Three-party Encrypted Key Exchange Without Server Public-Keys

Chun-Li Lin, Hung-Min Sun, Michael Steiner and Tzonelih Hwang

Abstract— **Three-party key-exchange protocols with password authentication — clients share an easy-to-remember password with a trusted server only — are very suitable for applications requiring secure communications between many light-weight clients (end users); it is simply impractical that every two clients share a common secret. In 1995, Steiner, Tsudik and Waidner proposed a realization of such a three-party protocol based on the *Encrypted Key Exchange* (EKE) protocols. However, their protocol was later demonstrated to be vulnerable to off-line and undetectable on-line guessing attacks. In 2000, Lin, Sun and Hwang proposed a secure three-party protocol with server public-keys. However, the approach of using server public-keys is not always a satisfactory solution and is impractical for some environments. In this article, we propose a secure three-party EKE protocol without server public-keys.**

I. Introduction

In 1992, Bellovin and Merritt [1] proposed the *Encrypted Key Exchange* (EKE) family of key exchange protocols, which allow people to use easy-to-remember (and therefore intrinsically weak) passwords without being threatened by dictionary attacks [2]. In the EKE protocol, two communication parties A and B securely share a password in advance, and authentication is achieved after these two parties obtain a common ephemeral session key.

In 1995, Steiner, Tsudik, and Waidner [3] proposed a three-party EKE protocol (hereafter referred to as STW-3PEKE) in which all clients share a password with a trusted server S only and in which S mediates between two communication parties to allow their mutual authentication. The three-party EKE protocol is particularly well-suited for applications that require secure communication between many light-weight and mobile clients (end users): On the one hand, it is impractical that every two clients share a common secret. On the other hand, a heavyweight infrastructure, e.g., public keys and a public key infrastructure, is often not tolerable.

However, Ding and Horster [4] showed in 1995 that the STW-3PEKE is not resistant to undetectable on-line password guessing attacks. They divided password guessing attacks into three classes:

1. *Detectable on-line password guessing attacks*: An attacker attempts to use a guessed password in an on-line transaction. He verifies the correctness of his guess using the response from S. A failed guess can be detected and logged by the server S.
2. *Undetectable on-line password guessing attacks*: Similar to above, an attacker tries to verify a password guess in an on-line transaction. However, a failed guess can *not* be detected and logged by S, as S is not able to distinguish an honest request from a malicious one.
3. *Off-line password guessing attacks*: An attacker guesses a password and verifies his guess off-line. No participation of S is required, so S does not notice the attack.

Among the three classes of attacks, off-line password guessing attacks are the most promising ones for an attacker. Undetectable on-line password guessing attacks are less critical than off-line attacks. Nevertheless, a secure 3PEKE protocol should ideally resist both types of undetectable attacks. Detectable on-line password guessing attacks cannot be avoided. However, they can be handled appropriately, e.g., by introducing exponentially increasing delays after failed attempts and locking the account after an excessive amount of failures.

In [5], Lin, Sun and Hwang pointed out that the STW-3PEKE is not only vulnerable to undetectable on-line password guessing attacks but also vulnerable to an off-line password guessing attack. They also proposed a 3PEKE protocol (hereafter referred to as LSH-3PEKE), in which the server holds a permanent and publicly known public-key to prevent both off-line and undetectable on-line password guessing attacks.

The approach of using server public-keys in 3PEKE is suitable when the number of message exchanged is of most concern. However, communication parties have to obtain and verify the public-key of the server, a task which puts a high burden on the user [6]. In fact, key distribution services without public-keys are quite often superior in practice than PKIs or are at least widely deployed. Unfortunately, traditional three-party key distribution services such as Kerberos [7] and KryptoKnight [8] are all susceptible to dictionary attacks with weak passwords and do not immediately provide forward-security.

In this paper, we propose a new 3PEKE protocol which is resistant to both off-line and undetectable on-line password guessing attacks but does not require server public-keys. As such, it could serve as the basis for a key distribution service overcoming the deficiencies of Kerberos or KryptoKnight mentioned above.

The remainder of this paper is organized as follows. In Section II, we review related work on 3PEKE. In Section III, we propose a new 3PEKE protocol without server public-keys. Then, Section IV analyzes the security and

Manuscript received February 9, 2001. The associate editor coordinating the review of this letter and approving it for publication was Dr. L. Chen. This work was supported in part by the National Science Council of the Republic of China under Contract NSC90-2213-E-006-100.

The authors are with the Department of Computer Science and Information Engineering, National Cheng Kung University, Tainan, Taiwan 701 (e-mail: hmsun@mail.ncku.edu.tw), and the Fachrichtung Informatik, Universität des Saarlandes, Saarbrücken, Germany.

Publisher Item Identifier S 1089-7798(01)01256-9.

performance of the proposed protocol. Finally, concluding remarks are given in Section V.

II. Related Work on 3PEKE

A. Notation

The following notation is used throughout this paper:

A, B	Communication parties.
S	The trusted server.
P_A, P_B	Passwords securely shared by A (B) with S.
K_S	S's public key.
A^*	An attacker masquerading as A.
C_B	A random challenge generated by B.
$[M]_K$	Encryption of M using a symmetric encryption scheme with a (cryptographically strong) shared key K.
$\langle M \rangle_{P_I}$	Encryption of M using a password-based symmetric encryption scheme with I's password P_I. This scheme must behave like an ideal cipher to prevent production of verifiable ciphertext [6].
$\{M\}_K$	Encryption of M using an asymmetric encryption scheme with public key K. This scheme must be secure against adaptively chosen ciphertext attacks [9].
$f_K(M)$	Pseudo-random function (PRF) indexed by K and applied to M. Used mainly as message authentication codes.
$h(M)$	An one-way hash function applied to M.
$H_1(k), H_2(k)$	Independent key-derivation functions, e.g., taking different bits from a pseudo-random number generator seeded with k.
p, g	A large prime p and a generator g in the cyclic group $\mathbb{Z}_p^*$, a group in which the Diffie-Hellman problem is considered hard.
N_A, N_B, N_S	Random exponents chosen by A, B and S.
R_A, R_B	$R_A \equiv g^{N_A} \pmod p$, $R_B \equiv g^{N_B} \pmod p$.
$flow\ i$	The data sent in step i of a protocol.
$A \Rightarrow B : M$	A sending message M to B.

B. The STW-3PEKE

1. $A \Rightarrow B : \langle R_A \oplus B \rangle_{P_A}$
2. $B \Rightarrow S : A, \langle R_A \oplus B \rangle_{P_A}, \langle R_B \oplus A \rangle_{P_B}$
3. $S \Rightarrow B : {R_A}^{N_S}, {R_B}^{N_S}$
For B, $K \equiv ({R_A}^{N_S})^{N_B} \equiv g^{N_A \cdot N_B \cdot N_S} \pmod p$.
4. $B \Rightarrow A : {R_B}^{N_S}, [flow1]_K$
For A, $K \equiv ({R_B}^{N_S})^{N_A} \equiv g^{N_A \cdot N_B \cdot N_S} \pmod p$.
5. $A \Rightarrow B : [[flow1]_K]_K$

C. Undetectable On-line Guessing Attacks on STW-3PEKE

The following scenario [4] demonstrates that the STW-3PEKE is not resistant to undetectable on-line password guessing attacks. In this scenario, the attacker B, who is valid but malicious, completes the protocol with S and no participation of A is required.
1. B: Records $\langle R_A \oplus B \rangle_{P_A}$ from an arbitrary run.
B guesses a password $\tilde{P}_A$ and computes the value $\tilde{R}_A$, then sets $R_B = \tilde{R}_A$.
2. $B \Rightarrow S : A, \langle R_A \oplus B \rangle_{P_A}, \langle \tilde{R}_A \oplus A \rangle_{P_B}$
3. $S \Rightarrow B : {R_A}^{N_S}, \tilde{R}_A^{N_S}$
B compares the two values. If ${R_A}^{N_S} = \tilde{R}_A^{N_S}$, it follows that $R_A = \tilde{R}_A$ and so B has guessed the correct password $\tilde{P}_A = P_A$.

D. Off-line Guessing Attack on STW-3PEKE

The following shows an off-line guessing attack on STW-3PEKE proposed in [5]:
1. $A^* \Rightarrow B : X$
An arbitrary attacker sends on behalf of A a random number X to B, in which the length of X is the same as the length of $\langle R_A \oplus B \rangle_{P_A}$.
2. $B \Rightarrow S^* : A, X, \langle R_B \oplus A \rangle_{P_B}$
The attacker intercepts the message sent from B to S. He chooses two random exponents $\tilde{N}_A$ and $\tilde{N}_S$, computes $\tilde{R}_A \equiv g^{\tilde{N}_A} \pmod p$ and sends $\tilde{R}_A^{\tilde{N}_S}, Y$ to B, where Y is a random number in Z_p^*.
3. $S^* \Rightarrow B : \tilde{R}_A^{\tilde{N}_S}, Y$
For B, $K \equiv (\tilde{R}_A^{\tilde{N}_S})^{N_B} \equiv g^{\tilde{N}_A \cdot N_B \cdot \tilde{N}_S} \pmod p$.
4. $B \Rightarrow A^* : Y, [flow1]_K$
The attacker intercepts the message sent from B to A. Hereafter, he tests different passwords off-line until he finds the correct one: For each password guess $\tilde{P}_B$, get $\tilde{R}_B$ from $\langle R_B \oplus A \rangle_{P_B}$, compute $\tilde{K} \equiv (\tilde{R}_B^{\tilde{N}_S})^{\tilde{N}_A} \equiv g^{\tilde{N}_A \cdot \tilde{N}_B \cdot \tilde{N}_S} \pmod p$, decrypt $[flow1]_K$ with $\tilde{K}$ and check $flow1 \stackrel{?}{=} X$.

E. The LSH-3PEKE With Server's Public-key

The following shows the 3PEKE protocol proposed in [5] to solve above problems. The scheme assumes that the server's public key K_S is authenticated before it is used.
1. $A \Rightarrow B : A, \{ra, R_A, P_A\}_{K_S}$
ra is a random number generated by A and used as an one-time strong key shared by A and S.
2. $B \Rightarrow S : A, \{ra, R_A, P_A\}_{K_S}, \{rb, R_B, P_B\}_{K_S}$
rb is a one-time strong key shared by B and S.
3. $S \Rightarrow B : [B, R_B]_{ra}, [A, R_A]_{rb}$
For B, $K \equiv {R_A}^{N_B} \equiv g^{N_A \cdot N_B} \pmod p$.
4. $B \Rightarrow A : [B, R_B]_{ra}, [h(flow1), C_B]_K$
For A, $K \equiv {R_B}^{N_A} \equiv g^{N_A \cdot N_B} \pmod p$.
5. $A \Rightarrow B : C_B$

III. LSSH-3PEKE – A New Protocol Without Server's Public-key

In this section, we propose a new 3PEKE protocol without server public-keys. The benefits of such approach are addressed in Section I. As stated in [5], the most important requirement to prevent undetectable on-line guessing attacks is to provide authentication of A's and B's message to S. That is, the server S responds with flow 3 of LSH-3PEKE only if S is sure that the prior messages did originate from A and B. To satisfy such a requirement and to prevent off-line guessing attacks simultaneously, some form of public-key techniques is unavoidable. We use Diffie-Hellman key exchanges to produce one-time keys instead of the usage of the server's long-term public-key. The protocol works as follows:

1. $A \Rightarrow S : A, B$
A sends IDs A, B to S as an initial request.
2. $S \Rightarrow A : \langle g^{N_{S1}} \rangle_{P_A}, \langle g^{N_{S2}} \rangle_{P_B}$
N_{S1} and N_{S2} are exponents randomly chosen by S.
3. $A \Rightarrow B : A, R_A, f_{K_{A,S}}(A, B, g^{N_{S1}}), \langle g^{N_{S2}} \rangle_{P_B}$
A chooses a random exponent N_A, computes $R_A \equiv g^{N_A} \pmod p$ for the contribution of the session key with B, and computes $K_{A,S} \equiv (g^{N_{S1}})^{N_A} \pmod p$ as the one-time key with S.
4. $B \Rightarrow S : R_A, f_{K_{A,S}}(A, B, g^{N_{S1}}), R_B, f_{K_{B,S}}(A, B, g^{N_{S2}})$
B chooses a random exponent N_B, computes $R_B \equiv g^{N_B} \pmod p$ for the contribution of the session key with A, and computes $K_{B,S} \equiv (g^{N_{S2}})^{N_B} \pmod p$ as the one-time key with S.
5. $S \Rightarrow B : f_{K_{B,S}}(A, B, R_A, R_B), f_{K_{A,S}}(A, B, R_B, R_A)$
S computes $K_{A,S}$ and $K_{B,S}$ and authenticates A and B by verifying $f_{K_{A,S}}(A, B, g^{N_{S1}})$ and $f_{K_{B,S}}(A, B, g^{N_{S2}})$, respectively. If successful, S proceeds by sending two "one-time certificates" on the temporary DH public keys to B.
6. $B \Rightarrow A : R_B, f_{K_{A,S}}(A, B, R_B, R_A), f_{K'}(A, B, R_A)$
B validates R_A by checking $f_{K_{B,S}}(A, B, R_A, R_B)$ and computes the session key as $K \equiv H_1({R_A}^{N_B} \pmod p)$. Then, B forwards R_B and $f_{K_{A,S}}(A, B, R_B, R_A)$ to A together with an authentication and key confirmation message $f_{K'}(A, B, R_A)$, where $K' \equiv H_2({R_A}^{N_B} \pmod p)$.
7. $A \Rightarrow B : f_{K'}(A, B, R_B)$
A validates R_B by checking $f_{K_{A,S}}(A, B, R_B, R_A)$ and computes the session key as $K \equiv H_1({R_B}^{N_A} \pmod p)$. It then verifies the key confirmation message to authenticate B. Finally, A computes $K' \equiv H_2({R_B}^{N_A} \pmod p)$ and sends $f_{K'}(A, B, R_B)$ to B. This message allows B to authenticate A and confirm A's knowledge of the session key.

IV. Security and Performance Analysis

Here we only provide informal security analysis due to limitation of space. An off-line guessing attack will not work on our protocol because no verifiable information is encrypted by passwords. The only way to verify the decrypting result $g^{N'_{S1}}$ (or $g^{N'_{S2}}$) from a guessing password is to obtain the Diffie-Hellman one-time key $K'_{A,S}$ from $g^{N'_{S1}}$ and R_A (or $K'_{B,S}$ from $g^{N'_{S2}}$ and R_B). But this is the *Diffie-Hellman Problem* and considered computational infeasible. Besides, after step 4 of our protocol, S can accurately authenticate A and B, thus undetectable on-line guessing attacks also will not work. Notice that in our protocol R_A and R_B are transmitted in plaintext form. This decreases the encryption/decryption operations and reduces the risk of guessing attacks. The PRFs on R_A and R_B provide authentication by S to A and B such that the man-in-the-middle attack is impossible. Our protocol satisfies the property of perfect forward secrecy since a compromised password doesn't reveal any contributions (N_A and N_B) of an old session key. It also satisfies the property of known-key security because the ephemeral random exponents N_A and N_B are independent among every protocol run.

Clearly, the proposed protocol needs two steps more than LSH-3PEKE. However, considering that exponentiations are the main computational cost it turns out that LSH-3PEKE is computationally more expensive than the newly proposed scheme. Note that the best-known provably non-malleable encryption scheme [9] needs 5 and 3 exponentiations per encryption and decryption, respectively. Table I shows the detailed comparisons.

Moreover, as stated in Section I, the LSH-3PEKE either needs to get and validate server's public-key or needs additional storage device to safeguard server's public-key. Thus, the proposed protocol provides another solution to 3PEKE, especially for those environments where end users cannot be expected to carry around or correctly validate the server's public key.

TABLE I
Performance comparison LSH-3PEKE vs. LSSH-3PEKE.

	LSH-3PEKE			LSSH-3PEKE		
steps	5			7		
	A	B	S	A	B	S
modular exponentiation[1]	2(7)	2(7)	0(6)	3	3	4
public-key en/decryption	1/0	1/0	0/2	0	0	0
symmetric en(de)cryption	2	2	2	1	1	2
PRF operation	0	0	0	3	3	4
random numbers	2	3	0	1	1	2

[1]The second entry takes also the public-key en/decryption into account.

V. Conclusions

We propose a secure 3PEKE protocol with no need for server public-keys. Compared with LSH-3PEKE, the proposed protocol has considerably lower computational cost and, foremost, needs not to worry about the validation or compromise of server public-keys. All the parties have to do is just to remember their passwords.

References

[1] Steven M. Bellovin and Michael Merrit, "Encrypted key exchange: Password-based protocols secure against dictionary attacks," in *Proceedings of the IEEE Symposium on Research in Security and Privacy*, pp. 72–84, 1992.
[2] Robert Morris and Ken Thompson, "Password security: A case history," *Communications of the ACM*, pp. 594–597, 1979.
[3] Michael Steiner, Gene Tsudik, and Michael Waidner, "Refinement and extension of Encrypted Key Exchange," *ACM Operating Systems Review*, vol. 29, no. 3, pp. 22–30, 1995.
[4] Yun Ding and Patrick Horster, "Undetectable on-line password guessing attacks," *ACM Operating Systems Review*, vol. 29, no. 4, pp. 77–86, 1995.
[5] Chun-Li Lin, Hung-Min Sun, and Tzonelih Hwang, "Three-party encrypted key exchange: Attacks and a solution," *ACM Operating Systems Review*, vol. 34, no. 4, pp. 12–20, 2000.
[6] Michael Steiner, Peter Buhler, Thomas Eirich, and Michael Waidner, "Secure password-based cipher suite for TLS," *ACM Trans. on Information and System Security*, vol. 4, no. 2, 2001.
[7] B. Clifford Neuman and Theodore Ts'o' "Kerberos: An Authentication Service for Computer Networks," *IEEE Communications*, vol. 32, no. 9, pp. 33–38, 1994.
[8] Refik Molva, Gene Tsudik, Els van Herreweghen, and Stefano Zatti, "KryptoKnight Authentication and Key Distribution System," *Proceedings of the 1992 European Symposium on Research in Computer Security - ESORICS*, pp. 1–16, 1992.
[9] Ronald Cramer and Victor Shoup, "A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack," in *CRYPTO'98*, pp. 13–25.