Comments on Yoon and Yoo's Three-party Encrypted Key Exchange Protocol

Ya-Fen Chang[1], Wei-Cheng Shiao[2], and Chung-Yi Lin[2]
[1]Department of Computer Science and Information Engineering,,
National Taichung Institute of Technology, Taichung 404, Taiwan, R.O.C.
E-mail: cyf@cs.ccu.edu.tw
[2]Graduate School of Computer Science and Information Technology,
National Taichung Institute of Technology, Taichung 404, Taiwan, R.O.C.
E-mail: acooljoe101@xuite.net ; daniel0326@xuite.net

Abstract— In 2008, Yoon and Yoo proposed an improved three-party encrypted key exchange protocol to enhance Chang and Chang's scheme suffering from undetectable on-line password guessing attacks. They claimed that their fixed protocol is more secure than Chang and Chang's. Unfortunately, we find their protocol still suffers from undetectable on-line password guessing attacks. In this article, we will indicate why Yoon and Yoo's protocol is still insecure.

***Keywords*— Cryptography, key exchange, authentication, three-party, password guessing attacks**

1. Introduction

In 1976, Diffie and Hellman proposed a key exchange protocol such that two parties can securely communicate with a common secret key [1]. Because no authentication procedure is coupled with the exchanged message, Diffie-Hellman key exchange protocol cannot defend against man-in-the-middle attacks. As a result, Bellovin and Merritt proposed the Encrypted Key Exchange (EKE) protocol [2] in which users are permitted using easy-to-remember passwords without dictionary attacks. A password is shared between two parties, and these two parties may use the shared password to negotiate a common session key. Thus, two parties can communicate with each other secretly.

In 1995, Steiner et al. proposed a three-party EKE protocol (STW-3PEKE) base on EKE protocols [3]. Each user shares an easy-to-remember password with a trusted third party, server, and each user can securely exchange their secret keys via the server. Server is a coordinator to help two users, who tend to communicate with each other, authenticated mutually. The server encrypts the messages from two communication parties and authenticates them by using easy-to-remember passwords. Because only legal users can decrypt messages from server, only they can obtain thecorrect session keys.

Because easy-to-remember passwords are involved in 3PEKE protocols, the security of passwords needs to be taken into consideration. Ding and Horster divide password-guessing attacks into three classes [4].

1) Detectable on-line password guessing attacks: An adversary can use a guessed password in an on-line transaction. The Adversary can verify the guessed password's correctness by using server's response. But the mounted attack would be detected by server with the failed logged procedure.

2) Undetectable on-line password guessing attacks: Similar to above attacks, an adversary tries to guess one user's password in an on-line transaction. However, a failed guessing procedure would not be detected by server. That is, server cannot distinguish an honest request from a malicious one.

3) Off-line password guessing attacks: An adversary guesses a password and verifies his guess off-line. No participation of server is required, so server will not notice the attack.

In 2000, Lin et al. showed that STW-3PEKE suffers not only undetectable on-line password guessing attacks but also off-line password guessing attacks. Thus, they proposed another 3PEKE protocol (LSH-3PEKE) [5], in which the trusted server holds a publicly-known server's public key to prevent both of the password guessing attacks. The approach of employing server's public key in 3PEKE is suitable when the number of messages exchanged is concerned.

Some improvements claim that server's public key should not be used since passwords are sufficient to make the exchanged messages secure. In 2004 Chang and Chang presented a 3PEKE protocol without server's public key [6]. However, Yoon and Yoo's pointed that Chang and Chang's 3PEKE protocol suffers from undetectable on-line guessing attacks [7]. And they also proposed a method to enhance Chang and Chang's 3PEKE protocol. Unfortunately, we find their protocol still suffers from undetectable on-line password guessing attacks. In this article, we will indicate why Yoon and Yoo's protocol is still insecure.

The rest of this paper is organized as follows. Section 2 reviews Yoon and Yoo's improved 3PEKE schemes. Section 3 shows the security flaws of Yoon and Yoo's protocol. At last, some conclusions are drawn in Section 4.

2. A review of Yoon and Yoo's three-party encrypted key exchange scheme

Yoon and Yoo proposed an improved 3PEKE scheme. They claimed the proposed protocol can defend against undetectable on-line password guessing attacks which Chang and Chang's suffers from. In this section, we first list the used notations in Subsection 2.1 and review Yoon and Yoo's 3PEKE in Subsection 2.2.

2.1. Notations

In this subsection, we show the notations used in the paper.

Alice/Bob	two users who want to communicate with each other
server	a trusted third party which Alice and Bob have registered at
ID_a , ID_b , ID_s	identities of Alice, Bob and server, respectively
P_a , P_b	passwords secretly shared by Alice and Bob with server, respectively
$E_p(\,)$	a symmetric encryption scheme with a password P.
n_a, n_b	random numbers chosen by Alice and Bob, respectively
p	a large prime number
g	a generator in GF(p)
T_a , T_b , T_s	random exponents chosen by Alice, Bob and a server, respectively
M_a , M_b	$M_a = g^{T_a} \bmod p$, $M_b = g^{T_b} \bmod p$
$H_s(\,)$	a one-way trapdoor function, where only server knows the trapdoor
$F_k(\,)$	a pseudo-random hash function indexed by a key k.
K_{as}, K_{bs}	a one-time strong keys shared by Alice and Bob with server, respectively

2.2. Reviews of Yoon and Yoo's 3PEKE Protocol

Yoon and Yoo's 3PEKE protocol has six steps. The details are as follows.

1) First, Alice chooses two random numbers n_a and T_a and computes $M_a = g^{T_a} \bmod p$ and $K_{as} = M_a^{n_a} \bmod p$. She takes her password P_a to encrypt M_a and computes $H_s(n_a)$ and $F_{K_{as}}(M_a)$. Then she transfers {ID_a, ID_b, ID_s, $E_{P_a}(M_a)$, $H_s(n_a)$, $F_{K_{as}}(M_a)$ } to Bob.

2) After getting them, Bob chooses two random numbers chooses two random numbers n_b and T_b, and computes $M_b = g^{T_b} \bmod p$ and $K_{bs} = M_b^{n_b} \bmod p$. He takes his password P_b to encrypt M_b and computes $H_s(n_b)$ and $F_{K_{bs}}(M_b)$. Then he transfers {ID_a, ID_b, ID_s, $E_{P_b}(M_b)$, $H_s(n_b)$, $F_{K_{bs}}(M_b)$, $E_{P_a}(M_a)$, , $H_s(n_a)$, $F_{K_{as}}(M_a)$ } to server.

3) Server uses P_a and P_b to decrypt $E_{P_b}(M_b)$ and $E_{P_a}(M_a)$ to get M_a and M_b. Then, it retrieves n_a and n_b from $H_s(n_a)$ and $H_s(n_b)$ by using trapdoor. Server computes $K_{as} = M_a^{n_a} \bmod p$ and $K_{bs} = M_b^{n_b} \bmod p$ to authenticate the received $F_{K_{as}}(M_a)$ and $F_{K_{bs}}(M_b)$. If both authentication messages are valid, server chooses a random number T_s computes $M_a^{T_s} \bmod p$ and $M_b^{T_s} \bmod p$, and uses n_a and n_b to compute $M_a^{T_s} \oplus n_b$ and $M_b^{T_s} \oplus n_a$. At last, server computes $F_{K_{as}}(ID_a, ID_b, K_{as}, M_b^{T_s})$ and $F_{K_{bs}}(ID_a, ID_b, K_{bs}, M_a^{T_s})$, and sends { $M_a^{T_s} \oplus n_b$, $F_{K_{bs}}(ID_a, ID_b, K_{bs}, M_a^{T_s})$, $M_b^{T_s} \oplus n_a$, $F_{K_{as}}(ID_a, ID_b, K_{as}, M_b^{T_s})$ } to Bob.

4) Bob uses n_b to compute $M_a^{T_s} \oplus n_b \oplus n_b = M_a^{T_s}$. Then he takes K_{bs} and $M_a^{T_s}$ to verify $F_{K_{bs}}(ID_a, ID_b, K_{bs}, M_a^{T_s})$. If it is valid, Bob computes the session key $SK = (M_a^{T_s})^{T_b} \bmod p = g^{T_a T_b T_s} \bmod p$ and $F_{SK}(ID_b, SK)$. Finally, Bob sends { $M_b^{T_s} \oplus n_a$, $F_{K_{as}}(ID_a, ID_b, K_{as}, M_b^{T_s})$, $F_{SK}(ID_b, SK)$ } to Alice.

5) She uses n_a to compute $M_b^{T_s} \oplus n_a \oplus n_a = M_b^{T_s}$. And then she takes K_{as} and $M_b^{T_s}$ to verify $F_{K_{as}}(ID_a, ID_b, K_{as}, M_b^{T_s})$. If it is legal, Alice computes the session key $SK = (M_b^{T_s})^{T_a} \bmod p = g^{T_a T_b T_s} \bmod p$. Alice computes $F_{SK}(ID_b, SK)$ and checks whether the computation result equals the received one. If it holds, Alice successfully authenticates Bob. Then Alice computes and sends $F_{SK}(ID_a, SK)$ to Bob.

6) Bob verifies $F_{SK}(ID_a, SK)$ to authenticate Alice. If it is legal, Bob will know the matter that Alice has the same session key.

3. The security flaw of Yoon and Yoo's scheme

In the section, we will demonstrate the security flaw of Yoon and Yoo's 3PEKE protocol by showing it cannot defend against undetectable on-line password guessing attacks. By this attack, a legal user Bob can guess Alice's password P_b without being noiced by server. The details are as follows.

1) Alice sents {ID_a, ID_b, ID_s, $E_{P_a}(M_a)$, $H_s(n_a)$, $F_{K_{as}}(M_a)$ } to Bob.

2) Bob save the messages which are sent by Alice.

3) Then, Bob guesses $P_a' = P_a$ from the password dictionary and uses P_a' to decrypt $E_{P_a}(M_a)$. Then Bob will get M_a'.

4) Next, Bob chooses a random number n_b, and computes $K_{bs} = (M_a')^{n_b} \bmod p$. He takes his password P_b to encrypt M_a' and computes $H_s(n_b)$ and $F_{K_{bs}} = (M_a')$.

5) Bob transfers {ID_a, ID_b, ID_s, $E_{P_b} = (M_a')$, $H_s(n_b)$, $F_{K_{bs}} = (M_a')$, $E_{P_a}(M_a)$, $H_s(n_a)$, $F_{K_{as}}(M_a)$ } to server.

6) After getting the messages sent form Bob, server authenticates Alice and Bob by verifying

$F_{K_{as}}(M_a)$ and $F_{K_{bs}} = (M_a')$. If they are legal, server chooses a random number T_s, computes $(M_a')^{T_s} \bmod p$ and $M_a{}^{T_s} \bmod p$, and uses n_a and n_b to compute $M_a{}^{T_s} \oplus n_b$ and $(M_a')^{T_s} \oplus n_a$ .Finally, server sends { $M_a{}^{T_s} \oplus n_b$, $F_{K_{bs}}(ID_a, ID_b, K_{bs}, M_a{}^{T_s})$, $(M_a')^{T_s} \oplus n_a$, $F_{K_{as}}(ID_a, ID_b, K_{as}, (M_a')^{T_s})$ } to Bob.

7) First of all, Bob uses n_b to compute $M_a{}^{T_s} \oplus n_b \oplus n_b = M_a{}^{T_s}$ and $n_a' = M_a{}^{T_s} \oplus (M_a')^{T_s} \oplus n_a$. The Bob takes n_a' to compute $K_{as}' = (M_a')^{n_a'} \bmod p$ and $F_{K_{as}'}(ID_a, ID_b, K_{as}, (M_a')^{T_s})$ .At last, Bob compares the computation result with $F_{K_{as}}(ID_a, ID_b, K_{as}, M_a{}^{T_s})$ sent by the server. If they are equal, Bob successfully get Alice's password. Otherwise, Bob repeats Steps 3 to 7 until matching. As a result, undetectable on-line guessing attacks can be easily mounted on Yoon and Yoo's protocol.

4. Conclusions

With deep insight into the security flaw shown in the previous section, we find that Yoon and Yoo's scheme cannot defend against undetectable on-line password guessing attacks. A legal user may keep another legal user's message in one session key negotiation iteration. Then he may use the kept messages to guess another legal user's password with server's aid without being noticed. On the other hand, Yoon and Yoo's protocol employs a trapdoor function. Actually, a trapdoor function can be regarded as a public-key encryption function. This property may violate the design principle of 3PEKE since PKI (public key infrastructure) is still needed.

REFERENCES

[1] W. Diffie and M. E. Hellman, "New directions in cryptography," IEEE Transactions on Information Theory, 22 (1976) 644-654.

[2] S. M. Bellovin and M. Merrit, "Encrypted key exchange: password-based protocols secure against dictionary attacks," in: IEEE Computer Society Symposium on Research in Security and Privacy, Oakland, California, 1992, pp. 72-84.

[3] M. Steiner, G. Tsudik and M. Waidner, "Refinement and extension of encrypted key exchange," ACM Operating Systems Review, 29 (1995) 22-30.

[4] Y. Ding and P. Horster, "Undetectable on-line password guessing attacks," ACM Operating Systems Review, 29 (1995) 77-86.

[5] C.-L. Lin, H.-M. Sun and T. Hwang, "Three-party encrypted key exchange: attacks and a solution," ACM Operating Systems Review, 34 (2000) 12-20.

[6] C.-C. Chang and Y.-F. Chang, "A novel three-party encrypted key exchange protocol," Computer Standards & Interfaces 26 (2004) 471-476.

[7] E.-J. Yoon and K.-Y. Yoo, "Improving the novel three-party encrypted key exchange protocol," Computer Standard & Interfaces 30 (2008) 309-314.