On the security of a password-only authenticated three-party key exchange protocol

Junghyun Nam[a,*], Kim-Kwang Raymond Choo[b], Juryon Paik[c], Dongho Won[c]

[a]*Department of Computer Science, Konkuk University, Republic of Korea*
[b]*Information Assurance Research Group, Advanced Computing Research Centre, University of South Australia, Australia*
[c]*Department of Computer Engineering, Sungkyunkwan University, Republic of Korea*

Abstract

This note reports major previously unpublished security vulnerabilities in the password-only authenticated three-party key exchange protocol due to Lee and Hwang (Information Sciences, 180, 1702–1714, 2010): (1) the Lee-Hwang protocol is susceptible to a man-in-the-middle attack and thus fails to achieve implicit key authentication; (2) the protocol cannot protect clients' passwords against an offline dictionary attack; and (3) the indistinguishability-based security of the protocol can be easily broken even in the presence of a passive adversary.

Keywords: Password-only authenticated key exchange (PAKE), Three-party key exchange, Man-in-the-middle (MITM) attack, Offline dictionary attack, Semantic security

1. Introduction

Password-only authenticated key exchange (PAKE) protocols enable two or more parties communicating over a public network to generate a high-entropy key (known as a *session key*) from their low-entropy passwords which are easy for humans to remember. PAKE protocols are often designed to work in the three-party setting, in which each party (commonly called a client) needs to remember only a single password shared with a trusted server. The

*Email: jhnam@kku.ac.kr

August 28, 2013

design of secure yet efficient three-party PAKE protocols is notoriously hard and continues to be a subject of active research. A key challenge in designing such protocols is to prevent potential attacks by a malicious client, who is registered with the server and thus is able to set up normal protocol sessions with other clients.

The brief contribution of this note is to present previously unpublished flaws in the S-EA-3PAKE protocol, a three-party PAKE protocol proposed by Lee and Hwang [4]. The design of the S-EA-3PAKE protocol is relatively simple and efficient, and carries a claimed proof of security in the ROR model due to Abdalla, Fouque and Pointcheval [1]. However, despite the claim of provable security, this protocol exhibits major security weaknesses. First, the protocol fails to achieve *implicit key authentication*, which is the fundamental security property that any given key exchange protocol is expected to provide. We demonstrate this by mounting a man-in-the-middle attack against the protocol. The attacker could be any malicious client. Second, the protocol is vulnerable to an offline dictionary attack by a malicious client and thus other clients cannot be guaranteed of the security of their passwords. Third, the protocol does not achieve semantic security of session keys; that is, session keys established by S-EA-3PAKE are distinguishable from random keys. We show this by mounting a passive attack in the ROR model, thereby invalidating the existing proof of security for S-EA-3PAKE.

2. The S-EA-3PAKE protocol

The S-EA-3PAKE protocol [4] is built upon Abdalla and Pointcheval's 2-party PAKE protocol called SPAKE [2]. Let A and B be two clients who wish to establish a session key, and pw_A and pw_B denote the passwords of A and B, respectively, shared with a trusted server S. The public parameters required by S-EA-3PAKE include: (1) a large prime p and a generator g of $\mathbb{Z}_p^*$, (2) two random elements M and N of $\mathbb{Z}_p^*$, (3) a cryptographic hash function H used as a key derivation function, and (4) a pair of message authentication code (MAC) generation/verification algorithms $(\mathsf{Mac}, \mathsf{Ver})$, where Ver outputs a bit, with 1 meaning `accept` and 0 meaning `reject`. S-EA-3PAKE proceeds as follows:

Step 1. A sends S and B a protocol initiation message $M_{init} = \langle A, B \rangle$ which states "A wants to establish a session key with B".

Table 1: Establishing the secret keys k_{AS} and k_{BS}

$pw_A \to k_{AS}$	$pw_B \to k_{BS}$
A chooses a random $x \in \mathbb{Z}_p^*$, computes $X = g^x$ and $X^* = X \cdot M^{pw_A}$, and sends X^* to S. At the same time, S chooses a random $u \in \mathbb{Z}_p^*$, computes $U = g^u$ and $U^* = U \cdot N^{pw_A}$, and sends U^* to A. Then, A and S set $k_{AS} = g^{xu}$.	B chooses a random $y \in \mathbb{Z}_p^*$, computes $Y = g^y$ and $Y^* = Y \cdot M^{pw_B}$, and sends Y^* to S. At the same time, S chooses a random $v \in \mathbb{Z}_p^*$, computes $V = g^v$ and $V^* = V \cdot N^{pw_B}$, and sends V^* to B. Then, B and S set $k_{BS} = g^{yv}$.

Step 2. A and S establish a shared secret key k_{AS} by running the 2-party protocol SPAKE. Likewise, B and S establish a shared secret key k_{BS}. More precisely, k_{AS} and k_{BS} are established as shown in Table 1.

Step 3. A (resp. B) computes the authenticator $\sigma_{AS} = \mathsf{Mac}_{k_{AS}}(A\|S)$ (resp. $\sigma_{BS} = \mathsf{Mac}_{k_{BS}}(B\|S)$) and sends it to S.

Step 4. S aborts if either $\mathsf{Ver}_{k_{AS}}(A\|S, \sigma_{AS}) = 1$ or $\mathsf{Ver}_{k_{BS}}(B\|S, \sigma_{BS}) = 1$ is untrue. Otherwise, S selects a random $s \in \mathbb{Z}_p^*$, computes

$$\begin{gathered} \overline{X} = X^s, \quad \overline{Y} = Y^s, \\ \overline{X}^* = \overline{Y} \cdot k_{AS}, \quad \overline{Y}^* = \overline{X} \cdot k_{BS}, \\ \sigma_{SA} = \mathsf{Mac}_{k_{AS}}(S\|A), \quad \sigma_{SB} = \mathsf{Mac}_{k_{BS}}(S\|B), \end{gathered}$$

and sends $\langle \overline{X}^*, \sigma_{SA} \rangle$ and $\langle \overline{Y}^*, \sigma_{SB} \rangle$ to A and B, respectively.

Step 5. A checks if $\mathsf{Ver}_{k_{AS}}(S\|A, \sigma_{SA}) = 1$, and aborts if the check fails. Otherwise, A computes the key derivation secret, $K_A = (\overline{X}^*/k_{AS})^x$, and the session key, $sk_A = H(A\|B\|K_A)$. Meanwhile, B checks if $\mathsf{Ver}_{k_{BS}}(S\|B, \sigma_{SB}) = 1$, and aborts if the check fails. Otherwise, B computes $K_B = (\overline{Y}^*/k_{BS})^y$ and $sk_B = H(A\|B\|K_B)$.

Step 6. A and B perform key confirmation by exchanging $\sigma_{AB} = \mathsf{Mac}_{sk_A}(A\|B)$ and $\sigma_{BA} = \mathsf{Mac}_{sk_B}(B\|A)$ and verifying them in a straightforward way.

The correctness of S-EA-3PAKE can be easily verified from $K_A = K_B = g^{xys}$.

3. Previously unpublished flaws

3.1. No implicit key authentication

Implicit key authentication of S-EA-3PAKE can be violated via a man-in-the-middle attack by a malicious (registered) client C. A possible attack scenario is as follows:

1. The attacker C blocks the protocol initiation message $M_{init} = \langle A, B \rangle$ from reaching S and instead, sends (to S) two forged initiation messages $M'_{init} = \langle A, C \rangle$ and $M''_{init} = \langle C, B \rangle$ which state, respectively, "A wants to establish a session key with C" and "C wants to establish a session key with B". As a result, S will think that there are two protocol sessions running concurrently; let $\Pi_{A,C}$ denote the session between A and C and $\Pi_{C,B}$ denote the session between C and B.
2. In both the sessions $\Pi_{A,C}$ and $\Pi_{C,B}$, C performs Steps 2 through 5 as per the protocol specification with its true identity. This can go undetected since none of the authenticators, σ_{AS} σ_{BS}, σ_{SA} and σ_{SB}, can confirm who the actual protocol participants are. As a result, C will share a session key, $sk_{A,C}$, with A and another session key, $sk_{C,B}$, with B.
3. With $sk_{A,C}$ and $sk_{C,B}$ in hand, C can perform Step 6 (of both sessions) in the straightforward way without being detected; C simply replaces $\sigma_{AB} = \mathsf{Mac}_{sk_{A,C}}(A\|B)$ and $\sigma_{BA} = \mathsf{Mac}_{sk_{C,B}}(B\|A)$, respectively, with $\sigma'_{AB} = \mathsf{Mac}_{sk_{C,B}}(A\|B)$ and $\sigma'_{BA} = \mathsf{Mac}_{sk_{A,C}}(B\|A)$.

At the end of the attack scenario, A and B believe that they have established a secure session with each other sharing a key, while in fact they have shared their keys with the attacker C. Consequently, S-EA-3PAKE fails to achieve implicit key authentication.

3.2. No password security

We now show that S-EA-3PAKE cannot protect clients' passwords against an offline dictionary attack. Assume a malicious client C who wants to find out the passwords of A and B. Let pw_C be the password of C. Then, an offline dictionary attack by C against both A and B can be mounted as follows:

Phase 1 (Gathering Password Verifiers Online). C conducts a type of man-in-the-middle attack to obtain information required to verify password guesses.

1. C blocks the initiation message $M_{init} = \langle A, B\rangle$ from reaching S and instead, sends two forged initiation messages $M'_{init} = \langle A, C\rangle$ and $M''_{init} = \langle C, B\rangle$, thereby deceiving S into thinking that there are two protocol sessions, $\Pi_{A,C}$ and $\Pi_{C,B}$, running concurrently.
2. C then performs Steps 2 through 5 of both sessions as specified by the protocol except for the following:
   - When A and B send $X^* = X \cdot M^{pw_A}$ and $Y^* = Y \cdot M^{pw_B}$ in Step 2, C makes a copy of these messages for later use.
   - C sends the same Step 2 message $Z^* = g^z \cdot M^{pw_C}$ of its own for both sessions, where $z \in_R \mathbb{Z}_p^*$.
   - When S sends $\langle \overline{X}^*, \sigma_{SA}\rangle$ and $\langle \overline{Y}^*, \sigma_{SB}\rangle$, respectively, to A and B in Step 4 of the sessions, C replaces $\overline{X}^*$ and $\overline{Y}^*$ with $\widehat{X}^* = \overline{X}^* \cdot g^z$ and $\widehat{Y}^* = \overline{Y}^* \cdot g^z$, respectively.

   Let $\langle \overline{Z}^*, \sigma_{SC}\rangle$ and $\langle \overline{Z'}^*, \sigma'_{SC}\rangle$ denote the two messages sent by S to C in Step 4 of $\Pi_{A,C}$ and $\Pi_{C,B}$, respectively.
3. Now when A and B exchange the key confirmation messages $\sigma_{AB} = \mathsf{Mac}_{sk_A}(A\|B)$ and $\sigma_{BA} = \mathsf{Mac}_{sk_B}(B\|A)$, C intercepts these messages and instead, sends the clients 'a failure message' to trick them into believing that due to an unexpected error, their partner has failed to compute the session key and thus has aborted the protocol.

Phase 2 (Verifying Password Guesses Offline). C can now verify password guesses both on pw_A and pw_B using the obtained information $(X^*, \overline{Z}^*, \sigma_{AB})$ and $(Y^*, \overline{Z'}^*, \sigma_{BA})$, respectively. (For simplicity, we here describe this verification phase only for pw_A; the case for pw_B proceeds correspondingly.)

Step 1. C computes

$$\begin{aligned} K_C &= \Big(\frac{\overline{Z}^*}{k_{CS}}\Big)^z \\ &= \overline{X}^z \\ &= g^{xzs}, \end{aligned}$$

where k_{CS} is the secret key shared between C and S in Step 2 of session $\Pi_{A,C}$.

Step 2. Note that since $\overline{X}^*$ was replaced with $\widehat{X}^* = \overline{X}^* \cdot g^z$, A must have computed K_A as

$$\begin{aligned} K_A &= \Big(\frac{\widehat{X}^*}{k_{AS}}\Big)^x \\ &= \Big(\frac{\overline{X}^* \cdot g^z}{k_{AS}}\Big)^x \\ &= (g^{zs} \cdot g^z)^x \\ &= g^{xzs} \cdot g^{xz}. \end{aligned}$$

With this in mind, C makes a guess pw'_A on the password pw_A and computes

$$\begin{aligned} X' &= X^*/M^{pw'_A}, \\ K'_A &= K_C \cdot X'^z, \\ sk'_A &= H(A\|B\|K'_A), \\ \sigma'_{AB} &= \mathsf{Mac}_{sk'_A}(A\|B). \end{aligned}$$

Step 3. A verifies the correctness of pw'_A by checking that σ_{AB} is equal to σ'_{AB}. If they are equal, then pw'_A is the correct password with an overwhelming probability.

Step 4. C repeats Steps 2 & 3 (of this verification phase) until a correct password is found.

This offline dictionary attack can be trivially simplified to an insider-attacker version whereby one of the two clients, A and B, tries to discover the other client's password. After all, the S-EA-3PAKE protocol cannot prevent any (malicious) client from mounting an offline dictionary attack against any other client.

3.3. No semantic security

Finally, we point out that the S-EA-3PAKE protocol does not achieve the semantic security of session keys. In S-EA-3PAKE, the session key sk_A (resp. sk_B) is used as the MAC key in generating the authenticator $\sigma_{AB} = \mathsf{Mac}_{sk_A}(A\|B)$ (resp. $\sigma_{BA} = \mathsf{Mac}_{sk_B}(B\|A)$). This oversight leaks some information about the session key and allows an adversary to distinguish the real session key from a random key chosen from the session-key

space. Indeed, S-EA-3PAKE can be easily broken even in the presence of a passive adversary who asks only a single Execute and Test query. A simple attack by such an adversary $\mathcal{A}$ can be described as follows:

1. First, $\mathcal{A}$ makes an $\mathsf{Execute}(\Pi_A^*, \Pi_B^*, \Pi_S^*)$ query, where Π_A^*, Π_B^* and Π_S^* denote any instance of A, B and S, respectively. This query prompts an honest execution of the protocol between the three instances and will return the transcript of the protocol execution.
2. Next, $\mathcal{A}$ makes a $\mathsf{Test}(\Pi_A^*)$ query and receives a key $\overline{sk}$ in response to the query.
3. Then, $\mathcal{A}$ computes $\sigma'_{AB} = \mathsf{Mac}_{\overline{sk}}(A\|B)$ and checks if σ'_{AB} is equal to σ_{AB}. The key $\overline{sk}$ is real if they are equal and otherwise, it is random.

This attack invalidates the existing proof of security for S-EA-3PAKE [4]. We refer the reader to the work of Bellare, Pointcheval and Rogaway [3] for a possible countermeasure.

4. Concluding remarks

The model where S-EA-3PAKE was claimed to be provably secure does not allow the adversary to ask Corrupt queries and thus cannot capture any kind of attacks that can be mounted by malicious clients. Accordingly, neither the man-in-the-middle attack nor the offline dictionary attack described above can be captured in the proof model. This situation is clearly unacceptable, from both theoretic and practical perspectives, and highlights the importance of considering Corrupt queries when proving security of three-party PAKE protocols. Although both the man-in-the-middle attack and the dictionary attack can be easily prevented by modifying the computations of σ_{SA} and σ_{SB} to $\sigma_{SA} = \mathsf{Mac}_{k_{AS}}(S\|\overline{X}^*\|A\|B)$ and $\sigma_{SB} = \mathsf{Mac}_{k_{BS}}(S\|\overline{Y}^*\|B\|A)$, the existence of a security proof for the S-EA-3PAKE protocol in a stronger model remains an open question. We finally note that all the three attacks presented in this note against S-EA-3PAKE also apply to the S-IA-3PAKE protocol [4], a simplified variant of S-EA-3PAKE. This becomes clear as soon as we notice that S-IA-3PAKE is different from S-EA-3PAKE only in that it does not require the transmission of the authenticators σ_{AS} σ_{BS}, σ_{SA} and σ_{SB}.

References

[1] M. Abdalla, P. Fouque, D. Pointcheval, Password-based authenticated key exchange in the three-party setting, In Proceedings of PKC 2005, LNCS 3386: 65–84, 2005.

[2] M. Abdalla, D. Pointcheval, Simple password-based encrypted key exchange protocols, In Proceedings of CT-RSA 2005, LNCS 3376: 191–208, 2005.

[3] M. Bellare, D. Pointcheval, P. Rogaway, Authenticated key exchange secure against dictionary attacks, In Proceedings of EUROCRYPT 2000, LNCS 1807: 139–155, 2000.

[4] T. Lee, T. Hwang, Simple password-based three-party authenticated key exchange without server public keys, Information Sciences 180(9): 1702–1714, 2010.