Cryptanalysis of a Simple Three-party Key Exchange Protocol

He Debiao, Chen Jianhua and Hu Jin
School of Mathematics and Statistics, Wuhan University, Wuhan, Hubei 430072, China
E-mail: {hedebiao, chenjh_ecc, hujin_ecc}@163.com

Keywords: key exchange protocol, secure communication, password, dictionary attack, Diffie-Hellman assumption

Received: November 5, 2009

Key exchange protocols allow two or more parties communicating over a public network to establish a common secret key called a session key. Due to their significance in building a secure communication channel, a number of key exchange protocols have been suggested over the years for a variety of settings. Among these is the so-called S-3PAKE protocol proposed by Lu and Cao for password-authenticated key exchange in the three-party setting. In the current work, we are concerned with the password security of the S-3PAKE protocol. We first show that S-3PAKE is vulnerable to an off-line dictionary attack in which an attacker exhaustively enumerates all possible passwords in an off-line manner to determine the correct one. We then figure out how to eliminate the security vulnerability of S-3PAKE.

Povzetek: Prispevek se ukvarja z varnostjo v protokolu S-3PAKE.

1 Introduction

In 1992, Bellovin and Merritt [1] proposed a two-party encrypted key exchange protocol based on user passwords. Since then, many two-party password-based authenticated key exchange (2PAKE) protocols have been proposed. Because 2PAKE protocols are only suitable for the client − server architecture, some researchers extended 2PAKE protocols into 3PAKE protocols for three-party environments. Most existing 3PAKE protocols are designed for the client− client− server architecture, in which each client (user) shares his password with a trusted server and resorts to the server to authenticate the peer for establishing a session key. In 2004, Lee et al. [2] presented two enhanced three-party encrypted key exchange (3PEKE) protocols without using public key techniques, and showed that their protocols can resist several attacks, achieve mutual authentica-tion, and provide perfect forward secrecy. In 2005, Wen et al. [3] proposed a 3PAKE protocol using Weil pairing and claimed that their protocol is provably secure against active adversaries in the random oracle model. However, Nam et al. [4] showed that Wen et al. ' s protocol cannot resist a man-in-the-middle attack, and then interpreted their attack in the context of the formal proof model. Recently, Lu and Cao [5] proposed a new simple 3PAKE (S-3PAKE) protocol based on the chosen-basis computational Diffie-Hellman (CCDH) assumption. They claimed that their protocol is superior to similar protocols with respect to security and efficiency.

According to recent works [6-9], S-3PAKE is vulnerable to various attacks like man-in-the-middle attacks [6, 7], an unknown key-share attack [8], undetectable on-line dictionary attacks [7-8], and off-line dictionary attacks[9].

The above-mentioned attacks are not the only ones that can compromise the security of the S-3PAKE protocol. We found that S-3PAKE is not secure against an off-line dictionary attack. The present work reports this new (and more serious) security problem with S-3PAKE and, in addition, shows how to fix it.

In this paper, we will first review the S-3PAKE protocol, and then demonstrate the attacks on the S-3PAKE protocol. Furthermore, we will suggest a countermeasure to enhance the security of the S-3PAKE protocol against the attacks.

2 The S-3PAKE protocol

The notations used in the S-3PAKE protocol are described as in the following:

- (G , g , p) represents a finite cyclic group G generated by an element g with prime order p .
- M and N denote two elements in G .
- S represents a trusted server.
- A and B represent the initiator and the responder, respectively, of a protocol run.
- pw_A denotes the password shared between A and S .
- pw_B denotes the password shared between B and S .
- $H()$ and $H'()$ denote two distinct secure one-way hash functions.

The security of the S-3PAKE protocol mainly relies on the chosen-basis computational Diffie-Hellman CCDH) assumption [5], which is a variation of the computational Diffie-Hellman (CDH) assumption. In he CDH assumption, given g^u and g^v, where u and v are drawn randomly from Z_p, it is computationally infeasible to compute g^{uv}, which is denoted by CDH(g^u, g^v). And, the CCDH assumption is defined as in the following: The adversary is given three random elements, M, N, and X in G, with the goal of finding a triple of values (Y, u, v) such that u = CDH(X, Y) and v = CDH($X / M, Y / N$). The idea behind the CCDH assumption is that the adversary may be able to successfully compute either u (by choosing $Y = g$ to obtain $u = X$) or v (by choosing $Y = g \cdot N$ to obtain $v = X / M$), but not both. Note that all modular operations in this paper are performed under modulo p, and we drop the operator $\bmod p$ for clearness. Suppose that A and B request to authenticate each other and then resort to S for a session key agreement. The steps of the protocol, can be briefly described as in the following:

Step 1. A chooses a random number $x \in Z_p$, computes $X = g^x$ and $X^* = X \cdot M^{pw_A}$, and sends $A \| X^*$ to B.

Step 2. B selects a random number $y \in Z_p$, computes $Y = g^y$ and $Y^* = Y \cdot N^{pw_B}$, and sends $A \| X^* \| B \| Y^*$ to S.

Step 3. Upon receiving $A \| X^* \| B \| Y^*$, S first recovers X and Y by computing $X = X^* / M^{pw_A}$ and $Y = Y^* / N^{pw_A}$. Next, S selects a random number $z \in Z_p$ and computes $\overline{X} = X^z$ and $\overline{Y} = Y^z$. S then computes $pw_A^* = H(A \| S \| X)^{pw_A}$, $pw_B^* = H(B \| S \| Y)^{pw_B}$, $\overline{X}^* = \overline{X} \cdot pw_B^*$, $\overline{Y}^* = \overline{Y} \cdot pw_A^*$ and sends $\overline{X}^* \| \overline{Y}^*$ to B.

Step 4. After having received $\overline{X}^* \| \overline{Y}^*$, B computes $pw_B^* = H(B \| S \| Y)^{pw_B}$, $K = (\overline{X}^* / pw_B^*)^y$, $\alpha = H(A \| B \| K)$, and sends $\overline{Y}^* \| \alpha$ to A.

Step 5. With $\overline{Y}^* \| \alpha$ from B, A computes $pw_A^* = H(A \| S \| Y)^{pw_A}$, $K = (\overline{Y}^* / pw_A^*)^x$, and verifies that α is equal to $\alpha = H(A \| B \| K)$. If the verification fails, then A aborts the protocol. Otherwise, A computes the session key $SK_A = H'(A \| B \| K)$ and sends $\beta = H(B \| A \| K)$ to B.

Step 6. B verifies the correctness of β by checking that the equation $\beta = H(B \| A \| K)$ holds or not. If it holds, then B computes the session key $SK_B = H'(A \| B \| K)$. Otherwise, B aborts the protocol.

The correctness of S-3PAKE can be easily verified [5].

3 Off-line dictionary attack on S-3PAKE

Lu and Cao [5] claim that their S-3PAKE protocol is secure against off-line dictionary attacks. This argument may hold if there only exist honest clients who stick to the protocol specification. But, there could be malicious clients who deviate from the protocol.

Indeed, we found that S-3PAKE is not secure against an offline dictionary attack in the presence of a malicious client. Assume that B is a malicious client, and wants to find out the password of client A. Then the following description represents our off-line dictionary attack mounted by B against A's password.

In the S-3PAKE protocol, the user B can easily obtain valid information $A \| X^*$, since all transcripts are transmitted over an open network. Then he/she does the following steps.

Phase 1. The attacker B runs the protocol with the server S while playing dual roles of B itself and the victim A.

1) B computes $Y^* = 1 \cdot N^{pw_B}$. Then B sends $A \| X^* \| B \| Y^*$ to S.
2) Upon receiving $A \| X^* \| B \| Y^*$, S first recovers X and Y by computing $X = X^* / M^{pw_A}$ and $Y = Y^* / N^{pw_A} = 1$. Next, S selects a random number $z \in Z_p$ and computes $\overline{X} = X^z$ and $\overline{Y} = Y^z = 1$. S then computes $pw_A^* = H(A \| S \| X)^{pw_A}$, $pw_B^* = H(B \| S \| Y)^{pw_B}$, $\overline{X}^* = \overline{X} \cdot pw_B^*$, $\overline{Y}^* = \overline{Y} \cdot pw_A^* = pw_A^* = H(A \| S \| X)^{pw_A}$ and sends $\overline{X}^* \| \overline{Y}^*$ to B.

Phase 2. When B receives ($\overline{X}^* \| \overline{Y}^*$), he/she can carry out the off-line dictionary attack using $X^* = X \cdot M^{pw_A}$ and $\overline{Y}^* = H(A \| S \| X)^{pw_A}$. The process of the off-line dictionary attack is as follows.

Step1. B selects a password s from a uniformly distributed dictionary D.

Step2. B computes $X' = X^* / M^s$ and $\overline{Y}^{*\prime} = H(A \| S \| X')^s$.

Step3. B then verifies the correctness of s by checking that $\overline{Y}^{*\prime}$ is equal to $\overline{Y}^*$.

Step3. B repeats steps 1, 2, and 3 of this phase until a correct password if found.

This off-line dictionary attack may lead to devastating losses of passwords, because it can be mounted against any registered client and does not even require the participation of the victim, and the steps for verifying password guesses can be performed in an off-line manner by an automated program.

It's easy to say that if B let Y^* be $(p-1)\cdot N^{pw_B}$, he/she can get the correct password pw_A by the same way. In a similar way, A can guess the password of the client B if A is a malicious client.

4 Countermeasure

The random number z is used in the S-3PAKE protocol to randomizing X and Y, but if Y^* is set $1\cdot N^{pw_B}$ or $(p-1)\cdot N^{pw_B}$, the function of the random number z is destroyed, then a malicious client can get the password pw_A through the off-line dictionary attack. Fortunately, in order to make the S-3PAKE protocol against the attack, we just need let the S check

$$X = 1, -1 \text{ and } Y = 1, -1$$

hold or not. If one of above equations holds, then S stops the protocol.

Theorem 1. The modified S-3PAKE scheme is secure against the off-line dictionary attack described in section 3.

Proof: When the server S checks $X = 1, -1$ and $Y = 1, -1$ hold or not. He will find the off-line dictionary attack by the malicious client. He stops the session, and records the attack in his database. After finding the attack for several times (three time for example), S stop the service supplied to the malicious client.

So, our countermeasure can withstand the off-line dictionary attack.

5 Conclusion

Herein, we have demonstrated that Lu-Cao's S-3PAKE protocol is potentially vulnerable to an off-line dictionary attack. The reason for the attack is due to the fact that the function of randomization is destroy when X or Y is set some special value. To enhance the security of the S-3PAKE protocol, we have suggested a countermeasure to resist our described attacks while the merits of the original protocol are left unchanged.

Reference

[1] S.M. Bellovin, M. Merritt (1992) "Encrypted key exchange: password-based protocols secure against dictionary attacks," *Proc. 1992 IEEE Symposium on Research in Security and Privacy*, pp. 72– 84.

[2] T.F. Lee, T. Hwang, C.L. Lin (2004) "Enhanced three-party encrypted key exchange without server public keys," *Computers Security* 23 (7), pp. 571–577.

[3] H.A. Wen, T.F. Lee, T. Hwang (2005) "Provably secure three-party password-based authenticated key exchange protocol using Weil pairing," *IEE Proceedings Communications* 152 (2) , pp. 138–143.

[4] J. Nam, Y. Lee, S. Kim, D. Won (2007) "Security weakness in a three-party pairing-based protocol for password authenticated key exchange," *Information Sciences* 177 (6), pp. 1364– 1375.

[5] R. Lu, Z. Cao (2007) "Simple three-party key exchange protocol," Computers Security 26 (1), pp. 94– 97.

[6] H.-R. Chung and W.-C. Ku (2008) "Three weaknesses in a simple three-party key exchange protocol," *Inform. Sciences* 178(1), pp. 220–229.

[7] H. Guo, Z. Li, Y. Mu, and X. Zhang (2008) "Cryptanalysis of simple threeparty key exchange protocol," Computers & Security, 27(1), pp. 16–21.

[8] R. C.-W. Phan, W.-C. Yau, and B.-M. Goi (2008) "Cryptanalysis of simple three-party key exchange protocol (S-3PAKE)," Inform. Sciences,178, (13), pp. 2849–2856.

[9] J. Nam, J. Paik, H. Kang, et al. (2009) An Off-Line Dictionary Attack on a Simple Three-Party Key Exchange Protocol, *IEEE COMMUNICATIONS LETTERS*, 13(3), pp. 205-207.