Cryptanalysis of Server-Aided Password-Based Authenticated Key Exchange Protocols

Junghyun Nam\textsuperscript{1}, Kim-Kwang Raymond Choo\textsuperscript{2}, Juryon Paik\textsuperscript{3}, and Dongho Won\textsuperscript{3,}\textsuperscript{*}

\textsuperscript{1}Department of Computer Engineering, Konkuk University, 322 Danwol-dong, Chungju-si, Chungcheongbuk-do 380-701, Korea
jhnam@kku.ac.kr

\textsuperscript{2}School of Computer and Information Science, University of South Australia
Mawson Lakes, SA 5095, Australia
raymond.choo@unisa.edu.au

\textsuperscript{3}Department of Computer Engineering, Sungkyunkwan University, 300 Cheoncheon-dong, Jangan-gu, Suwon-si, Gyeonggi-do 440-746, Korea
wise96@ece.skku.ac.kr, dhwon@security.re.kr

\textsuperscript{*}The corresponding author

Abstract

Protocols for password-based authenticated key exchange (PAKE) enable two or more parties communicating over a public network to build a secure communication channel using their easy-to-remember passwords. However, off-line dictionary attacks have always been a major security concern in designing such password-based protocols. Compared with the two-party setting, the concern is significantly increased in the three-party setting where insider attacks may be mounted. In this paper, we identified an inherent flaw in the design of Nam et al.’s three-party PAKE protocol (IEEE Communications Letters, 13(3), 2009) and Lu and Cao’s protocol (Computers & Security, 26(1), 2007) and demonstrated that both protocols are susceptible to a previously unpublished off-line dictionary attack. We hope that by identifying this design flaw, similar structural mistakes can be avoided in future design. We conclude the paper with a simple countermeasure.

Keywords: Secure communication, password, key exchange protocol, dictionary attack.

1. Introduction

Protocols for password-based authenticated key exchange (PAKE) enable two or more parties communicating over a public network to generate a high-entropy cryptographic key (also known as a session key) from their low-entropy passwords which are easy for humans to remember [6, 5, 11, 9, 13, 8, 7, 2, 4, 3, 1, 12, 16, 10, 15, 17]. One of the major difficulties in designing a PAKE protocol is to prevent off-line dictionary attacks, in which
an attacker exhaustively enumerates all possible passwords in an off-line manner to find out the correct password [11, 1, 14]. Such attacks have been used by both criminals as well as law enforcement and digital forensics practitioners to enable access to password-protected data (e.g. on smartphones and portable devices based on RIM BlackBerry and Apple iOS platforms - see Elcomsoft Phone Password Breaker http://www.elcomsoft.com/eppb.html). The challenge of designing PAKE protocols secure against off-line dictionary attacks is compounded in the three-party setting. Unlike the two-party setting where the two parties are assumed to share a common password between them, the three-party setting assumes that each party, commonly called a client, shares no passwords with other clients but holds their individual password shared only with a trusted server. This implies that three-party PAKE protocols have to consider the security of passwords against attacks by malicious inside clients who can trivially know the session key [2, 16, 17].

In 2007, Lu and Cao [12] proposed a simple three-party PAKE protocol, named S-3PAKE, which is built upon the earlier two-party PAKE protocol of Abdalla and Pointcheval [4]. The main design goal of S-3PAKE is to provide both security and efficiency without recourse to the use of server’s public keys. Two years later in 2009, Nam et al. [14] demonstrated that the S-3PAKE protocol is vulnerable to an off-line dictionary attack which can be mounted by any client against any other client even without the participation of the victim. Nam et al. also proposed a revised protocol, which we denote by S-3PAKE+, and claimed that it provides resilience against off-line dictionary attacks. We found, however, that the S-3PAKE+ protocol is susceptible to a new off-line dictionary attack (see Section 3). The new attack also applies to the S-3PAKE protocol. By identifying this vulnerability, we hope that similar structural mistakes can be avoided in the future design of three-party PAKE protocols. We conclude the paper with a simple countermeasure against dictionary attacks (see Section 4).

2. The S-3PAKE+ Protocol

This section reviews the S-3PAKE+ protocol [14]. The protocol participants are server S and two clients A and B. The server S assists the clients A and B in establishing a session key by providing them with a central authentication service. Let $pw_A$ and $pw_B$ be the passwords of A and B, respectively. Each client’s password is assumed to be shared with the authentication server S via a secure channel. The public system parameters used in the protocol are as follows:

- A finite cyclic group $G$ of prime order $q$ and a random generator $g$ of the group $G$.
- Two one-way hash functions $G$ and $H$. The outputs of $G$ are the elements of the cyclic group $G$ while the outputs of $H$ are $\ell$-bit strings, where $\ell$ is a security parameter representing the length of session keys.

With the secret passwords and the public parameters established, the S-3PAKE+ protocol proceeds as follows:

1. A chooses a random number $x \in \mathbb{Z}_q$ and computes $X = g^x$, $\overline{pw}_A = G(pw_A)$ and $X^* = X \cdot \overline{pw}_A$. Then A sends $\langle A, X^* \rangle$ to B.
2. B selects a random number $y \in \mathbb{Z}_q$ and computes $Y = g^y$, $\overline{pw}_B = G(pw_B)$ and $Y^* = Y \cdot \overline{pw}_B$. B then sends $\langle A, X^*, B, Y^* \rangle$ to S.
Figure 1. Nam et al.’s three-party PAKE protocol
3. Upon receiving \(\langle A, X^*, B, Y^* \rangle\), \(S\) first recovers \(X\) and \(Y\) by computing \(X = X^*/G(pw_A)\) and \(Y = Y^*/G(pw_B)\). \(S\) aborts the protocol if either \(X\) or \(Y\) is equal to 1. Otherwise, \(S\) selects a random number \(z \in \mathbb{Z}_q\), computes 
\[
X = X^* \cdot pw_B^z, \\
Y = Y^* \cdot pw_A^z, \\
pw_A^{*} = G(A \| B \| S \| X)^{pw_A}, \\
pw_B^{*} = G(B \| A \| S \| Y)^{pw_B}, \\
\overline{X} = \overline{X} \cdot pw_B^z, \\
\overline{Y} = \overline{Y} \cdot pw_A^z,
\]
and sends \(\langle \overline{X}^*, \overline{Y}^* \rangle\) to \(B\).

4. After receiving \(\langle \overline{X}^*, \overline{Y}^* \rangle\), \(B\) computes 
\[
pw_B^{*} = G(B \| A \| S \| Y)^{pw_B}, \\
\overline{X} = \overline{X}^* / pw_B^z, \\
K = \overline{X}^y, \\
\alpha = G(A \| B \| K),
\]
and sends \(\langle \overline{Y}^*, \alpha \rangle\) to \(A\).

5. With \(\langle \overline{Y}^*, \alpha \rangle\) from \(B\), \(A\) computes 
\[
pw_A^{*} = G(A \| B \| S \| X)^{pw_A}, \\
\overline{Y} = \overline{Y}^* / pw_A^z, \\
K = \overline{Y}^x,
\]
and verifies that \(\alpha\) is equal to \(G(A \| B \| K)\). If the verification fails, then \(A\) aborts the protocol. Otherwise, \(A\) computes the session key 
\(SK = H(A \| B \| K)\)
and sends 
\(\beta = G(B \| A \| K)\)
to \(B\).

6. \(B\) verifies the correctness of \(\beta\) by checking that the equation \(\beta = G(B \| A \| K)\) holds. If it holds, then \(B\) computes the session key 
\(SK = H(A \| B \| K)\).
Otherwise, \(B\) aborts the protocol.

The correctness of S-3PAKE\(^+\) can be easily verified from the equations 
\[
K = (\overline{X}^{* \cdot pw_B^z})^y = \overline{X}^y = g^{xy^z}.
\]
and
\[
K = \left( \frac{\overline{Y}^x}{pw_A^x} \right)
\]
\[
= \overline{Y}^x
\]
\[
= g^{xyz}.
\]

Figure 1 shows a high-level depiction of S-3PAKE\(^+\).

3. Off-Line Dictionary Attack

We now show that the S-3PAKE\(^+\) protocol described in Figure 1 is vulnerable to an off-line dictionary attack. As mentioned in the Introduction, S-3PAKE\(^+\) was proposed to address the vulnerability of Lu and Cao’s protocol [12] to an off-line dictionary attack. Although the particular attack against Lu and Cao’s protocol is no longer valid against S-3PAKE\(^+\), we found that S-3PAKE\(^+\) is not secure against off-line dictionary attacks as described below.

We assume that \(A\), a malicious client, wants to find out the password of client \(B\).

**Step 1.** The attacker \(A\) runs the protocol with the client \(B\) and the server \(S\). In this run, the protocol proceeds as per its specification, except that \(A\) interferes with the following message exchanges between \(B\) and \(S\):

- When \(B\) sends \((A, X^*, B, Y^*)\) to \(S\), \(A\) eavesdrops on the message for later use.
- \(A\) replaces the message \((\overline{X}^*, Y^*)\) from \(S\) to \(B\) with the forged one \((\overline{\tilde{X}}^*, \overline{\tilde{Y}}^*)\), where \(\overline{\tilde{X}}^*\) is computed as
  \[
  \overline{\tilde{X}}^* = \overline{X}^* \cdot X.
  \]

**Step 2.** Since \((\overline{X}^*, \overline{Y}^*)\) was replaced with \((\overline{\tilde{X}}^*, \overline{\tilde{Y}}^*)\), \(B\) will compute \(\alpha\) as
\[
\alpha = G(A\|B\|\overline{\tilde{K}})
\]
where
\[
\overline{\tilde{K}} = \left( \frac{\overline{\tilde{X}}^*}{pw_B^x} \right)^y
\]
\[
= \left( \frac{\overline{X}^* \cdot X}{pw_B^x} \right)^y
\]
\[
= \left( \frac{\overline{X} \cdot pw_B^x \cdot X}{pw_B^x} \right)^y
\]
\[
= (\overline{X} \cdot X)^y
\]
\[
= g^{xyz} \cdot g^{xy}.
\]

\(B\) then sends the message \((\overline{\tilde{Y}}^*, \alpha)\) to \(A\) as per the protocol specification.

**Step 3.** Once in possession of \(\overline{\tilde{Y}}^*\) and \(\alpha\), the attacker \(A\) aborts the protocol session alleging that session-key computation has failed due to an unexpected error. \(A\) then computes
\[
K = \left( \frac{\overline{\tilde{Y}}^*}{pw_A^x} \right)^x
\]
\[
= g^{xyz}.
\]
Step 4. A makes a guess $pw_B'$ for the password $pw_B$ and computes
\begin{align*}
Y' &= Y^*/G(pw_B'), \\
\tilde{K}' &= K \cdot Y'^x, \\
\alpha' &= G(A || B || \tilde{K}').
\end{align*}

Step 5. Now, A verifies the correctness of $pw_B'$ by checking that $\alpha$ is equal to $\alpha'$. Notice that if $pw_B'$ and $pw_B$ are equal, then the equation $\alpha = \alpha'$ ought to be satisfied.

Step 6. A repeats Steps 4 and 5 until a correct password is found.

A similar off-line dictionary attack as described above also applies to Lu and Cao’s protocol [12]. Due to similarity, we here only provide a brief description of the attack (using the same notations as used in [12]).

Step 1. The attacker $A$ eavesdrops on the message $A || X || B || Y$ from $B$ to $S$, and replaces the message $X'||Y'$ from $S$ to $B$ with the forged one $X'||\tilde{Y}'$ where $\tilde{Y}'$ is computed as
\[ \tilde{Y}' = Y' \cdot g^x. \]

Step 2. Upon receiving $X'||\tilde{Y}'$, $B$ computes $\alpha = H(A, B, \tilde{K})$ where
\[ \tilde{K} = g^{xy} \cdot g^{xy}, \]
and sends the message $X'||\alpha$ to $A$.

Step 3. After receiving $X'||\alpha$, $A$ aborts the protocol session and computes
\[ K = \left( \frac{X'}{H(A, S, g^x,pw2)} \right)^x = g^{xy}. \]

Step 4. $A$ makes a guess $pw2'$ for $B$’s password $pw2$, and computes
\begin{align*}
\tilde{K}' &= K \cdot (Y/N^pw2')^x, \\
\alpha' &= H(A, B, \tilde{K}').
\end{align*}

Step 5. $A$ verifies the correctness of $pw2'$ by checking that $\alpha$ is equal to $\alpha'$.

Step 6. $A$ repeats Steps 4 and 5 until a correct password is found.

4. Countermeasure and Conclusion

This section presents a simple countermeasure against the off-line dictionary attack. Our idea is to use a block cipher in encrypting $X$ and $Y$ as well as $X$ and $Y$. Consider a block cipher $E : \{0, 1\}^n \times G \rightarrow G$. Each key $k \in \{0, 1\}^n$ defines a permutation $E_k = E(k, \cdot)$ on $G$. Let $D_k$ be the inverse of $E_k$. We will use a hash function,
\[ F : \{0, 1\}^* \rightarrow \{0, 1\}^n, \]
Figure 2. An improved three-party PAKE protocol
to generate the keys for the cipher; and define
\[ k_{A,1} = F(A, B, S, pw_A), \]
\[ k_{B,1} = F(A, B, S, pw_B), \]
\[ k_{A,2} = F(A, B, S, pw_A, X^*), \]
\[ k_{B,2} = F(A, B, S, pw_B, Y^*). \]

We then change the computations of \( X^*, Y^*, \overline{X}^* \) and \( \overline{Y}^* \) to
\[ X^* = E_{k_{A,1}}(X), \]
\[ Y^* = E_{k_{B,1}}(Y), \]
\[ \overline{X}^* = E_{k_{B,2}}(\overline{X}), \]
\[ \overline{Y}^* = E_{k_{A,2}}(\overline{Y}). \]

Consequently, to recover \( X, Y, \overline{X} \) and \( \overline{Y} \), we would need to compute
\[ X = D_{k_{A,1}}(X^*), \]
\[ Y = D_{k_{B,1}}(Y^*), \]
\[ \overline{X} = D_{k_{B,2}}(\overline{X}^*), \]
\[ \overline{Y} = D_{k_{A,2}}(\overline{Y}^*). \]

In other words, we no longer need to compute \( \overline{pw}_A, \overline{pw}_B, pw^*_A \) and \( pw^*_B \); and leave the remaining protocol details unchanged. However, \( S \) must abort the protocol immediately if either \( X \) or \( Y \) is found to be equal to 1. Otherwise, the protocol still suffers from a variant of the off-line dictionary attack. It is easy to see that our modification results in no additional computation/communication costs. The off-line dictionary attack described in Section 3. is no longer valid against our improved protocol since the symmetric encryptions destroy the algebraic property required for the attack. The improved protocol is shown in Figure 2.

We finally note that the protocol S-3PAKE + (and S-3PAKE) is vulnerable to undetectable on-line dictionary attacks, in which each password guess is verified via a new on-line transaction with the server. But, as mentioned in [14], this vulnerability is not just a failure of S-3PAKE + but it’s an inherent weakness of all the three-party PAKE protocols that does not allow the server to authenticate messages from clients. Hence, no quick tweak is available but a major change is necessary to prevent undetectable on-line dictionary attacks. One way to prevent such on-line attacks is to let \( A \) (resp. \( B \)) (1) establish a MAC (message authentication code) key with \( S \) by running a 2-party PAKE protocol, (2) generate a MAC of \( X^* \) (resp. \( Y^* \)) using the MAC key, and (3) send the MAC together with \( X^* \) (resp. \( Y^* \)) to \( S \) who can then use it in verifying the authenticity of \( X^* \) (resp. \( Y^* \)). This approach requires additional message exchanges between the server and the clients. However, an increase in communication/computation costs seems to be inevitable to avoid on-line dictionary attacks.

Acknowledgements

This work was supported by Priority Research Centers Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and
References


**Authors**

**Junghyun Nam**
received the B.E. degree in Information Engineering from Sungkyunkwan University, Korea, in 1997. He received his M.S. degree in Computer Science from University of Louisiana, Lafayette, in 2002, and the Ph.D. degree in Computer Engineering from Sungkyunkwan University, Korea, in 2006. He is now an associate professor in Konkuk University, Korea. His research interests include cryptography and computer security.

**Kim-Kwang Raymond Choo**
received the PhD in Information Security from Queensland University of Technology in 2006. He has (co-)authored a number of publications including a book published in Springer’s “Advances in Information Security” book series and six refereed monographs; and is the recipient of the 2010 ACT Pearcey Award, 2009 Fulbright Scholarship, 2008 Australia Day Achievement Medallion, British Computer Society’s Wilkes Award for the best paper published in the 2007 volume of *The Computer Journal*, 2007 Queensland University of Technology Faculty of Information Technology Executive Dean’s outstanding Ph.D. thesis commendation, and the 2005 Australasian Conference on Information Security and Privacy Best Student Paper Award.

**Juryon Paik**
received the B.E. degree in Information Engineering from Sungkyunkwan University, Korea, in 1997. She received her M.E. and Ph.D. degrees in Computer Engineering from Sungkyunkwan University in 2005 and 2008, respectively. Currently, she is a research professor at the De-
partment of Computer Engineering, Sungkyunkwan University. Her research interests include XML mining, semantic mining, and web search engines.

Dongho Won
received his B.E., M.E., and Ph.D. degrees from Sungkyunkwan University in 1976, 1978, and 1988, respectively. After working at ETRI (Electronics & Telecommunications Research Institute) from 1978 to 1980, he joined Sungkyunkwan University in 1982, where he is currently Professor of School of Information and Communication Engineering. In the year 2002, he served as the President of KIISC (Korea Institute of Information Security & Cryptology). He was the Program Committee Chairman of the 8th International Conference on Information Security and Cryptology (ICISC 2005). His research interests are on cryptology and information security.