Comments on Weaknesses in Two Group Diffie-Hellman Key Exchange Protocols

Jin Wook Byun and Dong Hoon Lee
Center for Information Security Technologies (CIST),
Korea University, Anam Dong, Sungbuk Gu, Seoul, Korea
{byunstar, donghlee}@korea.ac.kr

July 1, 2005

Abstract

In [3], Tang presented two password guessing attacks such as off-line and undetectable on-line dictionary attacks against password-based group Diffie-Hellman key exchange protocols by Byun and Lee [2]. In this paper, we present countermeasures for two attacks by Tang.

1 Introduction

Very recently, Byun and Lee suggested two provably secure N-party encrypted Diffie-Hellman key exchange protocols using different passwords [2]. One is an N-party EKE-U in the unicast network and the other is an N-party EKE-M in the multicast network. In [3], Tang showed that N-party EKE-U and N-party EKE-M protocols suffered from off-line dictionary attacks and undetectable on-line guessing attack by malicious insider attackers, respectively. In this paper, we present countermeasures for the two attacks by Tang.

2 Attack on N-party EKE-U and its Countermeasure

2.1 Off-line Dictionary Attack on N-party EKE-U

Let G=$\langle g \rangle$ be a cyclic group of prime order q. In [3], Tang first presented an off-line dictionary attack on N-party EKE-U protocol by malicious insider attacker as follows.[1]

- **Step 1** : A malicious user $\mathcal{U}_j$ first selects two random values α, β, and sends m_j to its neighbor $\mathcal{U}_{j+1}$ where

$$m_j = \mathcal{E}_{pw_i}(X_j),\ X_j = \{g^{\alpha}, g^{\alpha\beta}, g^{\gamma_3}, ..., g^{\gamma_j}, g^{V_j\xi_j}\}$$
$$\gamma_k = V_j(\xi_j/x_k) \text{ where } x_k \in Z_q^* \text{ and } 3 \leq k \leq j$$
$$V_j = v_1 \cdot v_2 \cdots v_j,\ \xi_j = x_1 \cdot x_2 \cdots x_j.$$

- **Step 2** : $\mathcal{U}_{j+1}$ just forwards m_j to server $\mathcal{S}$. $\mathcal{S}$ decrypts m_j with password pw_j and computes m_{j+1} with a password pw_{j+1} and a randomly selected value v_{j+1} where

[1] For detailed descriptions of N-party EKE-U protocol, please refer to the paper [2].

$$m_{j+1} = \mathcal{E}_{pw_{j+1}}(X_{j+1}),\ X_{j+1} = \{g^{\alpha v_{j+1}}, g^{\alpha\beta v_{j+1}}, g^{\gamma_3}, ..., g^{\gamma_{j+1}}, g^{V_{j+1}\xi_j}\}$$
$$\gamma_k = g^{V_{j+1}(\xi_{j+1}/x_k)} \text{ where } x_k \in Z_q^* \text{ and } 3 \le k \le j+1$$
$$V_{j+1} = v_1 \cdot v_2 \cdots v_{j+1},\ \xi_{j+1} = x_1 \cdot x_2 \cdots x_{j+1}.$$

$\mathcal{S}$ sends m_{j+1} to $\mathcal{U}_{j+1}$

- **Step 3** : $\mathcal{U}_{j+1}$ mounts an off-line dictionary attack on pw_{j+1} with the message m_{j+1}. $\mathcal{U}_{j+1}$ chooses an appropriate password pw'_{j+1} and decrypts m_{j+1} as

$$\mathcal{D}_{pw'_{j+1}}(m_{j+1}) = \{g_1, g_2, ..., g_{j+1}\} \text{ where } g_l \in \mathrm{G} \text{ and } 1 \le l \le j+1$$

$\mathcal{U}_{j+1}$ checks $g_1^\beta = g_2$. This relation leads to an off-line dictionary attack. (1)

Next we design an N-party EKE-U to be secure against off-line dictionary attacks by insider attackers.

2.2 Countermeasure

The main idea to prevent the malicious insider attacks is that we apply an ephemeral session key instead of password to encrypt keying material between server and clients. In the protocol, we use two encryption functions; one is an ideal cipher $\mathcal{E}$ which is a random one-to-one function such that $\mathcal{E}_K : M \to C$, where $|M| = |C|$ and the other function is a symmetric encryption E which has adaptively chosen ciphertext security. $\mathcal{H}_i$ is an ideal hash function such that $\mathcal{H}_i : \{0,1\}^* \to \{0,1\}^l$ for $1 \le i \le 4$. The detail descriptions are as follows.

2.3 Description of Modified N-party EKE-U

In our protocol three types of functions are used. All clients or server contribute to generation of a common session key by using function $\phi_{c,i}$, $\pi_{c,i}$, and $\xi_{s,i}$ for positive integer i. The description of functions are as follows:

$$\phi_{c,i}(\{\alpha_1, .., \alpha_{i-1}, \alpha_i\}, x) = \{\alpha_1^x, .., \alpha_{i-1}^x, \alpha_i, \alpha_i^x\},$$
$$\pi_{c,i}(\{\alpha_1, .., \alpha_i\}) = \{\alpha_1, .., \alpha_{i-1}\},$$
$$\xi_{s,i}(\{\alpha_1, \alpha_2, .., \alpha_i\}, x) = \{\alpha_1^x, \alpha_2^x, .., \alpha_i^x\}.$$

In the up-flow, C_1 first chooses two numbers in Z_q^* randomly, calculates $X_1 = \phi_{c,1}(X_0, x_1) = \{g^{v_1},\ g^{v_1 x_1}\}$, and sends m_1 to C_2, which is an encryption of X_1 with the password pw_1.[2] Upon receiving m_1, C_2 executes a **TF** protocol with server S. In the **TF** protocol, C_2 sends m_1 and $\zeta_{c_2}(= \mathcal{E}_{pw_2}(g^{a_2}))$ to S for a randomly selected value $a_2 \in Z_q^*$. Then S selects a random number v_2, b_2 and calculates $X_1' = \xi_{s,1}(X_1, v_2)$. S also computes $\zeta_{s_2}(= E_{pw_2}(g^{b2}))$, $sk_2(= \mathcal{H}(C_2||S||g^{a_2}||g^{b_2}||g^{a_2 b_2}))$, $\eta_2(=E_{sk_2}(X_1'))$, and $Mac_2 = \mathcal{H}(sk_2||2)$, and then sends $\zeta_{s_2}, \eta_2, Mac_2$ back to C_2. Mac_2 is used for key confirmation of sk_2 on client sides. For a key confirmation on server sides, we can use an additional key confirmation of $Mac_2' = h(sk_2||S)$. This is the end of **TF** protocol.

On receiving $\eta_2 = E_{sk_2}(X_1')$, C_2 first calculates sk_2 by decrypting ζ_{s_2} with password pw_2, and decrypts η_2 to get X_1'. Next C_2 chooses its own random number x_2 and computes $X_2 = \phi_{c,2}(X_1, x_2)$. Finally C_2 sends a ciphertext $m_2 = E_{sk_2}(X_2)$ to the next client C_3. The above process is repeated up to C_{n-2}. The last client C_{n-1} chooses a random number x_{n-1}, and

[2]For $2 \le i \le n-1$, m_i is encrypted with sk_i ephemerally generated between clients and server.

calculates $X_{n-1} = \pi_{c,n-1}(\phi_{c,n-1}(X'_{n-2}, x_{n-1}))$. The function $\pi_{c,n-1}$ only eliminates the last element of $\phi_{c,n-1}(X'_{n-2}, x_{n-1})$. Finally the client C_{n-1} encrypts X_{n-1} with sk_{n-1}, and sends the ciphertext, m_{n-1} to the server S. By using the function $\pi_{c,n-1}$, the protocol does not allow the server to get the last element of $\phi_{c,n-1}(X'_{n-2}, x_{n-1})$, hence the server is not able to compute a session key. We illustrate an example of N-party EKE-U in Figure 3.

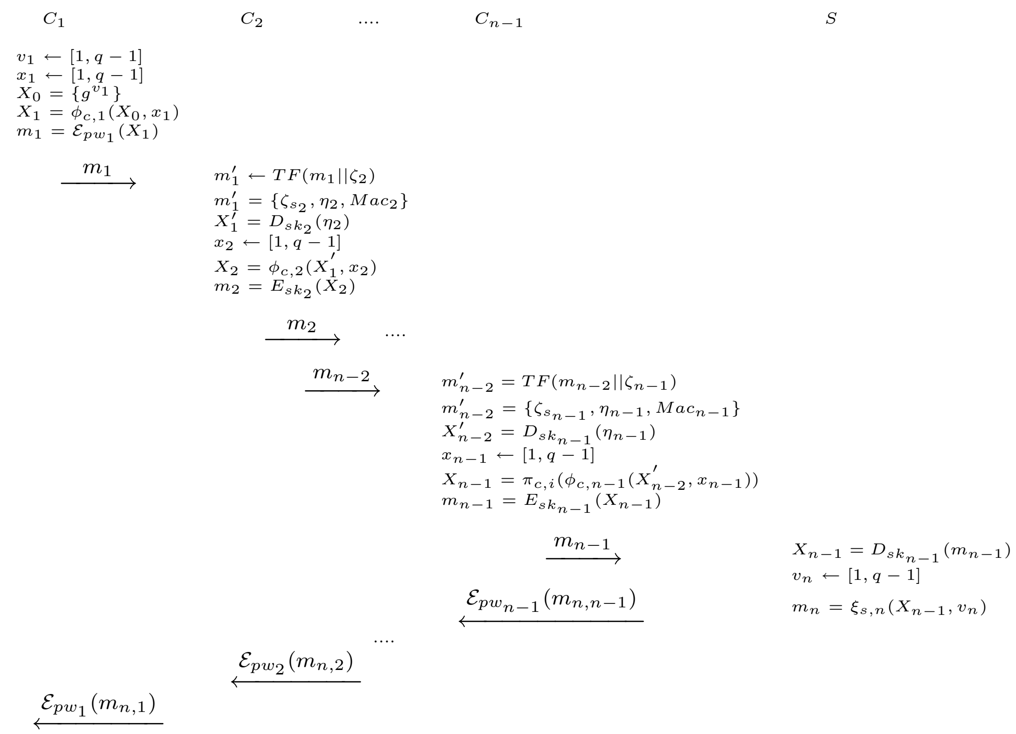

Figure 1: N-party EKE-U

3 Attack on N-party EKE-M and its Countermeasure

3.1 Undetectable On-line Dictionary Attack on N-party EKE-M

Second, Tang presented undetectable on-line guessing attack on N-party EKE-M protocol.[3] The attack is summarized as follows.

- **Step 1** : In the first round, a malicious insider attacker $\mathcal{U}_j$ impersonates $\mathcal{U}_i$, and broadcasts $\mathcal{E}_{pw'_i}(g^{x_i})$ to a server $\mathcal{S}$ by using an appropriate password pw'_i and randomly selected x_i.
- **Step 2** : After finishing the second round, $\mathcal{A}$ can get $\mathcal{E}_{pw_i}(g^{s_i})$ and $m_i = sk_i \oplus N$ sent by S. $\mathcal{U}_j$ computes ephemeral session key $sk'_i = h(sid'||(\mathcal{D}_{pw'_i}(\mathcal{E}_{pw_i}(g^{s_i})))^{x_i})$
- **Step 3** : $\mathcal{U}_j$ checks $N = m_i \oplus sk'_i$ where $\oplus$ denotes exclusive-or operator. This relation leads to an undetectable on-line guessing attack.

[3] For detailed descriptions of N-party EKE-M protocol, please refer to the paper [2].

C_i		S
	$\xrightarrow{m_{i-1}\|\|\zeta_{c_i}}$	For $i = 2$, $X_{i-1} = D_{pw_{i-1}}(m_{i-1})$
		For $i > 2$, $X_{i-1} = D_{sk_{i-1}}(m_{i-1})$
		$v_i, b_i \leftarrow [1, q-1]$
		$X'_{i-1} = \xi_{s,i}(X_{i-1}, v_i)$
		$\zeta_{s_i} = \mathcal{E}_{pw_i}(g^{b_i})$
		$sk_i = \mathcal{H}(C_i\|\|S\|\|g^{a_i}\|\|g^{b_i}\|\|g^{a_i b_i})$
		$Mac_i = \mathcal{H}(sk_i\|\|i)$
	$\xleftarrow{\zeta_{s_i}\|\|\eta_i\|\|Mac_i}$	$\eta_i = E_{sk_i}(X'_{i-1})$

Figure 2: TF protocol

C_1	C_2	C_3	S
$X_1 = \{g^{v_1}, g^{v_1 x_1}\}$ $\xrightarrow{m_1 = \mathcal{E}_{pw_1}(X_1)}$			
	$\zeta_{s_2} = \mathcal{E}_{pw_2}(g^{b_2})$ $sk_2 = \mathcal{H}(C_2\|\|S\|\|g^{a_2}\|\|g^{b_2}\|\|g^{a_2 b_2})$ $\eta_2 = E_{sk_2}(g^{v_1 v_2}, g^{v_1 x_1 v_2})$ $Mac_2 = H(sk_2\|\|2)$ $X_2 = (g^{v_1 v_2 x_2}, g^{v_1 x_1 v_2}, g^{v_1 x_1 v_2 x_2})$ $\xrightarrow{m_2 = E_{sk_2}(X_2)}$		
		$\zeta_3 = \mathcal{E}_{pw_3}(g^{b_3})$ $sk_3 = \mathcal{H}(C_3\|\|S\|\|g^{a_3}\|\|g^{b_3}\|\|g^{a_3 b_3})$ $\eta_3 = E_{sk_3}(g^{v_1 v_2 x_2 v_3 x_3}, g^{v_1 x_1 v_2 v_3}, g^{v_1 x_1 v_2 x_2 v_3})$ $X_3 = \{g^{v_1 v_2 x_2 v_3 x_3}, g^{v_1 x_1 v_2 v_3 x_3}, g^{v_1 x_1 v_2 x_2 v_3}, g^{v_1 x_1 v_2 x_2 v_3 x_3}\}$ $X'_3 = \pi_{c,3}(X_3)$ $\xrightarrow{m_3 = E_{sk_3}(X'_3)}$	$m_3 = \{g^{v_1 v_2 x_2 v_3 x_3 v_4}, g^{v_1 x_1 v_2 v_3 x_3 v_4}, g^{v_1 x_1 v_2 x_2 v_3 v_4}\}$
		$\xleftarrow{\mathcal{E}_{pw_3}(g^{v_1 x_1 v_2 x_2 v_3 v_4})}$	
	$\xleftarrow{\mathcal{E}_{pw_2}(g^{v_1 x_1 v_2 v_3 x_3 v_4})}$		
$\xleftarrow{\mathcal{E}_{pw_1}(g^{v_1 v_2 x_2 v_3 x_3 v_4})}$			

Figure 3: An example of N-party EKE-U (N=4)

3.2 Countermeasure

The main idea to prevent an undetectable on-line guessing attack is that we use an authenticator $\mathcal{H}_2(sk_1||C_1)$ for an ephemeral session key between clients and server. The malicious user can not generate the authenticator since he does not get s_i, hence the server can detect on-line guessing attack. The detailed explanation is as follows.

	S	C_1	C_2	...	C_{n-1}
Round 1	$s_i \leftarrow [1, q-1]$ $\mathcal{E}_{pw_i}(g^{s_i})$	$x_1 \leftarrow [1, q-1]$ $\mathcal{E}_{pw_1}(g^{x_1})$	$x_2 \leftarrow [1, q-1]$ $\mathcal{E}_{pw_2}(g^{x_2})$	... ...	$x_{n-1} \leftarrow [1, q-1]$ $\mathcal{E}_{pw_{n-1}}(g^{x_{n-1}})$
Round 2	$\mathcal{H}_2(sk_i\|\|S)$	$\mathcal{H}_2(sk_1\|\|C_1)$	$\mathcal{H}_2(sk_2\|\|C_2)$	...	$\mathcal{H}_2(sk_{n-1}\|\|C_{n-1})$
Round 3	$N \leftarrow [1, q-1]$ $sk_1 \oplus N\|\|...\|\|sk_{n-1} \oplus N$				

Figure 4: N-party EKE-M

3.3 Description of Modified N-party EKE-M

Let G=$\langle g \rangle$ be cyclic group of prime order q. $sk_i(= \mathcal{H}_1(sid'||g^{x_i s_i}))$ is an ephemeral key generated between S and client C_i in the first round, where $sid' = \mathcal{E}_{pw_1}(g^{x_1})||\mathcal{E}_{pw_2}(g^{x_2})||...||\mathcal{E}_{pw_{n-1}}(g^{x_{n-1}})$. A common group key between clients is $sk = \mathcal{H}_3(SIDS||N)$, where $SIDS = sid'||sk_1 \oplus N||sk_2 \oplus N||...||sk_{n-1} \oplus N$ and N is a random value chosen from Z_q^*.

- In the first round, the single server S sends $\mathcal{E}_{pw_i}(g^{s_i})$ to $n-1$ clients concurrently. Simultaneously each client C_i, $1 \le i \le n-1$, also sends $\mathcal{E}_{pw_i}(g^{x_i})$ to the single-server concurrently in the first round. After the first round finished S and C_i, $1 \le i \le n-1$, share an ephemeral *Diffie-Hellman* key, $sk_i = \mathcal{H}_1(sid'||g^{x_i s_i})$.
- In the second round, server and clients broadcast authenticators $\mathcal{H}(sk_i||S)$ and $\mathcal{H}(sk_i||C_i)$ for sk_i, respectively. S and C_i checks that its authenticator is valid by using sk_i.
- In the third round, S selects a random value N from Z_q^* and hides it by exclusive-or operation with the ephemeral key sk_i. S sends $N \oplus sk_i$ to C_i, $1 \le i \le n-1$, concurrently. After the second round finished all clients can get a random secret N using its sk_i, and generate a common session key, $sk = \mathcal{H}_2(SIDS||N)$.

To add the mutual authentication (key confirmation) to N-party EKE-M protocol, we can use the additional *authenticator* $\mathcal{H}_4(sk||i)$ described in [1].

4 Conclusion

We present countermeasures for off-line and undetectable on-line dictionary attacks by malicious insider attackers.

Acknowledgement

We very thank Ik Rae Jeong and Qiang Tang for valuable discussions.

References

[1] E. Bresson, O. Chevassut, D. Pointcheval, and J. J. Quisquater, “Provably authenticated group diffie-hellman key exchange”, *In proceedings of 8th ACM Conference on Computer and Communications Security*, pp. 255-264, 2001.

[2] J. W. Byun and D. H. Lee, “N-party Encrypted Diffie-Hellman Key Exchange Using Different Passwords”, *In Proc. of ACNS05'*, LNCS Vol. 3531, page 75-90, Springer-Verlag, 2005.

[3] Q. Tang, “Weaknesses in two group Diffie-Hellman Key Exchange Protocols”, Cryptology ePrint Archive 2005/197, 2005.