Four-Party Encryption Key Exchange

Bing-Chang Chen, Hung-Min Sun and Tzonelih Hwang
Department of Computer Science and Information Engineering
National Cheng-Kung University
Tainan, Taiwan 70101
E-mail: hmsun@mail.ncku.edu.tw
Phone: +886-6-2757575 ext. 62535
Fax: +886-6-2747076

Abstract

Key exchange protocol is important for sending secret messages using the session key between two parties. In order to reach the objective, the premise is to generate a session key securely. Encryption key exchange was first proposed to generate a session key with a weak authenticated password against guessing attacks. Next, another authenticated key exchange protocols for three-party, two clients who request the session key and one server who authenticates the user's identity and assists in generating a session key, were proposed. Once the clients are under diverse circumstances, they have to be authenticate by their own server. That is, it needs four-party to complete this protocol. In this paper, we focus on the four-party authenticated key exchange protocol, and presents four-party EKE.

1 Introduction

Encryption key exchange(EKE) using Diffie-Hellman [3] key exchange is a scheme proposed by Bellovin and Merritt [1] which enables to generate a session key securely between two authenticated parties. In addition to key exchange, both parties have to authenticate the other party's identity and make sure he communicates with the right person. In general, people are used to choosing an easy-to-remember password for authentication. But that will cause the system intruded by an attacker using guessing attacks. In order to prevent this kind of attacks, the scheme, EKE which allowed users to choose a weak password to authenticate, was presented to solve this problem.

In addition to two-party authenticated key exchange [2, 6, 10, 13, 15], three-party authenticated key exchange scheme is also presented [5]. It includes three roles, two clients and one server respectively. Its main goal is to make two clients possess a session key through the server. The clients must be recognized by the server. The server manages all the clients, and helps them to generate a session key. All of the clients trust the server.

Moreover, authenticated key exchange schemes are divided into two classes according the modes of generating session key. One is the scheme of key transfer, the other is the scheme of key agreement. The session key of former is chosen and transferred by the host, while the other is agreed by the host and user after authenticating the user. Theoretically, key agreement protocol is fairer and more secure than key agreement protocol because the session key of the former is computed by the owners of key. Nevertheless, the session key of key transfer protocol is chosen by one party and directly distributes and communicates by encrypted form. Once the interceptor gets the communicated information, she has the opportunity to obtain the session key. However, the attacker has no information about the session key generated by key agreement protocol using Diffie-Hellman key exchange scheme.

Key agreement method is fair to two-party authenticated key exchange protocols while it is also secure to three-party protocols. For original three-party schemes [4, 5, 8, 9, 11, 12], the server transfers the same session key to two requested clients. Once the key is used to communicate between these two users, the server can wiretap the messages using the stored session key if the server is untrusted. Even if the chosen session key is discovered or leaked, the security crisis always exists when proceeding communication. Therefore, the key agreement method can clear up this problem. The session key is decided by the two clients rather the server. No one can be aware of this session key but the persons who generated the key.

In three-party EKE, the clients are managed and authenticated by the same server. Once the clients come from different circumstances, they need to be govern and certified by diverse servers. The users only trust the server under their own domain. Therefore, four-party EKE which has four roles, two servers and two clients, is presented to solve this problem. Similarly, four-party EKE like three-party EKE has the same key generating modes, key transfer and key agreement. In this paper, we propose a four-party EKE protocol with key agreement which makes the protocol fairer. In Section 2, we describe more detailed about four-party EKE and also its notation used in this protocol which presents in Section 3. In addition, the security analysis of this protocol is represented in Section 4.

2 Description of Four-party EKE

In the past, encrypted key exchange(EKE) were divided into two types, two-party and three-party. EKE is a key exchange protocol which the participators present their passwords for authentication and then cooperate to generate the agreed session key. Two-party EKE is the original scheme proposed for one client and one server generating a session key. The main security goal is to prevent the guessing attacks while the two parties use their shared password to request and generate a session key. Two-party EKE may be extended to three-party EKE[14]. The members of three-party EKE consists of one server and two clients. The objective is to generate a session key between two clients and also authenticate the two clients by the server. This kind of protocol must ensure the two clients may be authenticated by the server or the protocol will be failed.

Due to the clients may be under the different domains or circumstances, they have their own trusted server under this situation. Therefore three-party EKE which only one server handles the whole protocol is not suited for use in this environment. The assignments of the servers are to authenticate the clients and also assist in generating the session key between the clients. Once the clients are under diverse circumstances, the servers should authenticate the clients which are under their domination. Four-party EKE is the only way to achieve the objective. Next, we will propose the four-party EKE protocol. Some notations used in this protocol are listed in Table 1 completely.

In Section 1, we refer to the modes of generating session key including key transfer and key agreement. The key transfer system is impracticable to content with four-party protocol. The clients are under their own infrastructure and only trust their own server. Therefore the party of choosing and distributing key is hard to decide. If one of the servers chooses the key, the client without her jurisdiction doesn't need to trust her, and vice versa. Hence key agreement system is the best way for the four party. The clients trust the session key their server sent, and key agreement tasks are progressed between servers.

Table 1. Protocol notations

A, B: Honest entities.
SA, SB: The servers of A and B respectively.
ka, kb: The passwords of A and B.
(x_{SA}, y_{SA}): The private and public key pair of SA.
(x_{SB}, y_{SB}): The private and public key pair of SB.
na, nb: The nonce keys of A and B.
K: The session key of A and B.
$\{m\}_k$: A message m encrypted by the asymmetric key k.
$[m]_k$: A message m encrypted by the symmetric key k.
c_{SA}, c_{SB}: Challenges of SA and SB.
x,y: Random numbers.
g: Generator

3 Proposed Four-party EKE

In this section, a new EKE scheme is proposed for supporting four parties to generate a session key. The four parties consist of two clients requested the session key and two server whose duties are to authenticate the clients under their jurisdiction and give assistance to generate the session key. Besides, the server of the requested client only communicates with the opposite server, and doesn't know the client under the opposite server, and vice versa.

Our idea of this kind of protocol is divided into two parts, one is the clients' tasks and the other is the servers' assignments. The clients only present their passwords to the servers for authentication and requesting the session key. The remaining works are implementing by the servers including authenticating the identities of the clients, generating the session key between the clients, explicit key confirmation, and transferring the session key securely to the clients at last.

The detailed messages communicated between the parties are listed in Table 2. To make the protocol clearly, we introduce the complete procedures as follows.

(1) Client A requests a session key with client B and transmits message 1 which includes the identities participated in this protocol except B, a nonce key na used to encrypt the session key later, and A's password ka for authentication 2 to B.

(2) Client B like A computes $\{B, SB, SA, nb, kb\}_{y_{SB}}$ and transmits message 2 to server SB.

(3) When receiving message 2, SB decrypts the message encrypted by the public key of SB. SB can authenticate B by checking the validity of kb. If it is true, she chooses a random number y to compute g^y and a challenge c_{SB}. Then SB sends message 3 to server SA.

(4) Message 3 consists two messages which all encrypted by the public key of SA, therefore SA can decrypt the messages. After decrypting the messages, SA can authenticate A by checking the validity of ka, and also get g^y. If the identity of A is true, SA chooses a random number x to compute g^x and a challenge c_{SA}. Then SA computes the session key $K = g^{xy}$ and transmits message 4 to SB.

(5) When message 4 is received, SB can get g^x and then compute $K = g^{xy}$. Therefore $[c_{SA}, c_{SB}]_K$ can be decrypted by K, then SB obtains c_{SA} and c_{SB}. In this step, SB has to send message 5 confirmed the session key and message 5' included the session key to SA and B individually.

(6) After message 5' is received, B can decrypt it using nb and get K.

(7) When SA receives message 5, the procedures of key confirmation are accomplished and also are proven the session key is true. Then SA uses the nonce key na to encrypt the session key K, and transmits message 6 to A. After receiving the message, A can decrypt it using na and get K.

Table 2. Four-party EKE

1. $A \rightarrow B : A, \{A, SA, SB, na, ka\}_{y_{SA}}$
2. $B \rightarrow SB :$
$\{A, SA, SB, na, ka\}_{y_{SA}}, \{B, SB, SA, nb, kb\}_{y_{SB}}$
3. $SB \rightarrow SA :$
$\{A, SA, SB, na, ka\}_{y_{SA}}, \{g^y, c_{SB}\}_{y_{SA}}$
4. $SA \rightarrow SB : \{g^x\}_{y_{SB}}, [c_{SA}, c_{SB}]_K$
5. $SB \rightarrow SA : c_{SA}$
5'. $SB \rightarrow B : [B, K]_{nb}$
6. $SA \rightarrow A : [A, K]_{na}$

4 Security Analysis

The main security property of EKE, two-party, three-party, or even four-party EKE, is to prevent guessing attacks by the intruders. Hence we will discuss whether the proposed protocol suffers from this attack. Because the passwords of client A and B are included in $\{A, SA, SB, na, ka\}_{y_{SA}}$ and $\{B, SB, SA, nb, kb\}_{y_{SB}}$, which are encrypted by the public keys of server SA and SB, no one can get ka or kb unless he got the private key to decrypt. Therefore the guessing attacks cannot succeed. If someone got the nonce keys na or nb, he can guess the passwords by exhaustive search from message 1 or 2. But the nonce key is a one-time encrypted key chosen at random, it is impossible to be gotten by the intruder.

Another attack is the replay attack which the attacker resends the former message and gets the session key without any meaningful data such as passwords. In this protocol, this attack is useless when an attacker replays message 1 or message 2. Because the attacker cannot decrypt message 5' or message 6 without nb or na no matter he disguises client B or A. In addition to replaying the clients' messages, it is also useless to play the role of servers by replaying their messages. The attacker's objective is to masquerade the servers to obtain their session key or passwords. If message 4, 5, 5' or 6 is replayed, the attacker will be rewarded for nothing and get no information. Because of replaying, the messages can not be changed and the attacker has no useful message from the unchanged message.

The securest scheme is to get up to perfect forward secrecy which is when the session key is leaked, the passwords cannot also be known. On the contrary, when the passwords are divulged, the session key is also secure. In this protocol, if the session key K was leaked, the attacker only got the challenges c_{SA} and c_{SB} from message 4. The passwords are embeded in the encryption, no one can decrypt the message unless he has the corresponding private key. Moreover, the challenges c_{SA} and c_{SB} have no directly relation with the encryption messages of message 1 and 2, it is impossible for the attacker to guess the password as a result of revealing the session key. Similarly, having the passwords cannot help the attacker to get the session key.

One of the requirements of this protocol is to prevent that the servers have knowledge about whom the client under her government communicated with. That is, SA doesn't know who B is and also SB doesn't know who A is. The purpose of this requirement is to avoid the server's eavesdropping between the clients. The server only knows the opposite server in this protocol.

5 Conclusions

In this paper, we proposed a new kind of EKE protocol, four-party EKE, which allows two clients under different severs to generate a session key and also authenti-

cate their identities. Unlike three-party EKE, it needs two servers to support the clients creating the session key. Also, we present the security analysis of this protocol such as preventing guessing attack, replaying attack, and achieving perfect forward secrecy.

References

[1] S.M. Bellovin and M. Merritt, "Encrypted key exchange: Password-based protocols secure against dictionary attacks," In *Proceedings of the 1992 IEEE Computer Society Conference on Research in Security and Privacy*,1992, pp. 72-84.

[2] S.M. Bellovin and M. Merritt, "Augmented encrypted key exchange: A password-based protocol secure against dictionary attacks and password file compromise," Technical report, AT&T Bell Laboratories,1994.

[3] W. Diffie and M. Hellman, "New directions in cryptography," *IEEEE Transactions on Information Theory*, NOv. 1976.

[4] L. Gong, "Optimal authentication protocols resistant to password guessing attacks," In *Proceedings of the 8th IEEE Computer Security Foundation Workshop*, County Kerry, Ireland, June 1995, pp. 24-29.

[5] L. Gong, M.A. Lomas, R. Needham, and J. Saltzer, "Protecting poorly chosen secrets from guessing attacks," *IEEE Journal on Selected Areas in Communications*, Vol. 11, No. 5, June 1993, pp. 648-656.

[6] D. Jablon, "Strong password-only authenticated key exchange," *Computer Communication Review*, Vol. 26, No. 5, Oct. 1996, pp.5-26.

[7] D. Jablon, "Extended password methods immune to dictionary attack," In *WETICE '97 Enterprise Security Workshop*, Cambridge, MA, June 1997.

[8] S. Keung and K. Siu, "Efficient protocols secure against guessing and replay attacks," *Proceedings of the 4th International Conference on Computer Communications and Networks*, 1995, pp. 105-112.

[9] T. Kwon and J. Song, "Efficient key exchange and authentication protocols protecting weak secrets," *IEICE Trans. Fundamentals*, Vol.E81-A, No. 1, Jan. 1998, pp. 156-163.

[10] T. Kwon and J. Song, "Secure agreement scheme for g^{xy} via password authentication," *Electronics Letters*, Vol. 35, No. 11, May 1999, pp. 892-893.

[11] T. Kwon, M. Kang, S. Jung and J. Song, "An Improvement of the password-based authentication protocol(K1P) on security against replay attacks," *IEICE Trans. Communications*, Vol.E82-B, No. 7, July 1999, pp. 991-997.

[12] T. Kwon, M. Kang, and J. Song, "An Adaptable and reliable authentication protocol for communication networks," *Proc. IEEE INFOCOM 97*, Kobe, Japan, 1997, pp. 738-745.

[13] S. Lucks, "Open key exchange: How to defeat dictionary attacks without encrypting public keys," *Proceedings of the Security Protocol Workshop '97*, Springer-Verlag, April 1997.

[14] M. Steiner, G. Tsudik and M. Waidner, "Refinement and extension of encrypted key exchange," *Operating Systems Review*, Vol. 29, Iss. 3, July 1995, pp. 22-30.

[15] T. Wu, "The secure remote password protocol," *Internet Society Symposium on Network and Distributed System Security*, 1998.