Proxy Key Re-encapsulation Mechanism for Group Communications

Chunbo Ma and Jun Ao
School of Information and Communication,
Guilin University of Electronic and Technology, Guilin, Guangxi, 541004, P. R. China
machunbo@guet.edu.cn

Abstract. Many practical applications use hybrid encryption mechanism to deal with large plaintext messages or real-time communication since the performance of the public key encryption scheme is poor. The key encapsulation is a crucial part in hybrid encryption mechanism, which allows a sender to generate a random session key and distribute it to recipient. In this paper we present a proxy key re-encapsulation scheme for group communication. The proxy in our scheme is allowed to transform the encapsulated message corresponding to group A's public key into one that can be decapsulated by the member in group B. It can be used in cases when a group users need to perform sensitive operation without holding the necessary secret key.
Keywords. Proxy, Re-encapsulation, Group communication, Public key crypto

1. Introduction

In some network applications, we have to distribute same message to all n group members. A simple approach for achieving this goal is that the sender encrypts the message respectively for each member of the group. Obviously, the cost of using the simple approach in large groups is very high. Therefore, how to efficiently distribute a message in this scenario and ensure the network security has attracted lots of attention. To due with this problem, some methods used for group communication were proposed, such as [1][2][3][4][5].

To exert the virtue of symmetrical and unsymmetrical crypto, the hybrid encryption mechanism is used in group communication. Cramer and Shoup [6] first presented the notion of hybrid encryption schemes in 1998, and followed by [7][8][9]. Generally speaking, this kind of scheme consists of two parts, one is key encapsulation mechanism (KEM), and another is data encapsulation mechanism (DEM). The KEM is similar to the ordinary encryption component. What they are different is that the target of the KEM is to transmit the "session key" not encrypted message. And the "session key" is random selected by the sender, but the encrypted message maybe comes from an attacker.

The re-encryption [10][11] can be used in some scenarios. For example, proxy in firewall is allowed to transform a ciphertext corresponding to Server-1's public key into one that can be decrypted by Server-2's private key. In order to ensure the security of data, we require that the proxy who preserves the secret transform key can't obtain the plaintext via its transform key. There are some other applications, such as secure email forward, secure storage and so on [12].

The re-capsulation technology is similar to re-encryption. Generally speaking, re-capsulation is such a mechanism that the proxy diverts the ciphertext from Alice to Bob. However, the proxy can't obtain any information on encapsulated message via its secret transform key. In practice, this kind of encapsulation scheme is divided into two categories by proxy functions, namely bidirectional and unidirectional. In a bidirectional scheme the proxy secret key can be used to divert ciphertext both from Alice to Bob and from Bob to Alice. In a unidirectional scheme, the proxy secret key is only allowed to divert ciphertext either from Alice to Bob or from Bob to Alice.

In this paper we present a bidirectional proxy key re-encapsulation mechanism for group communication. In this scheme, Proxy diverts the ciphertext from group A to group B, such that every member in group B can decapsulate the ciphertext independently. The Proxy who holds the transform key can't obtain any information about the encapsulated key, and this characteristic ensures the security of the transmission.

The rest of paper consists of following sections. In section 2, we introduce some related works. In section 3, we give the security model and complexity assumptions. The proposed group-based proxy re-encapsulation scheme is presented in section 4. In section 5, we discuss the security of the proposed scheme in standard model. Finally, we draw the conclusions in section 6.

2. Related Works

Dent [7] describes generic constructions for provably secure KEMs based on weak encryption algorithms and analyses the two most popular techniques for constructing a KEM. Then he presents several simple approaches to constructing a KEM based on weak assumption.

Several key encapsulation mechanisms have been devised in recent years. Smart [8] devises a key encapsulation to multiple parties based on the Diffie-Hellman problem. In his mechanism, the sender can encapsulate the "session key" for several recipients and the KEM takes multiple public keys as input. He investigates the naive concatenation method and proves its security in standard model. Finally, he presents a public key mKEM based on DDH problem and proves its security in random oracle model.

Barbosa and Farshim [13] present the concept of identity based key encapsulation to multiple parties and design a mID-KEM. They prove their mechanism in the random oracle model under DDH assumption.

The notion of "atomic proxy cryptography" was presented by Blaze et al. [11] in 1998. It provides securer and more efficient way than usual to deal with the scenario in which a proxy decrypts a ciphertext using Alice's private key and then encrypts the result using Bob's public key.

In 2003, Ivan and Dodis [14] designed proxy encryption for Elgamal, RSA, and an IBE scheme using secret sharing technique. In their Elgamal based scheme, PKG generates encrypt key EK and decrypt key DK for each user, and then DK is divided into two parts x_1 and x_2, which satisfy DK= $x_1 + x_2$. Moreover, they designed unidirectional and bidirectional proxy encryption scheme.

Recently, Canetti and Hohenberger [12] proposed a proxy re-encryption scheme secure against chosen ciphertext attack. They discuss its security in standard model. There are some other re-encryption schemes, such as Jakobsson's quorum controlled asymmetric proxy re-encryption [15], and the identity-based scheme presented by Green and Ateniese [16].

3. Background

3.1. Preliminaries

Let G_1 be a cyclic multiplicative group generated by g, whose order is a prime q and G_2 be a cyclic multiplicative group of the same order q. Assume that the discrete logarithm in both G_1 and G_2 is intractable. A bilinear pairing is a map $e: G_1 \times G_1 \to G_2$ and satisfies the following properties:

1. *Bilinear:* $e(g^a, p^b) = e(g, p)^{ab}$. For all g, $p \in G_1$ and $a, b \in \mathbb{Z}_q$, the equation holds.
2. *Non-degenerate:* There exists $p \in G_1$, if $e(g, p) = 1$, then $g = O$.
3. *Computable:* For g, $p \in G_1$, there is an efficient algorithm to compute $e(g, p)$.
4. *commutativity:* $e(g^a, p^b) = e(g^b, p^a)$. For all g, $p \in G_1$ and $a, b \in \mathbb{Z}_q$, the equation holds.

Typically, the map e will be derived from either the Weil or Tate pairing on an elliptic curve over a finite field. Pairings and other parameters should be selected in proactive for efficiency and security [17].

— ***Decisional Bilinear Diffie-Hellman Assumption*** [18]

We say that an algorithm π that outputs $b \in \{0,1\}$ has advantage ε in solving the **Decisional Bilinear Diffie-Hellman** (**DBDH**) problem in G_1 if

$$|\Pr[\pi(g, g^a, g^b, g^c, e(g,g)^{abc}) = 0] - \Pr[\pi(g, g^a, g^b, g^c, T) = 0]| \geq \varepsilon$$

where the probability is over the random bit of π, the random choice of $a, b, c \in \mathbb{Z}_q^*$, and the random choice of $T \in G_2$. The **DBDH** problem is intractable if there is no attacker in G_1 can solve the **DBDH** with non-negligible ε.

3.2. Security notions

The proposed proxy key re-encapsulation scheme consists of five algorithms, namely **KeyGen**,

ReKeyGen, **Encap**, **ReEnc** and **Decap**.

— **KeyGen**(1^{λ}). On input the security parameter, outputs the public key PK of each group and the corresponding private key d_i for each member.

— **ReKeyGen** (sk_1, sk_2) **.** On input two private key sk_1 and sk_2, outputs a bidirectional re-encapsulation key $rk_{1\leftrightarrow 2}$.

— **Encap** (PK, s) **.** On input a random number $s \in \mathbb{Z}_q^*$ and a public key PK, outputs a ciphertext C.

— **ReEnc** $(rk_{1\leftrightarrow 2}, C_1)$ **.** On input ciphertext C_1 and the re-encapsulation key $rk_{1\leftrightarrow 2}$, outputs a ciphertext C_2 or an error symbol $\perp$.

— **Decap** (sk, C) **.** On input ciphertext C and a private key sk, outputs the corresponding encapsulated key.

The indistinguishable chosen ciphertext attack (IND-CCA) [19] presented by Goldwasser and Micali has been widely used to analyze the security of an encryption scheme. In this model, several queries are available to the attacker to model his capability. Subsequently, Rackhoff and Simon [20] enhanced it and proposed adaptively chosen ciphertext attack (IND-CCA2). Since this notion is stronger, it is becoming a prevalent model in analyzing encryption scheme. Green and Ateniese [16] enhanced the model and used it to discuss the security of proxy re-encryption scheme, then followed by Canetti and Hohenberger [12].

In this part, we define adaptively chosen ciphertext security of the group-based proxy re-encapsulation scheme. Compared to the model mentioned in [12], the security is defined using the following game between an *Attacker (Alice)* and *Challenger (Bob).*

1. **Setup.** The *Challenger* initializes the system and gives the *Attacker* the resulting system parameters and the public key PK. It keeps private key to itself.
2. **Query phase 1.**
   - **Decapsulate queries.** The *Attacker* issues a query (c_{i1}, c_{i2}). The *Challenger* outputs **Decapsulate** (c_{i1}, c_{i2}), otherwise outputs error symbol $\perp$.
   - **Re-encapsulate queries**. The *Attacker* issues a query (c_{i1}, c_{i2}) encrypted using the public key of group A. The *Challenger* outputs **Re-encapsulate** $(rk_{A\leftrightarrow B}, \tilde{c}_{i1}, \tilde{c}_{i2})$. Obviously, the output is a ciphertext encrypted using the public key of group B.

   The *Attacker* is allowed to perform the **Query phase 1** several times.
3. **Challenge.** Once the *Attacker* decides that **Query phase 1** is over, the *Challenger* outputs two messages $\{c_1^*, c_2^*\}$ and T^* to the *Attacker*.
4. **Query phase 2.** The *Attacker* continues to adaptively issue **Decapsualte** queries and **Re-encapsualte** queries. The *Challenger* responds as in the phase 1. These queries may be asked adaptively as in **Query phase 1**, but the query on (c_1^*, c_2^*) is not permitted.
5. **Guess.** Upon receiving the messages, the *Attacker* guesses whether the encapsulated key is equal to T^*. If it is true, outputs $bit = 1$, otherwise outputs $bit = 0$.

The encryption scheme is secure against chosen ciphertext attack, if the *Attacker* has a negligible advantage ε to win the game.

4. The Proposed Proxy Re-Capsulation Scheme

We assume that there exist two groups in our scheme, namely A and B. The function of the Proxy is to transform ciphertext corresponding to the public key of group A into ciphertext for the public key of group B without revealing any information about the secret decryption keys or the encapsulate key. It means that our proxy re-encapsulation is a bidirectional scheme. The proposed scheme consists of following steps.

4.1. Initialize

Let G_1 be a cyclic multiplicative group generated by g, whose order is a prime q and G_2 be a cyclic multiplicative group of the same order q. A bilinear pairing is a map: $e:\mathrm{G}_1 \times \mathrm{G}_1 \to \mathrm{G}_2$ that can be efficiently computed.
PKG chooses $a, b \in \mathbb{Z}_q^*$ and $h \in \mathrm{G}_1$ uniformly at random, and then computes $g_1 = g^a$ and $g_2 = g^b$. The master private keys are a and b, and the master public keys are g_1, g_2 and h. Define one cryptographic hash functions $H: G_1 \to Z_q$

4.2. Key Generation

PKG chooses $l \in \mathbb{Z}_q^*$ uniformly at random as the tag of the group B. Using $PK_{B1} = g_1^{l^{-1}}$, $PK_{B2} = h^b$, $PK_{B3} = h^{l^{-1}}$ as group B's public keys. The private keys of the member $p_i \in B$ can be generated as follows:

1. PKG chooses $m_i \in Z_q^*$ uniformly at random and computes $n_i \in \mathbb{Z}_q^*$, such that $l \equiv (m_i + n_i) \bmod q$.
2. compute and output $d_{i1} = g_2^{m_i}$, $d_{i2} = g_2^{a n_i l^{-1}}$, and $d_{i3} = h^{b n_i l^{-1}}$.

The member p_i's private key is $d_i = \{d_{i1}, d_{i2}, d_{i3}\}$. PKG chooses $k \in \mathbb{Z}_q^*$ uniformly at random as the tag of the group A and publishes $PK_{A1} = g_1^{k^{-1}}$, $PK_{A2} = h^b$, $PK_{A2} = h^{k^{-1}}$, as group A's public keys. The private keys of the member $p_i \in A$ can be similarly generated as above.

4.3. Encapsulate

In order to encapsulate a key for the group A, the sender first chooses $s \in \mathbb{Z}_q^*$ uniformly at random, and computes the ciphertext

$$c_1 = g^s \qquad c_2 = ((PK_{A1})^z \cdot (PK_{A3}))^s .$$

The encapsulated key is $Key_s = e(g_1, g_2)^{zs}$, where $z = H(c_1)$. The sender sends (c_1, c_2) to the group A by broadcast over the Internet.

4.4. Re-encapsulate

In order to transform the ciphertext to group B, PKG generates a Re-encapsulation key $rk_{A \leftrightarrow B} = kl^{-1}$, and send it to Proxy. Then using the Re-encapsulation key the proxy can perform

$$\begin{aligned} \tilde{c}_1 &= c_1 \\ \tilde{c}_2 &= (c_2)^{zs(rk_{A\leftrightarrow B})} = (g_1^{k^{-1}z} \cdot h^{k^{-1}})^{s \cdot k \cdot l^{-1}} \\ &= (g_1^{l^{-1}z} \cdot h^{l^{-1}})^s = ((PK_{B1})^z \cdot (PK_{B3}))^s \end{aligned}$$

The Re-encapsulated ciphertext is $(\tilde{c}_1, \tilde{c}_2)$.

4.5. Decapsulate

After receiving the ciphertext $(\tilde{c}_1, \tilde{c}_2)$, the member $p_i \in B$ computes $z = H(c_1)$ and decapsulates as follows, otherwise outputs $\perp$ and rejects the ciphertext.

$$Key_s = e(\tilde{c}_2, d_{i1}) e(\tilde{c}_1, d_{i2}^z d_{i3}) / e(\tilde{c}_1, PK_{B2})$$

Any member $p_i \in B$ can decapsulate the ciphertext since

$$\begin{aligned} Key_s &= e(\tilde{c}_2, d_{i1}) e(\tilde{c}_1, d_{i2}^z d_{i3}) / e(\tilde{c}_1, PK_{B2}) \\ &= e((PK_{B1})^{zs} \cdot (PK_{B3})^s, g_2^{m_i}) e(g^s, g_2^{azn_i l^{-1}} h^{bn_i l^{-1}}) / e(g^s, h^b) \\ &= e(g_1^{zl^{-1}s} h^{l^{-1}s}, g_2^{m_i}) e(g^s, g_2^{azn_i l^{-1}}) e(g^s, h^{bn_i l^{-1}}) / e(g^s, h^b) \\ &= e(g_1, g_2)^{l^{-1}zsm_i} e(h^{l^{-1}}, g_2)^{sm_i} e(g_1, g_2)^{l^{-1}zsn_i} e(g^s, h^{bn_i l^{-1}}) / e(g^s, h^b) \\ &= e(g_1, g_2)^{zsl^{-1}(m_i+n_i)} e(h^{l^{-1}}, g_2)^{s(m_i+n_i)} / e(g_2^s, h) \\ &= e(g_1, g_2)^{zs} \end{aligned}$$

To the user in group A, he can get the decapsulated key from (c_1, c_2) similarly to the user in group B.

5. Security

In this section, we will discuss the security of the proposed proxy key re-encapsualtion scheme in standard model.

Theorem. *Suppose that the* ***DBDH*** *is intractable. Then our proxy key re-encapsulation scheme is secure against adaptively chosen ciphertext attack.*

Proof. Assume that if the attacker Alice has ability to break the proposed proxy key re-encapsulation scheme via chosen ciphertext attack with non-negligible probability ε, then we can prove that there exists challenger Bob that can solve **DBDH** problems with the same probability. In other words, given $g^a, g^b, g^c \in \mathrm{G}_1$ and $T \in \mathrm{G}_1$, Bob can decide if T is equal to $e(g,g)^{abc}$ with non-negligible probability by running Alice as a subroutine. The challenger Bob interacts with Alice by simulating **Decapsulate**, **Re-encapsulate** oracles.

Bob initializes the system, chooses random numbers $u, k \in \mathbb{Z}_q^*$. Let

$$g_1 = g^a \qquad g_2 = g^b$$
$$PK_{A1} = g_1^{k^{-1}} \qquad PK_{A2} = h^b \qquad PK_{A3} = h^{k^{-1}}$$
$$z^* = H(g^c) \qquad h = g_1^{-z^*} \cdot g^u .$$

Then Bob chooses a random number $\alpha \in \mathbb{Z}_q^*$ and publishes $PK_{B1} = g_1^{k^{-1}\cdot\alpha^{-1}}$, $PK_{B2} = h^b$ and $PK_{B2} = h^{k^{-1}\alpha^{-1}}$ as the public keys of group B.

Query phase 1.

- **Decapsulate queries**. To every new query (c_1, c_2), Bob computes and outputs $Key_s = e((c_2^k / c_1^u)^{z/(z-z^*)}, g_2)$ as the answer. We say Bob can output Key_s since
$$e((c_2^k / c_1^u)^{z/(z-z^*)}, g_2) = e((g_1^{k^{-1}\cdot z} \cdot g^{a\cdot k^{-1}\cdot z^*} \cdot g^{u\cdot k^{-1}})^{s\cdot k} / g^{s\cdot u})^{z/(z-z^*)}, g_2)$$
$$= e((g_1^{(z-z^*)} \cdot g^u)^s / g^{u\cdot s})^{z/(z-z^*)}, g_2)$$
$$= e(g_1^{z\cdot s}, g_2) = Key_s$$
- **Re-encapsulate queries**. To every new query (c_1, c_2), Bob computes
$$\tilde{c}_1 = c_1 \qquad \tilde{c}_2 = (c_2)^{\alpha^{-1}}$$

And then, Bob outputs $(\tilde{c}_1, \tilde{c}_2)$ as the answer.

Since $w, \alpha \in \mathbb{Z}_q^*$ are two random number, Alice can't distinguish the simulated answers from the actual results. Thereby, we say above simulation is perfect. Alice is allowed to perform **Decapsulate** and **Re-encapsulate** queries several times.

Challenge phase. When Alice decides Query phase 1 is over, Bob generates the challenge ciphertext.

$$c_1^* = g^c \qquad c_2^* = (g^c)^{u\cdot k^{-1}} .$$

The **Challenge phase** can be performed only once. We say (c_1^*, c_2^*) is a valid ciphertext since

$$c_2^* = ((PK_{A1})^z \cdot (PK_{A3}))^c = (g_1^{k^{-1}\cdot z} \cdot g_1^{-k^{-1}\cdot z^*} \cdot g^{u\cdot k^{-1}})^c$$

Note that $z = H(g^c) = z^*$, and then we have $c_2^* = (g^c)^{u\cdot k^{-1}}$. In this instance, the encapsulated key is $Key_c = e(g,g)^{abcz}$. Bob sends (c_1^*, c_2^*) and T^z to Alice as the challenge.

Query phase 2. Alice continues to adaptively issue **Decapsulate** and **Re-encapsulate** queries. Bob responds as in the phase 1. However, the query on (c_1^*, c_2^*) is not permitted.

Guess. After receiving the challenge message (c_1^*, c_2^*) and T^z, if the encapsulated key is T^z, Alice outputs $bit = 1$, otherwise outputs $bit = 0$. Thereafter, if Alice guesses $bit = 1$, Bob guesses $e(g,g)^{abc} = T$, otherwise guesses $e(g,g)^{abc} \neq T$.

Obviously, above simulation is perfect. We say that Alice can break the proxy re-encapsulate scheme with non-negligible probability ε. It means that Alice can output correct bit with probability ε. Then Bob can solve the **DBDH** with same probability ε by running Alice as a subroutine.

□

6. Conclusions

The notion of proxy cryptography is very useful in cases when one user needs to perform sensitive operation without holding the necessary secret key. This technology can be used in group-based key encapsulation mechanism. In this paper we design a proxy key re-encapsulation key mechanism, and analysis it security. As an important part of hybrid encryption scheme, it can improve the performance of the group communication, especially in the scenario of real-time communication.

References

1. D. Boneh, , C. Gentry, and B. Waters. Collusion Resistant Broadcast Encryption With Short Ciphertexts and Private Keys. In Advances in Cryptology-CRYPTO 2005. Springer-Verlag, Lecture Notes in Computer Science 3621, 258-275.
2. A. Fiat, and M. Naor. Broadcast Encryption. In Advances in Cryptology-CRYPTO, 1993. Springer-Verlag, Lecture Notes in Computer Science 773, 480-491.
3. Y. Dodis, and N. Fazio. Public key broadcast encryption secure against adaptive chosen ciphertext attack. In Workshop on Public Key Cryptography (PKC) 2003. Lecture Notes in Computer Science 2567,100-115.
4. D. Halevy and A. Shamir. “The 1sd broadcast encryption scheme”. In Proceedings of Cryptology-CRYPTO 2002. Springer-Verlag, Lecture Notes in Computer Science 2442: 47-60.
5. M. Luby and J. Staddon. “Combinatorial Bounds for Broadcast Encryption”. In Advances in Cryptology-EuroCrypt’98. Springer-Verlag, Lecture Notes in Computer Science 1403: 512-526.
6. R. Cramer and V. Shoup. A practival public key cryptiosystem provably secure against adaptive chosen ciphertext attack. Crypto’98, LNCS 1462, pp. 13-25.
7. W. Dent. A designer’s guide to KEMs. In Coding and Cryptography, Springer-Verlag LNCS 2898, pp. 133-151, 2003.
8. N. P. Smart. Efficient key encapsulation to multiple parties. In Proceedings SCN, 2004. LNCS 3352, pp. 208-219. 2005.
9. M. Stam. A key encapsulation mechanism for NTRU. In Coding and Cryptography, LNCS 3796, pp. 410-427, 2005.
10. M. Mambo and E. Okamoto. Proxy Cryptosystems: Delegation of the Power to Decrypt Ciphertexts. IEICE Trans. Fund. Electronics Communications and Computer Science, E80-A/1: 54-63, 1997.
11. M. Blaze, G. Bleumer, and M. Strauss. Divertible protocols and atomic proxy cryptography. In EUROCRYPT’98, LNCS 1403: 127-144.
12. R. Canetti, S. Hohenberger. Chosen-Ciphertext Secure Proxy Re-Encryption. Available at http://eprint.iacr.org/2007/171
13. M. Barbosa and P. Farshim. Efficient identity-based key encapsulation to multiple parties. Proc. Cryptography and Coding, Springer LNCS 3796, pp 428-441, 2005
14. A. Ivan, Y. Dodis. Proxy cryptography revisited. In Proceedings of the Tenth Network and Distributed System Security Symposium. February, 2003.
15. M. Jakobsson. On quorum controlled asymmetric proxy re-encryption. In Proceedings of Public Key Cryptography, 1999, 112-121.
16. M. Green, G. Ateniese. Identity-based Proxy Re-encryption. In Proceedings of ACNS 2007,

LNCS 4521: 288-306.

17. D. Boneh, B. Lynn, and H. Shacham. Short signatures from the Weil pairing. Advances in Cryptology -- Asiacrypt'2001, Gold Coast, Australia, Lecture Notes in Computer Science, 2248, Springer-Verlag (2001) 514-532.

18. D.Boneh and X. Boyen. Efficient Selective-ID Secure Identity Based Encryption Without Random Oracles. Advances in Cryptology Eurocrypt 2004. Berlin:Springer-Verlag,2004: 223-238.

19. S. Goldwasser and S. Micali. Probabilistic Encryption. Journal of Computer and System Sciences, 1984, 28: 270-299.

20. C. Rackhoff and D. R. Simon. Non interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In Advanced in Cryptology-CRYPTO'91. Springer-Verlag, 1992: 434-444.