Constructing a Conversation Key in Three-Party Communications Environment

Chin-Chen Chang,[1, 2] Yu-Wei Su[2] and Chia-Yin Lee[2]
[1]*Department of Information Engineering and Computer Science*
Feng Chia University, Taichung 407, Taiwan
[2]*Department of Computer Science and Information Engineering*
National Chung Cheng University, Chiayi 621, Taiwan

Abstract

This article proposes an efficient, less communication rounds, three-party encrypted key exchange protocol to achieve the authentication requirement. The protocol is provided with (1) no asymmetric encryption algorithm which is adopted to reduce the costs (such as any public-key infrastructure); (2) using pre-shared key to prevent adversaries that masquerade as legal users after guessing attacks; (3) avoiding the variant man-in-the-middle attacks on Diffie-Hellman based protocols; (4) achieving mutual authentication. With these four features, the proposed protocol is suitable for being applied for establishing secure channels between two clients, which are supported with the same trusted server.

.

Keywords: *Three-party encrypted key exchange, secure channels, guessing attacks.*

1. Introduction

In 1976, Diffie and Hellman [1] proposed a key exchange protocol, in which two parties can establish a common secret key. However, no authentication procedure was mounted on the exchanged message, Diffie-Hellman protocol suffered from man-in-the-middle attacks. An adversary can interrupt the transmitted messages between two parties in Diffie-Hellman protocol, and amends messages to further establish common secret key with the two parties respectively. Consequently, Steiner et al. [2] proposed the first three-party encrypted key exchange protocol (3PEKE) to extend and amend Diffie-Hellman protocol in 1995. In Steiner et al.'s protocol (STW-3PEKE), two clients individually pre-shared the secret (e.g. their passwords) with a trusted server. The relation chart between clients and the trusted server is shown as follows.

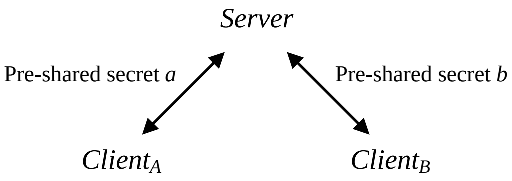

Figure 1. Relation of client and trusted server

These two clients authenticate each other and commonly establish a session key through the server. Unfortunately, STW-3PEKE protocol is vulnerable to undetectable on-line and off-line password guessing attacks [3]. Therefore, many 3PEKE protocols [3, 4, 5, 6] were proposed in the past decade to avoid password guessing attacks. Furthermore, those studies also mentioned different requirements, such as reducing communication rounds or computation costs.

Lin et al. [3] modified STW-3PEKE, named LSH-3PEKE, by employing the public-key cryptosystem to avoid the password guessing attack. Next, Lin et al. [4] further amended the former protocol to overcome the drawback which should encounter establishing certificate infrastructure (LSSH-3PEKE). Subsequently, Lee et al. [5] proposed an improved LSSH-3PEKE which focuses on reducing the number of transmission rounds. Recently, Sun et al. [6] proposed two improved 3PEKE protocols (SCH-3PEKE), respectively based on the password and the verifier. Compared to other protocols, SCH-3PEKE protocol is the most efficient solution in communication cost, where SCH-3PEKE only needs five transmission rounds to finish information delivery. Although SCH-3PEKE protocol resists against password guessing attacks and stolen-verifier attacks, it suffers instead from an easier attack and variant man-in-the-middle attacks. The clients are fooled into believing an adversary is a legal client and also pre-shares a secret with the same trusted server.

Summarizing the previous studies, it concludes some essential characteristics in 3PEKE protocol: (1) resisting against the guessing attack; (2) preventing the variant man-in-the-middle attack; (3) avoiding adopting public key infrastructure; (4) achieving mutual authentication. This article focuses on the above mentioned four essentials to propose a formal 3PEKE protocol. From our analyses, the proposed protocol satisfies the mentioned essentials with few transmission rounds and low computation costs. The proposed protocol not only achieves the scenario mentioned in [2] but also amends the variant man-in-the-middle attack in [6]. Moreover, the proposed protocol does not use any public-key cryptosystem in the server side to enhance the practicality.

In adopting self-encryption [7], our protocol provides the characteristic to authenticate two parties in one delivering round. This characteristic assists by overcoming the authentication problem with a few transmission rounds.

This paper is organized as follows. In Section 2, we review the SCH-3PEKE protocol and describe the flaws, the variant man-in-the-middle attack, of their protocol. Next, we propose an improved protocol in Section 3, and then discuss some relative analyses in Section 4. Finally, some conclusions are given in Section 5.

2. The Review of Sun et al.'s Password Based Protocol

SCH-3PEKE protocol includes two versions, the password and verifier based. Firstly, we introduce the notations used throughout this section, and then describe the password based version in the subsection.

2.1 Notations

p: a large prime number
g: a primitive element $\in GF(P)$
A, B: the two clients
S: the trusted server
P_A, P_B: two secret passwords, which A and B individually pre-share with S
N_A, N_B, N_S: three random numbers generated by A, B, and S, respectively
$E_K(.)$: a symmetric key encryption algorithm with the encryption key K
PK (.): an asymmetric key encryption algorithm with the public key PK
a, b: two random numbers generated by S

2.2 The Steps of Sun et al.'s Protocol

Suppose that *A* and *B* have pre-shared P_A and P_B with *S*. Assume that *A* wants to establish a session key with *B*. The detailed steps are shown as follows.

Step 1. *A* randomly chooses N_A, calculates $C_A = PK(g^{N_A},\ P_A)$ and sends C_A to *B*.

Step 2. *B* randomly chooses N_B, calculates $C_B = PK(g^{N_B},\ P_B)$ and sends C_A, C_B to *S*.

Step 3. After receiving C_A and C_B, *S* verifies the validity of C_A and C_B by decrypting C_A and C_B, and then verifying P_A and P_B. If the verification is correct, *S* randomly chooses *a*, *b*, and N_S; and calculates $R_A = (g^{N_B},\ g^b)^{N_S}$ and $R_B = (g^{N_A},\ g^a)^{N_S}$. Finally, *S* sends R_A, *a*, R_B, and *b* to *B*.

Step 4. After receiving the computed results from *S*, *B* can calculate a session key $K = R_B^{(N_B+b)}$, so that, $K = g^{(N_S)(N_A+a)(N_B+b)}$. Then, *B* computes $C_{BA} = E_K(C_A)$, and sends R_A, *a*, and C_{BA} to *A*.

Step 5. Simultaneously, after receiving the computed results from *S*, *A* is able to calculate $K = R_A^{(N_A+a)}$. Next, *A* verifies the validity of a session key *K* by decrypting $E_K(C_A)$. If the verified result is correct, then *A* sends $C_{AB} = E_K(C_{BA})$ to *B* for notifying that *A* has calculated *K* successfully.

After mounting the previously mentioned steps, *A* and *B* are able to coordinate a valid session key *K* with assistance from *S*. However, this protocol probably suffers from the variant man-in-the-middle attack.

2.3 The Variant Man-in-the-middle Attack

Nam et al. [8] pointed out that SCH-3PEKE protocol suffers from the variant man-in-the-middle attack. Suppose that an adversary *M* who also pre-shares the password P_M with *S*. *M* can wiretap the connection between *A* and *B* to read and interrupt the messages, which are transmitted to coordinate the session key. Nam et al. showed that *M* can masquerade as *A* to share a session key with *B*, and can also masquerade as *B* to share another session key with *A*, simultaneously by interrupting and counterfeiting the transmitted messages. The attack scenario is shown as follows.

Step 1. When *A* initially sends C_A to *B*, and then *B* sends C_A and C_B to *S*, and *M* interrupts C_A and C_B.

Step 2. *M* generates two random numbers N_{M1} and N_{M2}, and calculates $C_{M1} = PK(g^{N_{M1}},\ P_M)$; $C_{M2} = PK(g^{N_{M2}},\ P_M)$. Then *M* sends {$C_A$, C_{M1}} and {C_B, C_{M2}} to *S*.

Step 3. After receiving {C_A, C_{M1}} and {C_B, C_{M2}}, *S* assumes that two different session key establishments are demanded, respectively from *A* to *M*, and from *B* to *M*.

Step 4. *S* generates *a*, *b*, *m*1, *m*2, N_{S1}, and N_{S2} and then calculates

$$R_A = (g^{N_{M1}} \cdot g^{m1})^{N_{S1}},\ R_{M1} = (g^{N_A} \cdot g^{a})^{N_{S1}},$$

$$R_B = (g^{N_{M1}} \cdot g^{m1})^{N_{S2}},\ R_{M2} = (g^{N_B} \cdot g^{b})^{N_{S2}}.$$

Then, *S* sends {R_A, *a*, R_{M1}, *m*1} and {R_B, *b*, R_{M2}, *m*2} to *M*.

Step 5. After acquiring {R_A, *a*, R_{M1}, *m*1} and {R_B, *b*, R_{M2}, *m*2}, *M* can calculate:

$$K_{AM} = R_{M1}^{(N_{M1}+m1)} = g^{(N_{S1})(N_{M1}+m1)(N_A+a)}$$ and

$$K_{BM} = R_{M2}^{(N_{M2}+m2)} = g^{(N_{S2})(N_{M2}+m2)(N_B+b)}.$$

Step 6. *M* sends {R_{M2}, *m*2, R_B, *b*} to *B*.

Step 7. *B* calculates $K_{BM}=R_B^{(N_B+b)}=g^{(N_S)(N_{M2}+m2)(N_B+b)}$ and $C_{BA}=E_{K_{BM}}(C_A)$, and then sends {R_{M2}, *m*2, C_{BA}} to *A*. Simultaneously, *M* interrupts this message {R_{M2}, *m*2, C_{BA}} to avoid delivering it to *A*.

Step 8. *M* calculates $K_{AM}=R_A^{(N_A+a)}=g^{(N_S)(N_{M1}+m1)(N_A+a)}$ and $C_{MA}=E_{K_{AM}}(C_A)$, and sends the message {R_A, *a*, C_{MA}} to *A*.

Step 9. *A* calculates $K_{AM}=R_A^{(N_A+a)}=g^{(N_S)(N_{M1}+m1)(N_A+a)}$ and $C_{AB}=E_{K_{AM}}(C_{MA})$, and then sends C_{AB} to *B*. Simultaneously, *M* interrupts C_{AB} to avoid delivering it to *B*.

Step 10. *M* sends $C_{MB}=E_{K_{BM}}(C_{BA})$ to *B*.

After conducting the previously mentioned steps, *M* shares session keys K_{AM} and K_{BM} with *A* and *B*, respectively; neither *A* nor *B* can aware that the session keys are established with *M* individually.

3. The Proposed Scheme

According to the flaw of SCH-3PEKE, and summarizing previous relative studies, herein, we propose a 3PEKE protocol with the following main properties:

1. Using no asymmetric encryption algorithm (such as Public Key Cryptosystem) to reduce extra costs.
2. Using length sufficient pre-shared keys to prevent adversaries masquerading legal clients after guessing attacks.
3. Avoiding the variant man-in-the-middle attack, such as appearing in SCH-3PEKE protocol, on the Diffie-Hellman based protocols.
4. Achieving the mutual authentication.

We subsequently define used notations in Section 3.1, and next present the protocol details in Section 3.2.

3.1 Notations

p: a large prime number
g: a primitive element in *GF*(*p*)
A, *B*: the two clients
S: the trusted server
ID_A, ID_B: two identities of *A* and *B*
K_A, K_B: two secret keys, which *A* and *B* individually pre-share with *S*
N_A: a random nonce generated by *A*
E_K(.): a symmetric encryption algorithm with the encryption key *K*
a, *b*, *s*: three random numbers generated by *A*, *B*, and *S*, respectively

3.2 The Steps of Proposed Protocol

Assume that *A* and *B* pre-share K_A and K_B with *S* respectively, and *A* wants to corporately agree a session key *K* with *B*. The communication flowchart is shown in Figure 2. The proposed protocol is given as follows.

Step 1. *A* generates *a*, and sends $\{ID_A,\ ID_B,\ E_{K_A}(g^a \bmod p,\ K_A,\ N_A)\}$ to *S*.

Step 2. After receiving $E_{K_A}(g^a \bmod p,\ K_A,\ N_A)$, *S* decrypts it and verifies whether the K_A is equal to the pre-shared encryption key K_A. If the verification holds, S generates s and an encryption key K'_B and sends $\{ID_A,\ E_{K_B}(g^{as} \bmod p,\ K_B,\ K'_B)\}$ to B.

Step 3. To verify the validity of $\{ID_A,\ E_{K_B}(g^{as} \bmod p,\ K_B,\ K'_B)\}$, B uses K_B to decrypt the message and verifies whether it is equal to the pre-shared encryption key K_B. If the verification holds, B generates b and sends $\{E_{K_B}(g^{b} \bmod p,\ K'_B)\}$ to S. B can calculate the session key $K = g^{abs} \bmod p$. Finally, B updates K'_B to be the new pre-shared encryption key.

Step 4. After receiving $\{E_{K_B}(g^{b} \bmod p,\ K'_B)\}$, S decrypts it and verifies whether the K'_B is equal to the new pre-shared encryption key K'_B. If the verification holds, S generates an encryption key K'_A . Then sends $\{ID_B,\ E_{K_A}(g^{bs} \bmod p,\ N_A,\ K'_A)\}$ to A, and updates the encryption key K'_B pre-shares with B.

Step 5. A verifies the validity of $E_{K_A}(g^{bs} \bmod p,\ N_A,\ K'_A)$ after decrypting, and checking N_A, originally sent from A in Step 1. If the verification holds, A can calculate the session key $K = g^{abs} \bmod p$. Then, A updates K'_A to be the new pre-shared encryption key.

Step 6. Finally, A sends $\{E_{K_A}(K'_A)\}$ back to S to inform that A finishes the protocol, and sends $\{ID_A$, $E_K(K)\}$ to B simultaneously to inform that A definitely, successfully calculated the session key K.

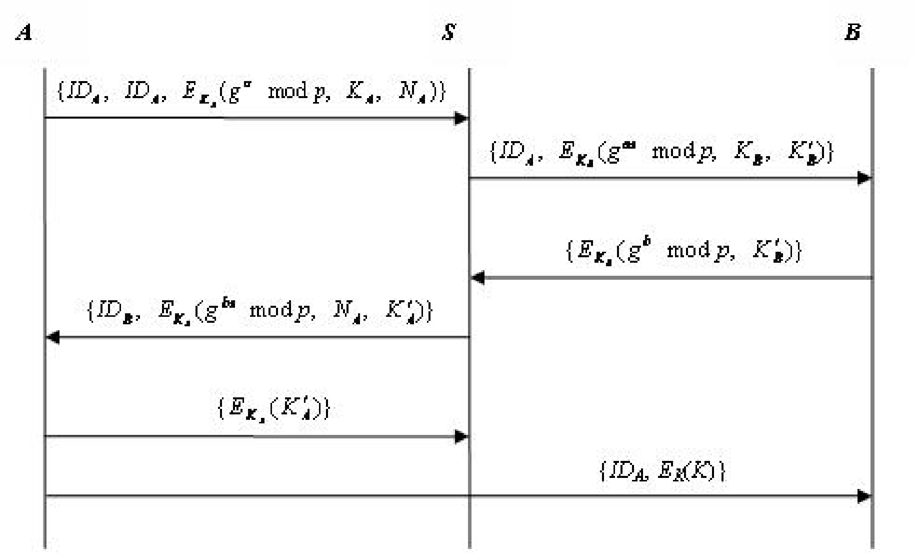
A
S
B
{ID_A, ID_A, E_K_A(g^a mod p, K_A, N_A)}
{ID_A, E_K_B(g^as mod p, K_B, K'_B)}
{E_K_B(g^b mod p, K'_B)}
{ID_B, E_K_A(g^bs mod p, N_A, K'_A)}
{E_K_A(K'_A)}
{ID_A, E_K(K)}

Figure 2. The data transmission of protocol.

Table 1. Comparisons of our scheme and others

Protocols	Our Protocol	LSH-3PEKE	LSSH-3PEKE	LHL-3PEKE	SCH-3PEKE
(1)	Yes	No	Yes	Yes	No
(2)	Yes	Yes	Yes	Yes	Yes
(3)	Yes	Yes	Yes	Yes	No
(4)	Yes	Yes	Yes	Yes	Yes
(5)	5	5	7	6	5

(1) Using no asymmetric encryption algorithm to reduce the costs
(2) Preventing the guessing attack on pre-shared secrets against adversaries masquerading as legal clients
(3) Avoiding the variant man-in-the-middle attack on Diffie-Hellman based protocols
(4) Achieving the mutual authentication
(5) Transmission rounds

5. Analyses and Comparisons

As mentioned in Section 1, 3PEKE protocols probably suffer from password guessing attacks when an adversary can discover the secrets (such as passwords, or pre-shared keys) by analyzing transmitted messages. Guessing attacks are categorized into two types: the off-line and on-line guessing attacks. The off-line guessing attack occurred when an adversary holds the specimens which involve the information related secrets. The adversary can perform dictionary attacks on weak passwords. With a high probability, the adversary can resolve the password possessing sufficient specimens. Another manner of off-line guessing attack is the brute-force guessing attack, but such an attack is generally successful to low entropy secrets. It is difficult to resolve high entropy secrets only by the brute-force guessing attack. The off-line guessing attack conducted in our protocol is reduced to low damage because the entropy of the pre-shared secrets depends on the key length of the adopted symmetric encryption algorithm. In a practical employment, the key length is more than 128-bit (e.g. AES-128, AES-192, and AES-256). In addition, changing shared key further repairs the off-line guessing attack drawback. Even if the adversary successfully guesses the pre-shared key K_A or K_B, they are both one-time secrets and expire in the next session. K_A and K_B are replaced by K'_A and K'_B after accomplishing the session key establishment.

The authentication from A to S is done by self-encryption on K_A. After Step 1, $E_{KA}(g^a \bmod p,\ K_A,\ N_A)$ is valid only in which K_A is correct; in other words, the adversary cannot masquerade as A to perform session key establishment without holding K_A. On the other hand, the authentication from S to B is similarly done by self-encryption on K_B. B verifies $\{ID_A,\ E_{K_B}(g^{as} \bmod p,\ K_B,\ K'_B)\}$ and identifies S both by self-encryption on K_B. After Step 3, A can identify S by confirming the generated nonce N_A on $E_{K_A}(g^{bs} \bmod p,\ N_A,\ K'_A)$ delivered from S, achieves the authentication from S to A. In the other side, S identifies B by checking updated shared key K'_B on $E_{K_B}(g^b \bmod p,\ K'_B)$ transmitted from B. As in the previously mentioned depiction, the four authentication requirements guarantee the availability of $(g^a \bmod p)$ and $(g^b \bmod p)$. So, ensuring the authentication between A and S, and between S and B avoids the variant man-in-the-middle attack which appears in Diffie-Hellman based protocols, such as appearing in SCH-3PEKE protocol. Furthermore, both A and B are identified through S, and the validity of messages $\{ID_A, E_K(K)\}$ delivered in Step 5, also by self-encryption. Thus, A and B can confirm the identity of each other. Table 1 shows the four essential requirements compared with other protocols.

5. Conclusions

This article proposes a practical key exchange protocol for three parties. In Section 4, we show that our protocol has the fewest transmission rounds and satisfies the four essentials mentioned. Employing self-encryption mechanism achieves authentication challenge; thus the validity of messages and identities can be ensured with assistance from the trusted server. Furthermore, self-encryption property also assists in avoiding utilizing public-key cryptosystem, eliminating the difficulty to implement infrastructures. Finally, self-encryption can be adopted to prevent the variant man-in-the-middle attacks on Diffie-Hellman based protocols.

Key exchange is a well-known security issue. Those two parties commonly establish a session key before secure communication, which is likewise in wild applications. As our proven characteristics, our result is suitable for being applied on some network environments which are provided with centralized servers that pre-share secret keys respectively with all the participants.

6. References

[1] Diffie, W. and Hellman, M. E., “New Directions in Cryptography,” *IEEE Transactions on Information Theory*, Vol. IT-22, No. 6, (1976) 644-654.

[2] Steiners, M., Tsudik, G. and Waidner, M., “Refinement and Extension of Encrypted Key Exchange,” *ACM Operating Systems Review*, Vol. 29, No. 3, (1995) 22-30.

[3] Lin, C.L., Sun, H.M. and Hwang, T., “Three-party Encrypted Key Exchange: Attacks and a Solution,” *ACM Operating Systems Review*, Vol. 34, No. 4, (2000) 12-20.

[4] Lin, C.L., Sun, H.M., Steiner, M. and Hwang T., “Three-party Encrypted Key Exchange without Server Public-keys,” *IEEE Communications Letters*, Vol. 5, No. 12, (2001) 497-499.

[5] Lee, T.F., Hwang, T., and Lin, C.L., “Enhanced Three-party Encrypted Key Exchange without Server Public Keys,” *Computers & Security*, Vol. 23, No. 7, (2004) 571-577.

[6] Sun, H.M., Chen, B.C. and Hwang, T., “Secure Key Agreement Protocols for Three-party against Guessing Attacks,” *The Journal of Systems and Software*, Vol. 75, (2005) 63-68.

[7] Hwang, K.F. and Chang, C.C., “A Self-encryption Mechanism for Authentication of Roaming and Teleconference Services,” *IEEE Transactions on Wireless Communications*, Vol. 2, No. 2, (2003) 400-407.

[8] Nam, J., Kim, S. and Won, D., “Attack on the Sun-Chen-Hwang’s Three-Party Key Agreement Protocols Using Passwords,” *IEICE Transactions on Fundamentals*, Vol. E89-A, No. 1, (2006) 209-212.