : 0, 1^* \times G^3 \rightarrow Z_q^*$, $H_3 : 0, 1^* \times G^2 \times 0, 1^n \times 0, 1^l \rightarrow Z_q^*$, $H_4 : G \rightarrow 0, 1^l$, $H_5 : 0, 1^l \rightarrow 0, 1^n$, $H_6 : G \times 0, 1^l \times 0, 1^n \times G \rightarrow Z_q^*$, $H_7 : 0, 1^* \times 0, 1^* \times G \rightarrow Z_q^*$. Where $n \in Z^+$ and $l \in Z^+$ denote random bit string length in encryption and decryption algorithm respectively. Output system main key $msk = \alpha$ and $sps = (q, G, g, g_1, n, l, H_1, H_2, H_3, H_4, H_5, H_6, H_7)$.
2. User key generation algorithm(UserKeyGen). Input sps , this algorithm randomly selects $v_U \in Z_q^*$ as user's private key, namely $SK_U = v_U$. Compute public key $pk_U = g^{v_U}$ and output private key SK_U and public key pk_U .
3. Certification generation algorithm. Input (sps, msk, ID_U, pk_U) , this algorithm randomly selects $\mu_U \in Z_q^*$. Compute $pk_U^1 = g^{v_U}$ and set $PK_U^2 = pk_U$, $PK_U = (pk_U^1, PK_U^2)$. Calculate $Cert_U = \mu_U + \alpha \cdot h_U$. Where $h_U = H_1(ID_U, PK_U)$. Output public key PK_U and certification $Cert_U$.
4. Encryption algorithm. Input (sps, M, ID_A, PK_A) . This algorithm computes $g_A = (g_1)^{h_A} PK_A^1 (PK_A^2)^{h'_A}$. Where $h_A = H_1(ID_A, PK_A)$, $h'_A = H_2(ID_A, PK_A, g_1)$. Randomly select $\lambda \in 0, 1^l$ and compute $r = H_3(ID_A, PK_A, M, \lambda)$, $C_1 = g^r$, $C_2 = \lambda \oplus H_4((g_A)^r)$, $C_3 = M \oplus H_5(\lambda)$. Randomly select $s \in Z_q^*$, calculate $C_4 = g^s$, $C_5 = s + r \cdot H_6(C_1, C_2, C_3, C_4)$. Output original ciphertext $C_A = (C_1, C_2, C_3, C_4, C_5)$.
5. Re-encryption key generation algorithm(ReKeyGen). Input $(sps, ID_A, SK_A, Cert_A, ID_B, PK_B)$. This algorithm computes $g_B = (g_1)^{h_B} PK_B^1 (PK_B^2)^{h'_B}$, where $h_B = H_1(ID_B, PK_B)$, $h'_B = H_2(ID_B, PK_B, g_1)$. Compute $u = H_7(ID_A, ID_B, (g_B)^{SK_A})$ and output re-encryption key $RK_{A \rightarrow B} = \frac{1}{u}(SK_A h'_A + Cert_A)$, where $h'_A = H_2(ID_A, PK_A, g_1)$.

Therefore,

$$RK_{A \rightarrow B} = \frac{1}{u}(SK_A h'_A + Cert_A) \quad (3)$$

$$= H_6(ID_A, ID_B, (PK_A^2)^{SK_B h'_B + Cert_A})^{-1}(SK_A h'_A + Cert_A). \quad (4)$$

6. Re-encryption algorithm. Input $(sps, C_A = (C_1, C_2, C_3, C_4, C_5), RK_{A \rightarrow B})$. This algorithm firstly judges that whether $g^{C_5} = C_4(C_1)^\sigma$ is true. $\sigma = H_6(C_1, C_2, C_3, C_4)$. If g^{C_5} is false, then ciphertext C_A is invalid. Otherwise, it computes $C'_1 = (C_1)^{RK_{A \rightarrow B}}$. Let $C'_2 = C_2$ and $C'_3 = C_3$. Output re-encryption ciphertext $C_B = (ID_A, C'_1, C'_2, C'_3)$.

Therefore,

$$C'_1 = (C_1)^{RK_{A \rightarrow B}} = ((g_1)^{h_A}(PK_A^1)(PK_A^2)^{h'_A})^{ru-1}. \quad (5)$$

7. Original ciphertext decryption algorithm. Input $(sps, C_A = (C_1, C_2, C_3, C_4, C_5), ID_A, SK_A, Cert_A)$. This algorithm firstly judges that whether $g^{C_5} = C_4(C_1)^\sigma$ is true. $\sigma = H_6(C_1, C_2, C_3, C_4)$. If g^{C_5} is false, then ciphertext C_A is invalid. Otherwise, it computes $\lambda = C_2 \oplus H_4((C_1)^{SK_A h'_A + Cert_A})$ and $M = C_3 \oplus H_5(\lambda)$. Where $h'_A = H_2(ID_A, PK_A), g_1$. Verify $C_1 = g^r$. If it is true, then output plaintext M . Otherwise, ciphertext C_A is invalid.
8. Re-encryption ciphertext decryption algorithm. Input $(sps, C_B = (ID_A, C'_1, C'_2, C'_3), ID_B, SK_B, Cert_B, ID_A, PK_A)$.

This algorithm computes $u = H_7(ID_A, ID_B, (PK_A^2)^{SK_B h'_B + Cert_B})$, $\lambda = C'_2 \oplus H_4((C'_1)^u)$ and $M = C'_3 \oplus H_5(\lambda)$, where $h'_B = H_2(ID_B, PK_B, g_1)$.

Verify $C'_1 = ((g_1)^{h_A} PK_A^1 (PK_A^2)^{h'_A})^{ru-1}$.

Where $h_A = H_1(ID_A, PK_A)$, $h'_A = H_2(ID_A, PK_A, g_1)$, $r = H_3(ID_A, PK_A, M, \lambda)$. If it is true, then output plaintext M . Otherwise, ciphertext C_B is invalid.

Because

1.

$$g^{C_5} = g^{s+r+\sigma} = g^s(g^r)^\sigma = C_4(C_1)^\sigma. \quad (6)$$

2.

$$C_2 \oplus H_4((C_1)^{SK_A h'_A + Cert_A}) = C_2 \oplus H_4((g^r)^{SK_A h'_A + Cert_A}) = \quad (7)$$

$$C_2 \oplus H_4((g_A^r)) = \lambda \oplus H_4((g_A^r)) \oplus H_4((g_A^r)) = \lambda. \quad (8)$$

3.

$$C'_2 \oplus H_4((C'_1)^u) = C'_2 \oplus H_4(((C_1)^{RK_{A \rightarrow B}})^u) \quad (9)$$

$$\lambda \oplus H_4((g_A)^r) \oplus H_4((g_1)^{h_A} PK_A^1 (PK_A^2)^{h'_A})^{ru-1} = \quad (10)$$

$$\lambda \oplus H_4((g_A)^r) \oplus H_4((g_A)^r) = \lambda. \quad (11)$$

So our new scheme is right.

5. Security analysis and performance evaluation.

5.1. Security analysis.

Theorem 5.1. *In random oracle model, if there is an adversary A_1 satisfying chosen-ciphertext attacks based on proxy re-encryption, its advantage is ε . It does q_U user inquiries, q_R re-encryption inquiries, q_D decryption inquiries, q_3 oracle H_3 inquiries, q_4 oracle H_4 inquiries, then there exists an algorithm B with a non-negligible advantage can solve the CDH problem in G_1 .*

Proof. Given an algorithm B , a random instance (G, q, g, g^a, g^b) , algorithm B simulates IND-CCA1 game to calculate g^{ab} .

1. System parameter set. B randomly selects an index value $\theta \in [1, q_U]$. Let $g_1 = g^\alpha$. B sends $(G, q, g, g_1, n, l, H_1, H_2, H_3, H_4, H_5, H_6, H_7)$ to adversary A_1 , where $(H_1 \dots H_7)$ is random oracle.
2. First stage inquiry. Adversary A_1 adaptively does inquiries. Algorithm B does the following responses.
 - H_1 inquiry. Algorithm B maintains a list $L_1 = (ID_l, PK_l, h_1)$. Original value of L_1 is zero. Adversary A_1 inputs (ID_l, PK_l) . If (ID_l, PK_l, h_1) has been in L_1 , algorithm B directly outputs h_1 for adversary A_1 . Otherwise, algorithm B randomly selects $h_1 \in Z_q^*$. It puts (ID_l, PK_l, h_1) into list L_1 and outputs h_1 to adversary A_1 .
 - H_2 inquiry. Algorithm B maintains a list $L_2 = (ID_l, PK_l, g_1, h_2)$. Original value of L_2 is zero. Adversary A_1 inputs (ID_l, PK_l, g_1) . If (ID_l, PK_l, g_1, h_2) has been in L_2 , algorithm B directly outputs h_2 for adversary A_1 . Otherwise, algorithm B randomly selects $h_2 \in Z_q^*$. It puts (ID_l, PK_l, g_1, h_2) into list L_2 and outputs h_2 to adversary A_1 .
 - H_3 inquiry. Algorithm B maintains a list $L_3 = (ID_l, PK_l, \lambda, h_3)$. Original value of L_3 is zero. Adversary A_1 inputs (ID_l, PK_l, M, λ) . If $(ID_l, PK_l, M, \lambda, h_3)$ has been in L_3 , algorithm B directly outputs h_3 for adversary A_1 . Otherwise, algorithm B randomly selects $h_3 \in Z_q^*$. It puts $(ID_l, PK_l, g_1, \lambda, h_3)$ into list L_3 and outputs h_3 to adversary A_1 .
 - H_4 inquiry. Algorithm B maintains a list $L_4 = (S, h_4)$. Original value of L_4 is zero. Adversary A_1 inputs S . If (S, h_4) has been in L_4 , algorithm B directly outputs h_4 for adversary A_1 . Otherwise, algorithm B randomly selects $h_4 \in 0, 1^l$. It puts (S, h_4) into list L_4 and outputs h_4 to adversary A_1 .
 - H_5 inquiry. Algorithm B maintains a list $L_5 = (\lambda, h_5)$. Original value of L_5 is zero. Adversary A_1 inputs λ . If (λ, h_5) has been in L_5 , algorithm B directly outputs h_5 for adversary A_1 . Otherwise, algorithm B randomly selects $h_5 \in 0, 1^n$. It puts (λ, h_5) into list L_5 and outputs h_5 to adversary A_1 .
 - H_6 inquiry. Algorithm B maintains a list $L_6 = (C_1, C_2, C_3, C_4, h_6)$. Original value of L_6 is zero. Adversary A_1 inputs (C_1, C_2, C_3, C_4) . If $(C_1, C_2, C_3, C_4, h_6)$ has been in L_6 , algorithm B directly outputs h_6 for adversary A_1 . Otherwise, algorithm B randomly selects $h_6 \in Z_q^*$. It puts $(C_1, C_2, C_3, C_4, h_6)$ into list L_6 and outputs h_6 to adversary A_1 .
 - H_7 inquiry. Algorithm B maintains a list $L_7 = (ID_1, ID_2, T, h_7)$. Original value of L_6 is zero. Adversary A_1 inputs (ID_1, ID_2, T) . If $L_7 = (ID_1, ID_2, T, h_7)$ has been in L_7 , algorithm B directly outputs h_7 for adversary A_1 . Otherwise, algorithm B randomly selects $h_7 \in Z_q^*$. It puts $L_7 = (ID_1, ID_2, T, h_7)$ into list L_7 and outputs h_7 to adversary A_1 .
 - User generation inquiry. Algorithm B maintains a list $L_{user} = (ID_1, PK_2, SK_1, Cert_1, \mu_1)$. Original value of L_{user} is zero. Adversary A_1 inputs (ID_1) . Algorithm B executes the followings:
if $(ID_1, PK_2, SK_1, Cert_1, \mu_1)$ has been in L_{user} , algorithm B directly outputs PK_1 for adversary A_1 . Otherwise, if $1 = \theta$, algorithm B randomly selects $v_\theta, \mu_\theta \in Z_q^*$, computes $PK_\theta = (g^{v_\theta}, g^{\mu_\theta})$. Let $SK_\theta = v_\theta$. It puts $(ID_\theta, PK_\theta, SK_\theta, \perp, \mu_\theta)$ into list L_{user} and outputs PK_θ to adversary A_1 . Otherwise, algorithm B randomly selects $(v_1, t_1, h_1 \in Z_q^*)$. Compute $PK_1 = (g^{v_1}, g^{t_1}, g^{-h_1})$. Let $SK_1 = v_1$ and $Cert_1 = t_1$. It puts (ID_1, PK_1, h_1) and $(ID_1, PK_1, SK_1, Cert_1, \perp)$ into L_1 and L_{user} respectively. Output PK_1 to adversary A_1 .

- Private key inquiry. Adversary A_1 inputs identity ID_1 . Algorithm B gets record $(ID_1, PK_1, SK_1, Cert_1, \mu_1)$ from list L_{user} and outputs SK_1 to adversary A_1 .
 - Certification inquiry. Adversary A_1 inputs identity ID_1 . If $ID_1 = ID_\theta$, then algorithm B exists this game. Otherwise, algorithm B gets record $(ID_1, PK_1, SK_1, Cert_1, \mu_1)$ from list L_{user} and outputs $Cert_1$ to adversary A_1 .
 - Re-encryption key inquiry. Adversary A_1 inputs identity (ID_1, ID_2) . If $ID_1 = ID_\theta$, then algorithm B exists this game. Otherwise, algorithm B gets record $(ID_1, PK_1, SK_1, Cert_1, \mu_1)$ and $(ID_2, PK_2, SK_2, Cert_2, \mu_2)$ from list L_{user} and executes algorithm $ReKeyGen(sps, ID_1, PK_1, SK_1, Cert_1, ID_2, PK_2)$ to generate re-encryption key $RK_{1 \rightarrow 2}$ for adversary A_1 .
 - Re-encryption inquiry. Adversary A_1 inputs identity $(ID_1, ID_2, C'_1 = (C_1, C_2, C_3, C_4, C_5,))$. Algorithm B firstly verifies that whether $g^{C_5} = C_4(C_1)^\sigma$. Where $\sigma = H_6(C_1, C_2, C_3, C_4)$. If this equation is true, then ciphertext C'_1 is invalid. Algorithm B refuses this inquiry. If it is true, then B does the following operation.
 If $ID_1 = ID_\theta$, algorithm B searches the record $(ID_1, PK_1, M, \lambda, h_3)$ from list L_3 , meanwhile, $C_1 = g^{h_3}$, $C_2 = \lambda \oplus H_4(((g_1)^{h_1} PK_1^1 (PK_1^2)^{h'_1})^{h_3})$, $C_3 = M \oplus H_5(\lambda)$, $h_1 = H_1(ID_1, PK_1)$ and $h'_1 = H_2(ID_1, PK_1, g_1)$. If this record is nonexistence, algorithm B refuses this inquiry. Otherwise, B firstly inquiries $H_7(ID_1, ID_2, ((g_1)^{h_2} PK_2^1 (PK_2^2)^{h'_2})^{SK_1})$ to get h_7 , where $h_2 = H_1(ID_2, PK_2)$, $h'_2 = H_2(ID_2, PK_2, g_1)$. Then it computes $C'_1 = ((g_1)^{h_1} PK_1^1 (PK_2^2)^{h'_1})^{h_3 h_{7+1}}$ and outputs re-encryption ciphertext $C_2 = (ID_1, C'_1, C_2, C_3)$.
 Otherwise, B firstly does re-encryption key inquiry for (ID_1, ID_2) and gets re-encryption key $RK_{i \rightarrow 2}$, then it executes re-encryption algorithm $ReEncryption(sps, C_1, RK_{i \rightarrow 2})$ to generate re-encryption ciphertext C_2 . B sends it to adversary A_1 .
 - Decryption inquiry. Adversary A_1 inputs identity (ID_1, C_1) . Algorithm B does the following operation.
 If $ID_1 = ID_\theta$, and $C'_1 = (C_1, C_2, C_3, C_4, C_5,)$ is an original ciphertext. Algorithm B firstly verifies that whether $g^{C_5} = C_4(C_1)^\sigma$. Where $\sigma = H_6(C_1, C_2, C_3, C_4)$. If this equation is true, then ciphertext C'_1 is invalid. Algorithm B refuses this inquiry. Otherwise, algorithm B searches the record $(ID_1, PK_1, M, \lambda, h_3)$ from list L_3 , meanwhile, $C_1 = g^{h_3}$, $C_2 = \lambda \oplus H_4(((g_1)^{h_1} PK_1^1 (PK_1^2)^{h'_1})^{h_3})$, $C_3 = M \oplus H_5(\lambda)$, $h_1 = H_1(ID_1, PK_1)$ and $h'_1 = H_2(ID_1, PK_1, g_1)$. If this record is existence, algorithm B refuses this inquiry. Otherwise, B outputs M (as the decryption of ciphertext C_1) to adversary A_1 .
 If $ID_1 \neq ID_\theta$, B knows private key and certification of identity ID_1 , it executes decryption algorithm to decrypt ciphertext C_1 .
3. Challenge stage. Adversary A_1 outputs identity ID_{ch} and two same length plaintexts (M_0, M_1) to start challenge. If $ID_{ch} \neq ID_\theta$, B stops the game. Otherwise, B randomly selects $\gamma \in 0, 1$, $C_2^* \in 0, 1^l$, $C_3^* \in 0, 1^n$ and $C_5^*, h_6^* \in Z_q^*$. Let $C_1^* = g^b$. Compute $C_4^* = g^{C_5^*} (C_1^*)^{-h_6^*}$. Then it adds $(C_1^*, C_2^*, C_3^*, C_4^*, h_5^*)$ into list L_6 and outputs $C_{ch} = (C_1^*, C_2^*, C_3^*, C_4^*, C_5^*)$ (as ciphertext of M_γ) to adversary A_1 .
 Obviously, decryption of ciphertext C_{ch} is $C_3^* \oplus C_2^* \oplus H_4((C_1^*)^{SK_\theta h'_\theta + Cert_\theta} = C_3^* \oplus C_2^* \oplus H_4((g^b)^{\mu_\theta + v_\theta h'_\theta + a h_\theta})$. Where $b = H_3(ID_\theta, PK_\theta, M_\gamma, \lambda^*)$, $\lambda^* \in 0, 1^l$, $h_\theta = H_1(ID_\theta, PK_\theta)$, $h'_\theta = H_1(ID_\theta, PK_\theta, g_1)$
4. Second stage inquiry. It is similar to stage1. The limit is that adversary A_1 cannot inquiry certification of challenge identity ID_{ch} . For any identity, $ID_U \neq ID_{ch}$. Adversary A_1 cannot do re-encryption key inquiry for (ID_{ch}, ID_U) and decryption

inquiry for (ID_{ch}, C_{ch}) and (ID_{der}, C_{der}) . Where C_{der} is the output of re-encryption inquiry $(ID_{ch}, ID_{der}, C_{ch})$.

5. Guess stage. Adversary A_1 outputs the guess γ' of γ . B ignores the guess of A_1 and randomly selects a record (S, h_4) , computes $T = (S/(g^b)^{\mu_\theta + \nu_\theta h'_\theta})^{1/h_\theta}$. If $S = (g^b)^{\mu_\theta + \nu_\theta h'_\theta + ah_\theta}$, then $T = g^{ab}$. Therefore, algorithm B with a non-negligible advantage can solve the CDH problem in G_1 .

5.2. Performance evaluation. We make a comparison to PRCE scheme[11] and CBCPR scheme[16] with our new scheme. Supposing that bilinear pairings in this scheme is $e : G \times G \rightarrow G_T$. G_T is bilinear target group. Table1 is the computation complexity with different schemes. Where symbols p , e_T , e and h denote bilinear pairings operation, exponential operation in G_T , exponential operation in G and Hash operation. Their coefficients are operation numbers. From the table, we can know that our new scheme needs the least operation time. In addition, it has the optimal encryption results.

TABLE 1. Performance comparison with different schemes

Stage	PRCE scheme	CBCPR scheme	New scheme
<i>Encryption</i>	$2p + e_T + 3e + 2h$	$3p + e_T + 3e + 2h$	$4e$
<i>ReKeyGen</i>	$2p + e_T + 3e + 3h$	$2p + e_T + 4e + 3h$	$2e$
<i>ReEncryption</i>	$6p + 2h$	$4p + h$	$3e$
<i>Deryption1</i>	$3p + 2e_T + 3e + 2h$	$4p + 3e + 2h$	$3e$
<i>Deryption2</i>	$5p + e_T + 2e + 2h$	$4p + e_T + 2e + h$	$4e$
<i>ReEnryp. key</i>	$3 G $	$ G_T + 3 G $	$ Z_q^* $
<i>Original ciphertext.</i>	$3 G + n + 2l$	$2 G_T + 2 G + n + l$	$2 G_T + Z_q^* + n + l$
<i>ReEnryp. ciphertext.</i>	$2 G_T + 3 G + n + l$	$ G_T + 3 G + n + 2l$	$ G + 2n + l$

6. Conclusions. In this paper, we propose a proxy re-encryption scheme based on elliptic curve group. This new scheme can meet indistinguishable against adaptively chosen-ciphertext attacks in random oracle model. We also give security proof and efficiency analysis in this paper. And comparison with other proxy re-encryption schemes shows that our scheme has high efficiency. In the future, we will study more advanced re-encryption schemes taking communication cost between authorized user and proxy into consideration.

Acknowledgment. The authors also gratefully acknowledge the helpful comments and suggestions of the reviewers, which have improved the presentation.

REFERENCES

- [1] B. R. Chang, H. F. Tsai, C. M. Chen, Evaluation of virtual machine performance and virtualized consolidation ratio in cloud computing system,, *International Journal of Journal of Information Hiding and Multimedia Signal Processing*, vol. 4, no. 3, pp. 192-200, 2013.
- [2] B. R. Chang, C. F. Huang, H. F. Tsai, et al., Rapid access control on ubuntu cloud computing with facial recognition and fingerprint identification, , *International Journal of Journal of Information Hiding and Multimedia Signal Processing*, vol. 3, no.2, pp. 176-190.
- [3] Q. C. Zhang, L. T. Yang, Z. K. Chen, Privacy Preserving Deep Computation Model on Cloud for Big Data Feature Learning, *IEEE Transactions on Computers*, vol. 65, no.5, pp. 1351-1362, 2016.
- [4] Q. C. Zhang, L. T. Yang, Z. K. Chen, Deep Computation Model for Unsupervised Feature Learning on Big Data, *IEEE Transactions on Services Computing*, vol. 9, no. 1, pp. 161-171, 2016.

- [5] Wang G, Liu Q, Wu J. Hierarchical attribute-based encryption for fine-grained access control in cloud storage services, *[C]//Proceedings of the 17th ACM conference on Computer and communications security. ACM, 2010*, pp. 735-737.
- [6] K. Liang , M. H. Au ,J. K. Liu , et al., A secure and efficient ciphertext-policy attribute-based proxy re-encryption for cloud data sharing, *International Journal of Future Generation Computer Systems*, vol. 52, pp. 95-108, 2015.
- [7] K. Liang, L. Fang, D. S. Wong, et al., A ciphertext-policy attribute-based proxy re-encryption scheme for data sharing in public clouds, *Journal of Concurrency and Computation: Practice and Experience*, , vol. 27, no. 8, pp. 2004-2027, 2015.
- [8] Y. Zhou , H. Deng, Q. Wu , et al., Identity-based proxy re-encryption version 2: making mobile access easy in cloud, *Journal of Future Generation Computer Systems*, vol. 62, pp. 128-139, 2016.
- [9] S. Deepa , M. Abdul, Design of ID-based Contributory Key Management Scheme using Elliptic Curve Points for Broadcast Encryption, *International Journal of Computer Applications*, vol. 129, no. 11, pp. 16-23, 2015.
- [10] S. B. Chavan , A. Kumar, Efficient Fine-grained access control for Shared Data using Proxy Re-encryption, *International Journal of Current Engineering and Technology*, vol.6, no.4, pp. 1323-1325 2016.
- [11] I. UllahN. U. AminA. I. Umar. Proxy Promised Signcryption Scheme Based on Elliptic Curve Crypto System, *International Journal of International Journal of Computer.*, vol. 20, no 1.2016. , pp. 167-173.
- [12] V. Thangam , K. Chandrasekaran, Elliptic Curve Based Proxy Re-Encryption, *Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies*, paper 121, pp. 1-6, 2016.
- [13] W. Sun , S. Yu, W. Lou, et al., Protecting Your Right: Verifiable Attribute-based Keyword Search with Fine-grainedOwner-enforced Search Authorization in the Cloud, , *International Journal of IEEE Transactions on Parallel & Distributed Systems*, vol. 1, pp.226-234, 2014.
- [14] Q. Tang , H. Ma, X. Chen, Extend the Concept of Public Key Encryption with Delegated Search, , *International Journal of Computer Journal*, vol. 58, no. 4, pp.724-734, 2013.
- [15] A. Arriaga, Q. Tang, P. Ryan, Trapdoor Privacy in Asymmetric Searchable Encryption Schemes, *[M]// Progress in Cryptology C AFRICACRYPT 2014. Springer International Publishing*, pp.31-50, 2014.
- [16] J. Li , Z. Zhao, Y. Zhang, Certificate-based conditional proxy re-encryption, *International Conference on Network and System Security, Springer International Publishing*, pp. 299-310, 2014.