Insider attack on a password-based group key agreement

Hongtu Li\textsuperscript{a}, Liang Hu\textsuperscript{a}, Wei Yuan\textsuperscript{a}, Hongwei Li\textsuperscript{a}, Jianfeng Chu\textsuperscript{a,*}

\textsuperscript{a}College of computer science and technology, Jilin University, No.2699 Qianjin Street, ChangChun, 130012, China

Abstract

In 2009, Zheng et al. proposed an efficient password-based group key agreement protocol resistant to the dictionary attacks by adding password-authentication services to a non-authenticated multi-party key agreement protocol proposed by Horng. They claimed that the proposed protocol is very efficient since it only requires constant rounds to agree upon a session key, and each user broadcasts a constant number of messages and only requires four exponentiations. Under the Decisional Diffie-Hellman assumption, they shown the proposed protocol is provably secure in both the ideal-cipher model and the random-oracle model. But in this paper, we show that the protocol Zheng et al. proposed is vulnerable to an active insider attack.

© 2011 Published by Elsevier Ltd. Open access under CC BY-NC-ND license.
Selection and/or peer-review under responsibility of [CEIS 2011]

Key words: password-based, authentication, group key agreement, insider attack

1. Introduction

Authenticated group key agreement (GKA) protocols enable a set of users communicating over an insecure, open network to establish a shared secret called session key and furthermore to be guaranteed that they are indeed sharing this session key with each other (i.e., with their intended partners). The session key may be subsequently used to achieve some cryptographic goals such as confidentiality or data integrity. Authenticated GKA protocols allow two or more users to agree upon session key even in the presence of active adversaries. These protocols are designed to deal with the problem to ensure users in the group setting that no other principals aside from members of the group can learn any knowledge about

* Corresponding author: Jianfeng Chu, Tel.: +86 13756492587
E-mail address: chujf@jlu.edu.cn.
the session key. Hence, authenticated GKA protocols can provide a natural secure mechanism for achieving secure multicasting communication in numerous group-oriented scenarios, such as video conferencing, secure replicated database, collaborative applications and distributed computations.

Since the elegant two-party key agreement protocol [1] was proposed by Diffie-Hellman in 1976, many papers have extended this two-party Diffie-Hellman protocol to the group setting. These GKA protocols are classified into two kinds: non-authenticated [2–5,15] and authenticated [7,9,11,12,14]. Burmester and Desmedt [2, 3] proposed a well-known non-authenticated GKA protocol (BD for short) with constant number of rounds under the broadcast channel. Horng [4] focused on the computational efficiency of GKA protocols and proposed a non-authenticated GKA protocol which requires only two rounds of communication. Recently, Katz and Yung [6] proposed a scalable compiler that transforms any GKA protocol into an authenticated one. They also applied their compiler to add authentication services for the BD protocol. However, this solution needs the support provided by the Public Key Infrastructure (PKI) which leads to be more computation overhead.

Password is one of the ideal authentication approaches to agree a session key in the absence of PKI or pre-distributed symmetric keys. There are several works about how to design the PGKA protocol [8–12]. Zheng et al. presented an efficient and secure password-based group key agreement protocol in static group setting according to adding password-authentication services to the protocol proposed by Horng [4]. In their protocol, the legitimate users can share only a low entropy human-memorable password and communicating over an insecure channel controlled by the active adversary, to agree upon a high-entropy session key among themselves. They emphasize that their protocol is provably secure in the random-oracle and ideal-cipher models under the Decisional Diffie-Hellman (DDH) assumption. But in this paper, we point out that Zheng et al.’s password-based group key agreement protocol is not authenticated and easy to be forged or modified. And we give a method that the insider attacker can force all group members.

The remainder of this paper is organized as follows: In Section 2, we review the computational assumptions and Zheng et al.’s password-based group key agreement protocols. In Section 3, we propose the attack method in details. In Section 4, we conclude.

2. Preliminaries

2.1. Decisional Diffie-Hellman (DDH) assumption

Let $G = \langle g \rangle$ be any finite cyclic group of prime order $q$. Informally, the DDH assumption is that it is difficult to distinguish the following real Diffie-Hellman distribution $\Gamma_{\text{real}}$ and random Diffie-Hellman distribution $\Gamma_{\text{rand}}$:

$$
\Gamma_{\text{real}} = \{ g^x, g^y, g^{xy} | x, y \in \mathbb{Z}_q \},
\Gamma_{\text{rand}} = \{ g^x, g^y, g^z | x, y, z \in \mathbb{Z}_q \}
$$

More formally, if we define the advantage function $\text{Adv}_{\text{DDH}}(\mathbb{G}, \mathbb{A})$ as

$$
\text{Adv}_{\text{DDH}}(\mathbb{G}, \mathbb{A}) = \Pr[\mathbb{A}(\cdot) = 1 | Y \subseteq \Gamma_{\text{real}}] - \Pr[\mathbb{A}(\cdot) = 1 | Y \subseteq \Gamma_{\text{rand}}],
$$

we say that the DDH assumption holds in group $G$ if $\text{Adv}_{\text{DDH}}(\mathbb{G}, \mathbb{A})$ is negligible for any probabilistic polynomial time adversary $\mathbb{A}$. We denote $\text{Adv}_{\text{DDH}}(\mathbb{G}, \mathbb{A})$ the maximum value of $\text{Adv}_{\text{DDH}}(\mathbb{G}, \mathbb{A})$ overall adversary $\mathbb{A}$ running in time at most $t$.

2.2. Multi-Decisional Diffie-Hellman (MDDH) assumption

We present another computational assumption based on the Diffie-Hellman assumption. Let us define real Multi Diffie-Hellman distribution $\Pi_{\text{real}}$ and random Multi Diffie-Hellman distribution $\Pi_{\text{rand}}$ of size $n$ as follows:
We define the advantage function as
\[
\text{Adv}_{\text{adv}}^\text{mddh}(A) = \Pr[A(x) = 1 \mid x \in \Pi_{\text{rand}}] - \Pr[A(Y) = 1 \mid Y \in \Pi_{\text{rand}}].
\]

The MDDH assumption holds if \(\text{Adv}_{\text{adv}}^\text{mddh}(A)\) is negligible for any probabilistic polynomial time adversary \(A\). Similar; we denote \(\text{Adv}^*\) the maximum value of \(\text{Adv}_{\text{adv}}^\text{mddh}(A)\) overall adversary \(A\) running in time at most \(t\).

2.3. Review of Zheng et al.’s protocol

At first, we present the following notations are used throughout this paper:

**Nomenclature**

- \(q\): a secure large prime.
- \(p\): a large prime such that \(p=2q+1\).
- \(G_q\): a subgroup of quadratic residues in \(\mathbb{Z}_p\), that is \(G_q = \{i^2 \mid i \in \mathbb{Z}_p^*\}\).
- \(g\): a generator for the subgroup \(G_q\).
- \((E_k, D_k)\): an ideal-cipher system. \(E_k\) is a keyed permutation over \(G_q\) and \(D_k\) is inverse of \(E_k\). \(k\) is the symmetric key.
- \(H\): a hash function for generating the symmetric key.
- \(G\): a hash function for generating the session key.
- \(F\): a hash function for key confirmations.

\(l_H\), \(l_G\) and \(l_F\) denote outputted bit-length of \(H\), \(G\) and \(F\), respectively. Without loss of generality, let \(U=\{u_1, u_2, \ldots, u_n\}\) be the initial set of users that want to generate a session key. Each user \(u_i\) has a specific index \(i\). Note that in the following, the indices are taken in a cycle, e.g.: \(u_i, u_{i+1}\) are the left and right neighbors of \(u_i\) for \(1 \leq i \leq n\), \(x \in \mathbb{Z}_p\) means that element \(x\) is chosen uniformly random in set \(X\).

Suppose \(n\) users share a low-entropy password \(pw\) which is uniformly drawn from a small dictionary of size \(N\), and wish to agree a high-entropy common session key among themselves. Their PGKA protocol is obtained by modifying the non-authenticated GKA protocol of Horng [4] by using password encrypted authentication mechanism. The protocol was described as follows:

Step 1: Each user \(u_i\) (\(1 \leq i \leq n\)) chooses a random nonce \(N_i\) and broadcasts \((u_i, N_i)\). Upon receiving all \((u_j, N_j)\) (\(1 \leq j \leq n, j \neq i\)), \(u_i\) sets session \(S=\{(u_i, N_i) \mid 1 \leq i \leq n\}\).

Step 2: Each user \(u_i\) chooses \(x_i \in \mathbb{Z}_p\), computes and broadcasts \(\hat{y}_i = E_k(x_i)\), where \(y_j = g^{x_j} \mod p\), \(k_i=H(S, i, pw)\).
Step3: Each user \( u_i \) decrypts \( y_{j-1} = D_{k_{\text{un}}}(\tilde{y}_{j-1}) \), \( y_{j+1} = D_{k_{\text{un}}}(\tilde{y}_{j+1}) \), and \( u_i \) computes left key \( z^l_i = R_i \), right key \( z^r_i = y^r_{j-1} \) \( \mod \ p \), where \( R_i \in \mathbb{Z}_p \).

\( u_i \), \( (1<i<n) \) computes left key \( z^l_i = y^l_{j+1} \) \( \mod \ p \), right key \( z^r_i = y^r_{j+1} \) \( \mod \ p \).

\( u_i \) computes left key \( z^l_i = y^l_{j+1} \) \( \mod \ p \), right key \( z^r_i = R_i \), where \( R_i \in \mathbb{Z}_p \).

Then cache user \( u_i \), \( (1<i<n) \) computes and broadcasts \( z_i = z^l_i z^r_i \) \( \mod \ p \). Notes that \( z^l_i = z^r_i \).

Step4: Each user \( u_i \) computes \( K_i = g^{z^l_i + z^r_i + ... + z^l_i z^r_i} \) \( \mod \ p \) exactly using the same approach in the Step3 of Horng’s protocol, then computes and broadcasts his key confirmation \( F_i = F(S, i, \alpha, K_i) \), where \( \alpha = \{\tilde{y}_i, z_i, F_i\} | 1 \leq j \leq n \} \).

Step5: After receiving and checking all key confirmations, user \( u_i \) computes session key as \( sk_i = g(S, \beta, K_i), \beta = \{\tilde{y}_i, z_i, F_i\} | 1 \leq j \leq n \} \).

3. Attack on Zheng et al.’s protocol

Although Zheng et al.’s declared their protocol provide security. In our analysis, we point out the flaw of Zheng et al.’s protocol in insider attack. The attack process is described as follows:

Let \( U = \{u_1, u_2, ..., u_n\} \) be the initial set of users that want to generate a session key. As description in section 2.3, the \( n \) users share a low-entropy password \( \text{pw} \). \( u_i \), \( u_{i+1} \) are the left and right neighbors of \( u_i \) for \( 1 \leq i \leq n \) \( (u_{n+1} = u_1, u_{n+2} = u_2) \). We suppose that user \( u^* \in U \) had participated in a previous session to make a group key. We assume \( u^* \) can control the network of \( u_i \in U \) who he/she want to impersonate.

Step1: Each user \( u_i \) \( (1 \leq i \leq n) \) chooses a random nonce \( N_i \) and broadcasts \( (U_i, N_i) \). Upon receiving all \( (u_j, N_j) \) \( (1 \leq j \neq i \leq n) \), \( U_i \) sets session \( \mathcal{S} = \{u_i(N_i)| 1 \leq i \leq n\} \).

Step2: Each user \( u_i \) chooses \( x_i \in \mathbb{Z}_p \), computes and broadcasts \( \tilde{y}_i = E_k(y_i), \) where \( y_i = g^{x_i} \) \( \mod \ p \), \( k_i = H(S, i, \text{pw}) \). Then \( u^* \in U \) who want to want to impersonate \( u_i \in U \) to the other users intercept and capture the message \( ur \) broadcasted and forge the message as follows:

\( u^* \) picks \( x^* \in \mathbb{Z}_p \), computes \( \tilde{y}^* = E_k(y^*), \) where \( y^* = g^{x^*} \) \( \mod \ p \), \( k_i = H(S, r, \text{pw}); \)

\( u^* \) broadcasts \( \tilde{y}^* \) as he/she was \( u_i \).

Step3: Each user \( u_i \) decrypts \( y_{j-1} = D_{k_{\text{un}}}(\tilde{y}_{j-1}) \), \( y_{j+1} = D_{k_{\text{un}}}(\tilde{y}_{j+1}) \). So user \( u_{r-1} \) and \( u_{r+1} \) decrypt \( y_r = D_{k_{\text{un}}}(\tilde{y}_r) \). User \( u^* \) as \( u_i \) decrypts \( y_{j-1} = D_{k_{\text{un}}}(\tilde{y}_{j-1}) \), \( y_{j+1} = D_{k_{\text{un}}}(\tilde{y}_{j+1}) \). And \( u_i \) computes left key \( z^l_i = R_i \), right key \( z^r_i = y^r_{j-1} \) \( \mod \ p \), where \( R_i \in \mathbb{Z}_p \).

\( u_i \), \( (1<i<n, i \neq r) \) computes left key \( z^l_i = y^l_{j-1} \) \( \mod \ p \), right key \( z^r_i = y^r_{j+1} \) \( \mod \ p \). So \( u^* \) as \( u_i \) computes left key \( z^l_i = y^l_{j-1} \) \( \mod \ p \), right key \( z^r_i = y^r_{j+1} \) \( \mod \ p \).

\( u_i \) computes left key \( z^l_i = y^l_{j-1} \) \( \mod \ p \), right key \( z^r_i = R_i \), where \( R_i \in \mathbb{Z}_p \).

Then each user \( u_i \), \( (1 \leq i \leq n, i \neq r) \) computes and broadcasts \( z_i = z^l_i z^r_i \) \( \mod \ p \). Notes that \( z^l_i = z^r_i \). Here \( z^l_i = z^r_i \) \( \mod \ p \).

Step4: Each user \( u_i \), \( (1 \leq i \leq n, i \neq r) \) computes \( K^* = g^{z^l_i + z^r_i + ... + z^l_i z^r_i} \) \( \mod \ p \) exactly using the same approach in the Step3 of Horng’s protocol, then computes and broadcasts his/her key confirmation \( F^* = F(S, i, \alpha, K^*), \) where \( \alpha = \{\tilde{y}_i, z_i, F_i\} | 1 \leq j \leq n, i \neq r\} \cup \{\tilde{y}^*, z^*, F^*\} \).

Step5: After receiving and checking all key confirmations, user \( u_i \) computes session key as \( sk^*_i = g(S, \beta^*, K^*), \beta^* = \{\tilde{y}_i, z_i, F_i\} | 1 \leq j \leq n, i \neq r\} \cup \{\tilde{y}^*, z^*, F^*\} \).

As in the above attack, any user can be an attacker who can impersonate any one of the users set to the others. As an insider user in Zheng et al.’s password-based group key agreement protocol, the others can’t authenticate him/her, because the \( n \) users share a common password.

4. Conclusion

In this paper, we have shown that Zheng et al.’s password-based key agreement is not as secure as stated by the authors. Our proposed attack compromised Zheng et al.’s protocol, causing the user of the
group to fail to agree upon a common communication key. Thus, the group members cannot communicate together security.

References


