Proxy Cryptography for Secure Inter-domain Information Exchanges

Shuanghe Peng Zhen Han
Research Center of Information Security Architecture, Beijing Jiaotong University
Beijing 100044,China
Shhpeng@sohu.com

Abstract

In today's interconnected world, the need to access to services across domains is increasing. How to make inter-domain data exchanges secure is what we concerned. Kerberos protocol can be used to secure inter-domain information exchanges, but it cannot be applied directly in environment where boundary device such as firewall exists between domains. Proxy ElGamal encryption scheme is a way to protect data exchanges between two domains. However, there are two disadvantages in using this proxy cryptosystem. In this article, an improved proxy ElGamal cryptosystem is proposed and two secure inter-domain authentication and proxy keys distribution protocols based on the improved proxy ElGamal cryptosystem are designed, one is for trusted domains and the other is for untrusted domains. The proposed protocols make data exchanges between domains more secure and efficient.

1 Introduction

Under Internet environment, we can regard the security domain as a connected cluster of communicating entities, consisted of different subjects and objects, which are protected by communication security policies. Conceptually, every communicating entity on the Internet must be a member of one or more security domains to which it belongs directly or indirectly. A communication security policy is required to protect specific network traffic in or out of a security domain. The demand for protecting the privacy and integrity of messages as they traverse security domains has been on increase in recent years. As a result, securing inter-domain communications, such as providing confidentiality, authenticity, and integrity of messages delivered between domains, is a critical issue. To solve this problem, we must find an effective way to prevent unauthorized users from accessing the content of delivered messages between domains.

Kerberos was developed by the MIT in the late 1980s [l, 2] and there are several versions of the protocol. The most popular ones are Versions 4 and 5, with Version 5 being the most recent. The basic messages in the Versions 4 and 5 protocols are the same although the details of the encoding differ [3]. In Kerberos, some server principals are special principals and are trusted by all other principals in that domain. These are the Authentication Server (AS) principal (sometimes also referred to as the Key Distribution Centre or KDC) and the Ticket Granting Server (TGS). Sometimes both the AS and the TGS are collectively referred to as the KDC. The AS is involved in the authentication of principals whereas the TGS's primary role is in the distribution of tickets which allow access to other servers. In many implementations, the AS and the TGS are often co-located. In an inter-domain situation, it is not generally the case that a principal in one domain trusts the AS and TGS of another domain. A principal in a domain trusts the AS and TGS in its own domain, and there is trust between AS and TGS in one domain with their counterparts in another domain. Kerberos protocol can be used to secure inter-domain information exchanges, but it cannot be applied directly in environment where gateway device exists between domains. In this condition, principals in a domain cannot establish communication channel directly with service providers in another domain. All inter-domain communication must be passed through gateway. The communication entities on the path from requester to response are client, local gateway, remote gateway and the service provider accordingly.

Proxy ElGamal cryptosystem is a method to secure data exchanges between domains. But there are two disadvantages in proxy ElGamal cryptosystem. One is the need to break large plaintext into smaller pieces before enciphering and the other is the need to generate a random number for each piece. So in this article, an improved proxy ElGamal

cryptosystem which allows a large confidential message to be encrypted efficiently is provided. Then two secure inter-domain authentication and proxy keys distribution protocols based on the improved proxy ElGamal cryptosystem are designed, one is for trusted domains and the other is for untrusted domains. The proposed protocols make data exchange between domains security and efficiency.

2 ElGamal Proxy Cryptosystem

For reader's conveniency, we first introduce the notation as follows:

p: a large prime number, its length is greater than 512 bits.

g: a generator for Z_p^*.

(x, y): a pair of public and private key, in which $y=g^x \bmod p$, where $x \in_R Z_{p-1}\backslash\{0\}$.

m: the original message.

H(.):a secure one-way hash function, such as SHA-1 [4].

||: denotes concatenation operation.

$K_{h1,\ h2}$: shared symmetric encryption key of h1 and h2.

e_h: encrypt public key of holder h.

d_h : decrypt private key of holder h.

E(m, k) or $\{m\}_k$: the ciphering of a message m by a key k.

D(c, k): the deciphering of a ciphertext c by a key k.

N_X: the nonce issued by principal X to prevent from replay attack.

U, U_{ID}: U is the requesting user with identity U_{ID}.

S, S_{ID}: S is the service provider with identity S_{ID}.

The primitive method of delegating decryption is showed in Figure 1. That is to "decrypt and re-encrypt". The original message is disclosed to the delegator and two different encrypt/decrypt key pairs are used, which make it inefficient at all.

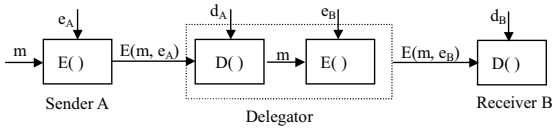

Figure 1. Primitive Method of Delegating Decryption

The concept of proxy cryptosystem was first introduced in [5]. An atomic function was introduced in [6]. An atomic function is a function used to convert a ciphertext for one key into a ciphertext for another key without revealing the secret decryption keys nor cleartext messages.

We show proxy cryptosystem in Figure 2.We will denote by *II* the atomic proxy function and by $\pi_{A\to B}$ the proxy key used to convert text encrypted with the public key e_A held by sender A to a ciphertext for receiver B such that: $D(II(E(m,e_A),\pi_{A\to B}),d_B)=m$.

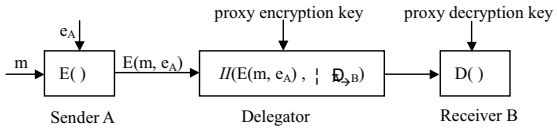

Figure 2 Proxy Cryptosystem

Function *II* is atomic in the sense that it computes $E(D(c,d_A),e_B)$ on a ciphertext c without revealing the intermediate result $D(c,d_A)$.

2.1 ElGamal Cryptosystem

In ElGamal cryptosystem[7] we define $\kappa=\{(p,g,x,y): y\equiv g^x \bmod p\}$, where (p, g, y) is public keys , while x is private key .To encrypt a plaintext m, a random integer $r \in_R Z_{p-1}\backslash\{0\}$ is selected .The cipher text consists of the pair (b, c) computed as follow:

$b = g^r \bmod p$ (1)

$c = my^r \bmod p$ (2)

To decrypt the cipher text (b, c) using private key x, we compute $c/b^x \bmod p$, the result is plaintext m.

The correctness of the ElGamal encryption scheme is easy to verify, we have

$c/b^x \bmod p = my^r (b^x)^{-1} \bmod p$

$=m\, g^{xr}(g^{rx})^{-1} \bmod p$

$=m$ (3)

2.2 Proxy ElGamal Encryption Scheme

In the proxy ElGamal encryption scheme, the private key x is split into x_1 and x_2 such that $x=x_1+x_2$. This split can be made when required and there can be a very large number of such possible splits resulting in different values of x_1 and x_2. Proxy changing algorithm receives x_1 and the receiver receives x_2. Proxy changing function and decryption function are similar to the original decryption function under x_1 and x_2 respectively. Proxy changing function thus performs a partial decryption given by

$c' \leftarrow c/b^{x_1} \bmod p$

(4)

The decryption function performs the decryption by performing

$m \leftarrow c' / b^{x_2} \bmod p$ (5)

The correctness of the Proxy El Gamal encryption function is easy to verify. We have

$c' / b^{x_2} \bmod p = (c / b^{x_1} \bmod p) / b^{x_2} \bmod p$

$= c / (b^{x_1+x_2}) \bmod p$

$= c / b^{x} \bmod p$ (6)

From here on we call the key x as the encryption key, the key x_1 as the proxy encryption key, the key x_2 as the proxy decryption key.

However, there are two disadvantages in the proxy ElGamal cryptosystem. One is that the plaintext m must be less than p-1. The other is that the random number r cannot be used repeatedly. Otherwise, the system is not secure against the know-plaintext attack. As long as we know the pair (c',c) of equation (5), b^{x_1} is obtained . When we know the pair (m, c') of equation (6), b^{x_2} is obtained. Then, the same random number r is used to encrypt other pieces of plaintext $\tilde{m}$ as follows :

$b = g^r \bmod p$ (7)

$\tilde{C} = \tilde{m}\, y^r \bmod p$ (8)

$\tilde{c}' = \tilde{c} / b^{x_1} \bmod p$ (9)

then the plaintext $\tilde{m}$ can be obtained by computing

$\tilde{c}' / b^{x_2} = \tilde{c} / (b^{x_1} b^{x_2}) = \tilde{c} / (b^{x}) = \tilde{m} \bmod p$ (10)

Therefore, the proxy ElGamal cryptosyatem needs to break a large plaintext into smaller pieces before enciphering, with the length of each piece being less than that of p. The proxy ElGamal cryptosystem also needs to generate a random r for each piece. It is obvious that the proxy ElGamal cryptosystem is time-consuming for encrypting a large plaintext. In next section , we propose an improved proxy El Gamal encryption scheme which makes a large plaintext to be encrypted without generating different random number r for each piece of plaintext.

3 An Improved Proxy ElGamal Encryption Scheme

In this section, an efficient proxy ElGamal encryption scheme is proposed for enciphering a large plaintext. The proposed scheme is based on the proxy ElGamal encryption scheme.

When a sender wants to deliver a confidential message m to a receiver, the following steps are performed:

Step 1. The sender breaks plaintext m into t pieces $m_1, m_2, \ldots, m_t$, each piece of length being 512 bits .

Step 2. The sender generates two random number r_1 and r_2, where r_1, $r_2 \in_R Z_{p-1} \backslash \{0\}$, and computes b_1 and b_2 as follows:

$b_1 = g^{r_1} \bmod p$ (11)

$b_2 = g^{r_2} \bmod p$ (12)

Step 3. The sender computes c_j, j=1,2,…, t, as follows:

$c_j = m_j \, (y^{r_1} (y^{r_2})^{2^j}) \bmod p$ (13)

We assume j = 0 as the preparation phase for sending message.

Step 4. The sender sends {b_1, b_2, c_j ,j=1,2,…,t} to the proxy through a public network .

After receiving {b_1, b_2, c_j ,j=1,2,…,t } from the sender, the proxy translates the ciphertext as follows:

$c_j' = c_j / (b_1^{x_1} (b_2^{x_1})^{2^j}) \bmod p$ (14)

Step 5. Proxy sends {b_1, b_2, c_j', j=1,2,…,t } to the receiver through a public network.

After receiving {b_1, b_2, c_j', j=1,2,…,t } from the proxy, the receiver recovers the plaintext m by computing as follows:

$m_j = c_j' / (b_1^{x_2} (b_2^{x_2})^{2^j}) \bmod p$ (15)

The correctness of our improve proxy ElGamal encryption scheme is easy to verify. We have

$c_j' / (b_1^{x_2} (b_2^{x_2})^{2^j}) \bmod p$

$= (c_j / (b_1^{x_1} (b_2^{x_1})^{2^j}) \bmod p) / (b_1^{x_2} (b_2^{x_2})^{2^j}) \bmod p$

$= c_j / ((b_1^{x_1} b_1^{x_2})(b_2^{x_1} b_2^{x_2})^{2^j}) \bmod p$

$= c_j / ((b_1^{x_1+x_2})(b_2^{x_1+x_2})^{2^j}) \bmod p$

$= c_j / (b_1^{x} (b_2^{x})^{2^j}) \bmod p$

$= c_j / ((g^{r_1})^{x} ((g^{r_2})^{x})^{2^j}) \bmod p$

$= c_j / (y^{r_1} (y^{r_2})^{2^j}) \bmod p$

$= m_j$ (16)

It should be noted that the sender generates only two random numbers r_1 and r_2 in our scheme, while the ElGamal cryptosystem requires t random numbers for t pieces of plaintext.

4 Communication Scheme for Inter Domain Data Exchange

In the inter domain authentication, we assume each domain has a domain manager and the domain manager acts as the authentication and authority center for its domain which makes the application system free from access control. This way improves the efficient of resource management. These domain

managers are organized as DNS-based PKI[8] tree hierarchy. In every sub-tree of the hierarchy, the domain manager shares a common key with each of his child node and his parent node.

When data is sent from one domain to another domain, it has to be checked by its local domain manager before sent to remote domain. When a user wants to request a service from a remote domain, the information flow of inter-domain is depicted in Figure 3.

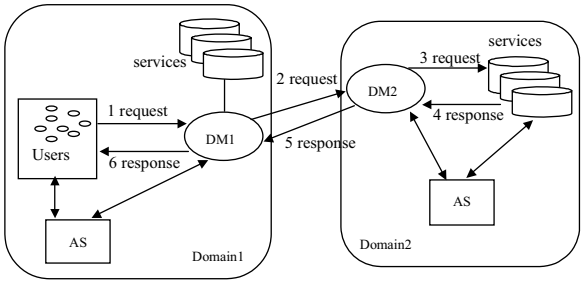

Figure 3 Secure scheme for inter-domain services access

First, mutual authentication between users and its local domain manager must be performed by means of authentication server AS in the request domain. At the same time, mutual authentication between service and its local domain manager must be performed by means of authentication server AS in the response domain. After authentication, data exchange can be performed.

According to section 2 and 3, in order to protect the data exchange between two domains, the main problem is how to securely distribute private proxy encryption key x_1 and proxy decryption key x_2 to proper entity. One is to the proxy and the other to the receiver.

The way to distribute the private proxy encryption key x_1 and proxy decryption key x_2 depends fully on the relationship between domain managers. We category the relationship between domain managers into two, one is trusted and the other is untested. Session key between trusted domain managers can be obtained by negotiation themselves, such as Diffie-Hellman key exchange algorithm. While session key obtained between untrusted domain managers must be supported by a trust third party.

After obtaining the correspondence key, supposed user U_i in domain1 wants to send data m to service provider S_j in domain2, the process describes as follows, see figure 3:

Step 1. U_i encrypts data m into triple (b_1, b_2, c) according to equations (11), (12) and (13) and then sends it to local domain manager DM1.

Step 2. DM1 re-encrypts the ciphertext c into c' by using proxy encryption key x_1, as equation (14) does, and then sends the triple (b_1, b_2, c') to the receiver DM2.

Step 3. DM2 decrypts the triple (b_1, b_2, c') by using proxy decryption key x_2, as equation (15) does, and then obtains the plaintext m, then forwards the plaintext m to service provider S_j by using standard HTTP protocol or correspondence TCP port according to different service provided by service provider.

Step 4~6.After service provider S_j gets the plaintext request m, the resource response return to users is depicture in step 4 ~ 6 just as step 1~3 does.

4.1 Authentication and Secure Proxy Encryption Key Distribution between Trusted Domains

If domain managers DM1 and DM2, as Figure 3 shows, trust each other, the session key $K_{DM1,DM2}$ between them can be obtained by using Diffie-Hellman key exchange algorithm. The message sequences of the authentication and the proxy encryption key distribution can be described in Figure 4.

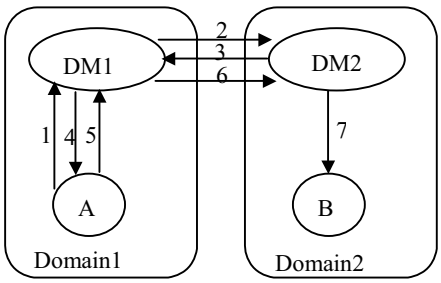

Figure 4. Data Exchange between Trusted Domains

We discuss the case where domain1 is the requester domain while domain2 is the resource domain. In this case, the receiver, i.e. DM2, can generate the private key x and split it into x_1 and x_2, and then reserve x_2 himself and securely distribute x_1, the proxy encryption key, to the proxy, i.e. DM1, by using session key between them.

Message 1. A→ DM1: A, B, N_A

Message 2. DM1 → DM2: {N_A, N_{DM1}, A, B}$K_{DM1,DM2}$

Message 3. DM2→ DM1: {A, B, N_A, N_{DM1}-1, p, g, y, x_1, N_{DM2}} $K_{DM1,DM2}$

Message 4. DM1 → A: p, g, y

Message 5. A→ DM1: b_1, b_2, c_j

Message 6. DM1 → DM2: $\{N_{DM2}+1\}\ K_{DM1,DM2}$, b_1, b_2, c_j'

Message 7. DM2 → B: M_j

In message 1, user A initiates the service request by sending his identity and a nonce to local domain manager DM1. In message 2, DM1 judges the request to find that the service is located in the remote domain. Then it encrypts the received request, his identity and a nonce using $K_{DM1, DM2}$ and forwards it to the remote domain manager DM2. In message 3 DM2 decrypts the received request and generates the ElGamal public parameters p, g, y and private key x and splits x into x_1 and x_2, then sends $\{A, B, N_A, N_{DM1}-1, p, g, y, x_1, N_{DM2}\}\ K_{DM1,DM2}$ to DM1. In message 4, DM1 decrypts and obtains proxy encryption key x_1 and forwards the public information p, g and y to user A. In message 5, user A computes b_1, b_2 and c_j according to equations (11) , (12) and (13) and sends them to DM1.In message 6, DM1 sends his authentication information $\{N_{DM2}+1\}\ K_{DM1, DM2}$ and b_1, b_2 and c_j' according to equations (14) to DM2.

We compute $H(m_j)$ and attach it to m_j to generate $M_j = m_j||H(m_j)$. Message M_j will be encrypted into ciphertext c_j by user A. Ciphertext c_j will be re-encrypted into the ciphertext c_j' by DM1. Further on, when c_j' is decrypted by DM2, m_j can be verified by its hash value.

The return data from B to A can adopt the same method as the request data does.

4.2 Authentication and Secure Proxy Encryption and Decryption Key Distribution between Untrusted Domains

While in untrusted case, the generation of the private key x and the distribution of its correspondence proxy encryption key and proxy decryption key , i.e.x_1 and x_2, can only be done by the trusted third party.

As compared with the above, this case becomes more complicated. Since they cannot authenticate each other only depending on themselves. They must authenticate the identities with each other with the aid of a third party being trusted by two of them, i.e. the domain manager DM0, which is the parent node of DM1 and DM2. We supposed that session key between DM0 and DM1 is $K_{DM1,DM0}$, session key between DM0 and DM2 is $K_{DM2,DM0}$, session key between DM1 and DM2 is $K_{DM1,DM2.}$ The message sequences of the authentication and proxy encryption key and proxy decryption key distribution of the untrusted domain managers can be described in Figure 5.

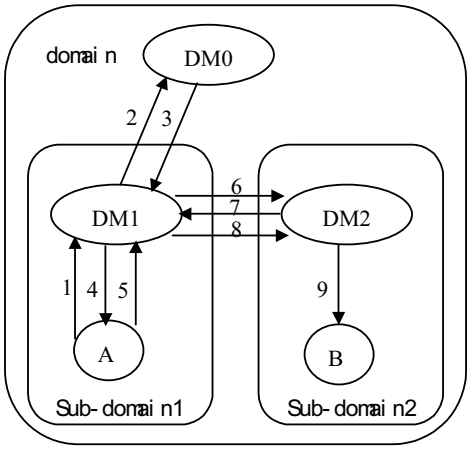

Figure 5 . Data Exchange between Untrusted Domains

Message 1. A→ DM1: A, B, N_A

Message 2. DM1 → DM0:
$\{N_A, N_{DM1}, A, B, DM1, DM2\}K_{DM1,DM0}$

Message 3. DM0→ DM1:
$\{A, B, DM1, DM2, K_{DM1,DM2}, N_A, N_{DM1}, p, g, y, x_1$
$\{DM1, DM2, K_{DM1,DM2}, N_{DM1}, p, g, y, x_2\}K_{DM2,DM0}\}K_{DM1,DM0}$

Message 4. DM1 → A: p, g, y

Message 5. A→ DM1: b_1, b_2, c_j

Message 6. DM1 → DM2:

$\{DM1, DM2, K_{DM1,DM2}, N_{DM1}, p, g, y, x_2\}K_{DM2,DM0}$

Message 7. DM2 → DM1:
$\{N_{DM1}-1, N_{DM2}\}K_{DM1,DM2}$

Message 8. DM1 → DM2: $\{N_{DM2}+1\}K_{DM1,DM2}, c_j'$

Message 9. DM2 →B: M_j

In message 1, user A initiates the service request by sending his identity and a nonce to local domain manager DM1. In message 2, DM1 judges the request to find that the service is located in the remote domain DM2 and the remote domain manager is not a trusted one. So it has to ask the aid of their parent node DM0 to generate session key between DM1 and DM2 and generate the ElGamal parameters. In

message 3, DM0 decrypts the request and authenticates DM1 and then generates the ElGamal public parameters p, g, y and private key x and splits x into x_1 and x_2, then sends $\{A,B,DM1,DM2,K_{DM1,DM2},N_A,N_{DM1},p,g,y,x_1$ $\{DM1,DM2,K_{DM1,DM2},N_{DM1},p,g,y,x_2\}K_{DM2,DM0}\}K_{DM1,DM0}$ to DM1. In message 4, DM1 decrypts and obtains proxy encryption key x_1 and forwards the public information p,g,y to user A. In message 5, user A computes b_1,b_2,c_j according to equations (11),(12) and (13) and sends them to DM1. In message 6, DM1 sends $\{DM1,DM2,K_{DM1,DM2},N_{DM1},p,g,y,x_2\}K_{DM2,DM0}$ to DM2. In Message 7, DM2 sends $\{N_{DM1}-1,N_{DM2}\}K_{DM1,DM2}$ to DM1. Then, DM1 authenticates DM2. In message 8, DM1 sends $\{N_{DM2}+1\}K_{DM1,DM2}$ and c_j' to DM2 for authentication.

The return data from B to A can adopt the same method as the request data does.

4.3 Security and Overhead Analysis

For convenience, the following notations are used to analyze the computational complexity:T_{mul} is the time for multiplication; T_{exp} is the time for exponentiation; and T_{inv} is the time for inversion.

4.3.1 Compute Cost Analysis

From user's view, at the preparation phase, i.e. j=0, only $4T_{exp}$ in equations (11), (12) and (13) is required, i.e. g^{r_1}, g^{r_2}, y^{r_1} and $(y^{r_2})^{2^j}$. And when plaintext is encrypted and sent, i.e. j > 0, we only need replace $(y^{r_2})^{2^j}$ with $(y^{r_2})^{2^{(j+1)}}$, which is equal to $(y^{r_2})^{2^j}.(y^{r_2})^{2^j}$, and then compute the ciphertext c_j as equation (13) does. So only $3T_{mul}$ is required.

From the local domain manager's view, at the preparation phase, i.e. j = 0, only $2T_{exp}$ is required in equation (14), i.e. $b_1^{x_1}$ and $(b_2^{x_1})^{2^j}$. And when ciphertext c_j is re-encrypted and sent, i.e. j > 0,we only need replace $(b_2^{x_1})^{2^j}$ with $(b_2^{x_1})^{2^{(j+1)}}$, which is equal to $(b_2^{x_1})^{2^j}.(b_2^{x_1})^{2^j}$, and then compute the ciphertext c_j' as equation (14) does. So only $(3T_{mul}+T_{inv})$ is required.

From the remote domain manager's view, at the preparation phase, i.e. j = 0, only $2T_{exp}$ is required in equation (15), i.e. $b_1^{x_2}$ and $(b_2^{x_2})^{2^j}$. And when ciphertext c_j' is received and decrypted, i.e. j >0,we only need replace $(b_2^{x_2})^{2^j}$ with $(b_2^{x_2})^{2^{(j+1)}}$, which is equal to $(b_2^{x_2})^{2^j}.(b_2^{x_2})^{2^j}$, and then obtain the plaintext m_j as equation (15) does. So only $(3T_{mul}+T_{inv})$ is required.

From the scheme of inter-domain data exchange described in 4.1 and 4.2, we know that when a large plaintext is sent, the preparation phase is needed only once. After preparation phase, when data is sent and received by different principals accordingly, only T_{mul} and T_{inv} is needed. So when a plaintext is divided into t pieces to be sent, the whole computational of it is $8T_{exp}+t(9T_{mul}+2T_{inv})$.

The overall computational complexity of different subjects and the total compute cost to send a large plaintext m in the proposed scheme is shown in table 1.

Table 1 Computational Complexity of inter-domain Data Exchange

Phase / Subject	Preparation	Send/Receive
User	$4T_{exp}$	$3T_{mul}$
Local Domain Manager	$2T_{exp}$	$3T_{mul}+T_{inv}$
Remote Domain Manager	$2T_{exp}$	$3T_{mul}+T_{inv}$
Total cost	$8T_{exp}+t(9T_{mul}+2T_{inv})$	

4.3.2 Security Analysis

From the communication scheme described above, we can see that when message m_j is sent from user A to the destination B, it does not reach the destination i.e. the service provider B directly. It is first encrypted into ciphertext c_j by sender A and then sent to local domain manager DM1.When DM1 receives the ciphertext c_j, he should re-encrypted it into ciphertext c_j' by using proxy encryption key x_1 and then sent it to remote domain manager DM2. When DM2 receives the ciphertext c_j', he should decrypted it by using proxy decryption key x_2 and then obtains the plaintext m_j. At last the plaintext m_j is forwarded to the destination B.

We can see from this process that message is kept

secret when it is transmitted across public network. It is difficult for an intruder to obtain the system-generated random number r_1 and r_2 directly from the equations (11) and (12). The difficulty relies on the complexity of computing discrete logarithms over finite fields.

If an intruder knows some plaintext and ciphertext triples, i.e. (m_j, c_j, c_j'), the intruder can obtain $(b_1^{x_1}(b_2^{x_1})^{2^j})$ and $(b_1^{x_2}(b_2^{x_2})^{2^j})$ by solving equation (14) and (15) . However, it is difficult for the intruder to obtain $b_1^{x_2}$ and $(b_2^{x_2})^{2^j}$ from $(b_1^{x_2}(b_2^{x_2})^{2^j})$. The security of it is based on the difficulty of factorization of a large integer. The scheme is thus secure against the chosen-plaintext attacks.

5 Conclusion

An improved proxy ElGamal cryptosystem is proposed and two inter-domain authentication and security proxy keys distribution protocols based on the improved proxy ElGamal cryptosystem are designed in this paper, one is for trusted domains and the other is for untrusted domains. The proposed scheme is more secure and efficient than the previous one. It is secure against the know-plaintext attack and it don't need one random number for each piece as the primitive proxy ElGamal cryptosystem does, only two random numbers r_1 and r_2 is needed to generate here, which allows a large confidentially message to be encrypted efficiently. The two inter-domain authentication and security proxy keys distribution protocols make data exchange between domains more secure and efficient.

Acknowledgments

Part of this research was supported by National High-Tech Research and Development Plan of China(Grant No.2002AA1Z2101).

Reference

[1] J.G. Steiner, B.C. Neumann & J.I. Schiller, "Kerberos:An Authentication Service for Open Network Systems",Usenix Conference Proceedings, pp 191-202, Feb. 1988.

[2] S. M. Bellovin and M. Merritt. Limitations of the Kerberos Authentication System. In Proceedings of the Winter 1991 Usenix Conference. January 1991.

[3] J. Kohl, B.C. Neumann & T.Y. Ts'o, "The Evolution of the Kerberos Authentication Service", In Distributed Open Systems.IEEE Computer Society Press, pages 78-94,1994.

[4] NIST FIPS PUB 180. Secure Hash Standard. NIST, May 1993.

[5].M. Mambo and E. Okamoto. Proxy cryptosystems: Delegation of the power to decrypt ciphertexts. In IEICE Trans. Fund. Electronics Communications and Comp. ci. E80-A/1, pages 54-63, 1997.

[6] Anca Ivan, Yevgeniy Dodis, Proxy Cryptography Revisited, Network and Distributed System Security Symposium (NDSS), February, 2003

[7] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In Proc. Of Crypto'84, pp. 10-18, 1984.

[8] R. Housley, W. Ford, W. Polk, and D. Solo.Internet X.509 Public Key Infrastructure:Certificate and CRL Profile. RFC 2459,http://www.ietf.org/rfc/rfc2459.txt, January 1999.