Security properties of two provably secure conference key agreement protocols

Qiang Tang and Chris J. Mitchell
Information Security Group
Royal Holloway, University of London
Egham, Surrey TW20 0EX, UK
{qiang.tang, c.mitchell}@rhul.ac.uk

17th June 2005

Abstract

In this paper we exhibit security vulnerabilities in two authenticated group key agreement schemes based on the group key agreement protocol of Burmester and Desmedt. One scheme was proposed by Burmester and Desmedt, and uses a separate authentication scheme to achieve authentication among the participants. The other was generated using the general protocol compiler of Katz and Yung. We show that not only do they fail to achieve some of the claimed security goals, but they also suffer from other security vulnerabilities.

1 Introduction

Since the pioneering work of Diffie and Hellman [9], key agreement has become a very active and fruitful research area in cryptography. The case of two-party key agreement has been well investigated, and a variety of provably secure schemes have been proposed (e.g., [1, 5, 7]). However, less attention has been given to group key agreement, which enables more than two participants to negotiate a session key. Of especial interest are authenticated group key agreement protocols, designed for use in a hostile environment where communications are over open networks which may be fully controlled by an adversary.

Of the existing group key agreement protocols, a number are based on the idea of extending the two-party Diffie-Hellman protocol [9] to the group setting (e.g., [3, 13, 15, 16, 17, 18]). Among these schemes, the cyclic group key agreement protocol due to Burmester and Desmedt (here referred to as the BD scheme) is particularly efficient; it has been rigorously proved to be secure against a passive adversary [4]. A number of authenticated group key agreement schemes based on the BD scheme have been proposed, including those in [3, 4, 8, 10, 11, 14, 19]. In this paper, we focus on the enhanced BD scheme [4] and a scheme due to Katz and Yung [14], referred to below as the KY scheme.

In the enhanced BD scheme, an interactive zero-knowledge proof scheme is used to achieve authentication among the conference participants, while in the KY scheme a signature mechanism is used to achieve authentication among the conference participants. Both schemes are more efficient than the scheme of Bresson et al. [2], which was the first authenticated group key agreement scheme proved to be secure in a formal model.

The main contribution of this paper lies in analysing the security properties of the BD scheme, and exhibiting security vulnerabilities in the enhanced BD scheme and the KY scheme. The rest of this paper is organised as follows. In section 2, we review three group key agreement schemes, namely the BD scheme, the enhanced BD scheme, and the KY scheme. In section 3, we give our observations on the security of these three schemes. In section 4, we conclude this paper.

2 Review of the target schemes

In all three schemes, the following parameters are made public during the initialisation stage: G is a multiplicative group with large prime order q, and g is a generator of G. We suppose all the potential participants and their identities are from the set $\{(U_1, ID_{U_1}), \cdots, (U_m, ID_{U_m})\}$, where m is a sufficiently large integer and ID_{U_i} $(1 \leq i \leq n \leq m)$ is the identity of U_i.

2.1 Description of the BD scheme

Suppose a set $S = \{U_1, \cdots, U_n\}$ $(n \leq m)$ of users wish to establish a session key; then each user U_i $(1 \leq i \leq n)$ performs the following steps. It should be noted that the indices of users (and values exchanged between users) are taken modulo n.

1. U_i chooses a random s_i $(0 \leq s_i < q)$, and broadcasts $Z_i = g^{s_i}$.
2. After receiving Z_{i-1} and Z_{i+1}, U_i computes and broadcasts X_i:

$$X_i = (Z_{i+1}/Z_{i-1})^{s_i}$$

3. After receiving every X_j $(1 \le j \le n)$, U_i computes the session key K_i as:

$$\begin{aligned} K_i &= (Z_{i-1})^{ns_i} \cdot (X_i)^{n-1} \cdot (X_{i+1})^{n-2} \cdots X_{i+n-2} \\ &= g^{ns_{i-1}s_i} \cdot (\frac{g^{s_is_{i+1}}}{g^{s_{i-1}s_i}})^{n-1} \cdot (\frac{g^{s_{i+1}s_{i+2}}}{g^{s_is_{i+1}}})^{n-2} \cdots \frac{g^{s_{i+n-2}s_{i+n-1}}}{g^{s_{i+n-3}s_{i+n-2}}} \\ &= g^{s_{i-1}s_i+s_is_{i+1}+s_{i+1}s_{i+2}+\cdots+s_{i+n-2}s_{i+n-1}} \\ &= g^{s_1s_2+s_2s_3+s_3s_4+\cdots+s_ns_1} \end{aligned}$$

If all the participants are honest, then all of them will compute the same session key because $K_1 = \cdots = K_n$. In [4], Burmester and Desmedt prove that this scheme is secure against a passive adversary.

2.2 Description of the enhanced BD scheme

The enhanced BD scheme provides partial authentication for the protocol messages by using an authentication scheme which is secure against adaptive chosen text attacks. The authentication scheme and the enhanced BD scheme operate as follows.

2.2.1 The authentication scheme

In the initialisation stage, the system selects four large primes p_2, p_3, q_2, q_3 satisfying $p_2 \le q_3$, $q_2|(p_2-1)$, and $q_3|(p_3-1)$. Let g_2 be a generator of a multiplicative group of order q_2 in $Z^*_{p_2}$, and g_3 be a generator of a multiplicative group of order q_3 in $Z^*_{p_3}$. Each user U_i $(1 \le i \le n)$ in the system publishes his public key $\{\beta_{i2}, \beta_{i3}, \gamma_{i3}\}$, where $\{\beta_{i2} = g_2^{a_{i2}}, \beta_{i3} = g_3^{a_{i3}}, \gamma_{i3} = g_3^{b_{i3}}\}$, and keeps $\{a_{i2}, a_{i3}, b_{i3}\}$ secret as his private key.

Suppose U_i wishes to prove knowledge of z to U_j $(j \ne i)$; the authentication scheme operates as follows. U_i sends z and $\gamma_{i2} = g_2^{b_{i2}}$ to U_j, where b_{i2} is randomly selected $(0 \le b_{i2} \le g_2)$. Simultaneously U_i proves to U_j that he knows the discrete logarithm base g_2 of $\beta_{i2}^z\gamma_{i2}$ and the discrete logarithm base g_3 of $\beta_{i3}^{\gamma_{i2}}\gamma_{i3}$, using the zero-knowledge discrete logarithm proof scheme of Chaum et al. [6], described below. U_j checks that $\gamma_{i2}^{q_2} \equiv 1 \pmod{p_2}$, $g_2^{q_2} \equiv \beta_{i2}^{q_2} \equiv 1 \pmod{p_2}$, $g_3^{q_3} \equiv \beta_{i3}^{q_3} \equiv \gamma_{i3}^{q_3} \pmod{p_3}$, that q_2, q_3 are primes, and that $p_2 \le q_3$. If any of the checks fail, U_j terminates the protocol. Otherwise U_j now believes that U_i knows z.

The Chaum et al. zero knowledge discrete logarithm proof scheme [6] operates as follows. Suppose P is a large prime, and that $\alpha^x \equiv \beta \pmod{P}$. Suppose also that P, α, β are made public and x is a secret of Alice. If Alice wants to prove her knowledge of x to Bob, she performs the following steps.

1. Alice selects T random numbers e_i $(0 \leq e_i < P-1, 1 \leq i \leq T)$. Alice computes and sends $h_i = \alpha^{e_i} \bmod P$ $(1 \leq i \leq T)$ to Bob.

2. Bob chooses and sends T random bits $b_i \in \{0, 1\}$ $(1 \leq i \leq T)$ to Alice.

3. For each bit b_i $(1 \leq i \leq T)$, if $b_i = 0$ Alice sets $s_i = e_i$; otherwise Alice computes $s_i = e_i - e_j \bmod P - 1$, where j is the minimal number for which $b_j = 1$. Finally, Alice sends $(x - e_j) \bmod P - 1$ and s_i $(1 \leq i \leq T)$ to Bob.

4. For each bit i $(1 \leq i \leq T)$, if $b_i = 0$ Bob checks $\alpha^{s_i} = h_i$; otherwise Bob checks that $\alpha^{s_i} = h_i h_j^{-1}$. Then Bob checks $\alpha^{x-e_j} = \beta h_j^{-1}$.

If all the checks succeed, Bob can confirm with a probability of $1 - (\frac{1}{2})^T$ that Alice knows x [6].

Burmester and Desmedt [4] claim that the above scheme is a secure authentication system, i.e., it has the following three security properties:

1. When only a passive adversary is present, U_i can successfully prove his knowledge of z to U_j with an overwhelming probability.

2. If an attacker impersonates U_i, then U_j can detect it with an overwhelming probability.

3. If an active attacker substitutes z with z' $(z \neq z')$, then U_j will reject it with an overwhelming probability.

2.2.2 The enhanced BD scheme

Suppose a set $S = \{U_1, \cdots, U_n\}$ $(n \leq m)$ of users wish to establish a session key; then each user U_i $(1 \leq i \leq n)$ performs the following steps. Note that the indices of users (and values exchanged between users) are taken modulo n.

1. U_i chooses a random s_i $(0 \leq s_i \leq q)$, and computes and broadcasts $Z_i = g^{s_i}$.

2. After receiving Z_{i-1} and Z_{i+1}, U_i proves his knowledge of s_i to U_{i+1}, and verifies U_{i-1}'s knowledge of s_{i-1}.

   If both the proof and the verification succeed, U_i computes and broadcasts X_i:
   $$X_i = (Z_{i+1}/Z_{i-1})^{s_i}$$

3. After receiving X_j $(1 \le j \le n)$, U_i computes the session key K_i:

$$\begin{aligned}
K_i &= (Z_{i-1})^{ns_i} \cdot (X_i)^{n-1} \cdot (X_{i+1})^{n-2} \cdots X_{i+n-2} \\
&= g^{ns_{i-1}s_i} \cdot (\frac{g^{s_i s_{i+1}}}{g^{s_{i-1}s_i}})^{n-1} \cdot (\frac{g^{s_{i+1}s_{i+2}}}{g^{s_i s_{i+1}}})^{n-2} \cdots \frac{g^{s_{i+n-2}s_{i+n-1}}}{g^{s_{i+n-3}s_{i+n-2}}} \\
&= g^{s_{i-1}s_i + s_i s_{i+1} + s_{i+1}s_{i+2} + \cdots + s_{i+n-2}s_{i+n-1}} \\
&= g^{s_1 s_2 + s_2 s_3 + s_3 s_4 + \cdots + s_n s_1}
\end{aligned}$$

If all the participants are honest, then all of them will compute the same session key because $K_1 = \cdots = K_n$.

Burmester and Desmedt [4] claim that the enhanced BD scheme is a secure key agreement scheme, i.e., in a protocol instance it is computationally infeasible for any set of active attackers to compute the same session key as that which is computed by the honest participants.

2.3 Description of the KY scheme

Katz and Yung [14] proposed a general protocol compiler that can transform a group key agreement protocol secure against a passive adversary into an authenticated group key agreement protocol secure against both passive and active adversaries. As an example, Katz and Yung transformed the unauthenticated BD scheme into an authenticated group key agreement protocol. Katz and Yung [14] prove that this protocol is secure against a active adversary, i.e., the advantage of any probabilistic polynomial time (PPT) active adversary is negligible.

The KY scheme [14] requires that, during the initialisation stage, each user U_i $(1 \le i \le m)$ generates a verification/signing key pair (PK_{U_i}, SK_{U_i}) by running $Gen(1^k)$[1], where k is a security parameter. Suppose a set $S = \{U_1, \cdots, U_n\}$ $(n \le m)$ of users wish to establish a session key; then each user U_i $(1 \le i \le n)$ performs the following steps. It should be noted that, as previously, the indices of users (and values exchanged between users) are taken modulo n. Throughout this paper, $||$ represents the string concatenation operator.

1. U_i chooses a random r_i $(0 \le r_i < q)$ and broadcasts ID_{U_i}, 0, and r_i.

2. After receiving the broadcast messages from all other participants, U_i sets $nonce_i = ((ID_{U_1}, r_1), \cdots, (ID_{U_n}, r_n))$ and stores it as part of its state information[2] .

[1] We suppose that $\Sigma = (Gen, Sign, Vrfy)$ is a signature scheme which is strongly unforgeable under adaptive chosen message attack (as defined in [14]).

[2] If all the messages are transported correctly, every user will possess the same state information.

U_i chooses a random number s_i ($0 \leq s_i < q$) and computes $Z_i = g^{s_i}$. Then U_i computes the signature $\sigma_{i1} = Sign_{SK_{U_i}}(1||Z_i||nonce_i)$ and broadcasts ID_{U_i}, 1, Z_i, and σ_{i1}.

3. When U_i receives the message ID_{U_j}, 1, Z_j, and σ_{j1} from user U_j ($1 \leq j \leq n, j \neq i$), he checks that: (1) U_j is an intended participant, (2) 1 is the next expected sequence number for a message from U_j, and (3) $Vrfy_{PK_{U_j}}(1||Z_j||nonce_i, \sigma_{j1}) = 1$, where an output of 1 signifies acceptance. If any of these checks fail, U_i terminates the protocol. Otherwise, U_i computes $X_i = (Z_{i+1}/Z_{i-1})^{s_i}$ and the signature $\sigma_{i2} = Sign_{SK_{U_i}}(2||X_i||nonce_i)$. Then U_i broadcasts ID_{U_i}, 2, X_i, and σ_{i2}.

4. When U_i receives the message ID_{U_j}, 2, X_j, and σ_{j2} from user U_j ($1 \leq j \leq n, j \neq i$), he checks that: (1) U_j is an intended participant, (2) 2 is the next expected sequence number for a message from U_j, and (3) $Vrfy_{PK_{U_j}}(2||X_j||nonce_i, \sigma_{j2}) = 1$. If any of these checks fail, U_i terminates the protocol. Then U_i computes the session key K_i:

$$\begin{aligned} K_i &= (Z_{i-1})^{ns_i} \cdot (X_i)^{n-1} \cdot (X_{i+1})^{n-2} \cdots X_{i+n-2} \\ &= g^{ns_{i-1}s_i} \cdot (\frac{g^{s_i s_{i+1}}}{g^{s_{i-1}s_i}})^{n-1} \cdot (\frac{g^{s_{i+1}s_{i+2}}}{g^{s_i s_{i+1}}})^{n-2} \cdots \frac{g^{s_{i+n-2}s_{i+n-1}}}{g^{s_{i+n-3}s_{i+n-2}}} \\ &= g^{s_{i-1}s_i + s_i s_{i+1} + s_{i+1}s_{i+2} + \cdots + s_{i+n-2}s_{i+n-1}} \\ &= g^{s_1 s_2 + s_2 s_3 + s_3 s_4 + \cdots + s_n s_1} \end{aligned}$$

If all the participants are honest, then all of them will compute the same session key $K = K_1 = \cdots = K_n = g^{s_1 s_2 + s_2 s_3 + \cdots + s_n s_1}$.

Katz and Yung [14] also proposed the following method to achieve key confirmation for the authenticated group key agreement scheme: after computing key K, each player U_i computes $x_i = F_K(ID_{U_i})$, signs x_i, and broadcasts x_i and the corresponding signature, where F represents a pseudo-random function. However, they did not specify how the signature is computed. To facilitate later discussion, we make the following definitions. If the signature is computed as $\sigma_{ij} = Sign_{SK_{U_i}}(x_i)$, where j is the next round number, we call it weak key confirmation. Otherwise, if the signature is computed as $\sigma_{ij} = Sign_{SK_{U_i}}(j||x_i||nonce_i)$, where j is the next round number, we call it strong key confirmation.

3 Properties of the target schemes

In this section we first describe certain security properties of the BD scheme, and then demonstrate security vulnerabilities in both the enhanced BD scheme and the KY scheme.

3.1 Security properties of the BD scheme

In the BD scheme, a malicious participant, say U_j $(1 \leq j \leq n)$, who can manipulate the communications in the network, is able to make any other participant, say U_i $(1 \leq i \leq n,\ i \neq j)$, compute the session key to be any value $K^* \in G$ chosen by U_j.

To achieve this, in the second step U_j intercepts the message X_{i-n+2} and prevents it from reaching U_i. U_j then waits until all the other messages have been received and computes the session key K in the normal way, i.e. as in step 3 of section 2.1. U_j now sends $X'_{i+n-2} = X_{i+n-2} \cdot \frac{K^*}{K}$ to U_i, pretending that it comes from U_{i+n-2}.

Lemma 3.1. *As a result of the above attack, U_i will compute the session key as K^*.*

Proof. This is immediate, since U_i will compute the session key as $K \cdot \frac{X'_{i+n-2}}{X_{i+n-2}} = K^*$, by definition of X'_{i+n-2}. □

In summary, in the BD scheme any participant capable of manipulating the messages received by another participant can completely control the value of the session key obtained by that participant. In the following subsections, we show that this property means that schemes derived from the BD scheme possess certain security vulnerabilities.

3.2 Security vulnerabilities in the enhanced BD scheme

The enhanced BD scheme suffers from the following potential security vulnerabilities.

3.2.1 Man-in-the-middle attack

To mount a man-in-the-middle attack, an active adversary proceeds as follows. In the first step of the protocol the adversary replaces the message Z_{i+1} sent to U_i with $Z'_{i+1} = Z^2_{i-1}$, for every i $(1 \leq i \leq n)$. Then we can prove that the protocol will end successfully and the adversary can compute the session key held by U_i $(1 \leq i \leq n)$.

Lemma 3.2. *Under the above attack, the protocol will end successfully, and the adversary can compute the session key held by U_i for every i $(1 \leq i \leq n)$.*

Proof. Under the attack, it is clear that the protocol will end successfully, because it is only required that U_i $(1 \leq i \leq n)$ proves his knowledge of s_i to U_{i+1} (while the adversary only changes the message that U_{i+1} sends to U_i).

It is also clear that, in the second step, U_i will broadcast $X_i = (Z'_{i+1}/Z_{i-1})^{s_i} = (Z_{i-1})^{s_i}$. Then, after intercepting all the broadcast values X_i $(1 \leq i \leq n)$, the adversary can compute the session key held by U_i as

$$\begin{aligned} K_i &= (Z_{i-1})^{ns_i} \cdot (X_i)^{n-1} \cdot (X_{i+1})^{n-2} \cdots X_{i+n-2} \\ &= (X_i)^n \cdot (X_i)^{n-1} \cdot (X_{i+1})^{n-2} \cdots X_{i+n-2} \end{aligned}$$

which involves only values broadcast by the various recipients. The result follows. □

3.2.2 Insider different key attack

In the enhanced BD scheme, it is only required that U_i $(1 \leq i \leq n)$ proves his knowledge of s_i to U_{i+1}. The authentication requirement can successfully prevent a malicious attacker from impersonating U_i $(1 \leq i \leq n)$ to send a forged message Z_i to the honest participant U_{i+1}. However, a malicious participant, say U_j $(1 \leq j \leq n)$, who can manipulate the communications in the network, is still able to make any other participant, say U_i $(1 \leq i \leq n, i \neq j)$, compute the session key to be any value $K^* \in G$.

In fact, any active outsider attacker can also mount such an attack by manipulating X_i $(1 \leq i \leq n)$ in step 2 of the protocol run, but the attacker cannot obtain any information about the session keys obtained by the legitimate participants.

3.2.3 Outsider impersonation attack

An outsider attacker can also impersonate a valid participant, U_i say, in some circumstances, but the attacker cannot obtain the session key. This attack is based on the following security vulnerability in the Chaum et al. zero-knowledge discrete logarithm proof scheme[3]. The vulnerability arises from the fact that proof scheme does not enable the prover to specify the verifier.

Suppose Alice wishes to prove her knowledge of x to Bob, then an attacker can concurrently impersonate Alice to prove knowledge of x to any other entity, Carol say. The attack can be mounted as follows.

1. Alice selects T random numbers e_i $(0 \leq e_i \leq P-1, 1 \leq i \leq T)$, and computes and sends $h_i = \alpha^{e_i} \bmod P$ $(1 \leq i \leq T)$ to Bob.

   The attacker intercepts the message, prevents it from reaching Bob, and forwards $h_i = \alpha^{e_i} \bmod P$ $(1 \leq i \leq T)$ to Carol pretending to be Alice (i.e. starting a second run of the protocol).

[3]It should be noted that this vulnerability exists not only in this specific scheme, but also exists in all such schemes with only a one-way proof of knowledge.

2. Carol chooses and sends random bits $e_i \in \{0,1\}$ $(1 \le i \le T)$ to the attacker as part of the second protocol run. The attacker then impersonates Bob to forward $e_i \in \{0,1\}$ $(1 \le i \le T)$ to Alice as the second message of the first protocol run.

3. For each bit b_i $(1 \le i \le T)$, if $b_i = 0$ Alice sets $s_i = e_i$; otherwise Alice computes $s_i = e_i - e_j \bmod P - 1$, where j is the minimal number that $b_j = 1$. Alice sends $x - e_j$ and s_i $(1 \le i \le T)$ to Bob as the third message of the first protocol run. The attacker intercepts the message, prevents it from reaching Bob, and forwards it to Carol as the third message of the second protocol run.

4. For each bit i $(1 \le i \le T)$, if $b_i = 0$ Carol checks $\alpha^{s_i} = h_i$; otherwise Carol checks that $\alpha^{s_i} = h_i h_j^{-1}$. Then Carol checks that $\alpha^{x-e_j} = \beta h_j^{-1}$.

It is easy to verify that Carol's checks will succeed and confirm that the attacker knows x. This attack conflicts with the claim made in [4] that the authentication technique is secure against any type of attack.

We now show how to use the above observation to attack the enhanced BD scheme. Suppose the attacker detects that a set $S = \{U_1, \cdots, U_n\}$ of users is starting the key agreement protocol to negotiate a session key (referred to below as the first protocol instance). The attacker impersonates U_i to initiate a second instance of the key agreement protocol among a different set $S' = \{U'_1, U'_2, \ldots, U'_{n'}\}$ of users, where $U'_i = U_i$. In these two protocol instances, the attacker performs the following steps.

1. In the first protocol instance, U_i chooses a random s_i $(0 \le s_i < q)$, and computes and broadcasts $Z_i = g^{s_i}$. The attacker intercepts the messages from both U_{i-1} and U_{i+1} to U_i and prevents them from reaching U_i.

   In the second protocol instance, the attacker impersonates U_i to broadcast $Z_i = g^{s_i}$. Other participants perform as required by the protocol. Suppose the messages sent by U'_{i-1} and U'_{i+1} are Z'_{i-1} and Z'_{i+1}.

   The attacker impersonates U_{i-1} and U_{i+1} to send Z'_{i-1} and Z'_{i+1} to U_i in the first protocol instance.

2. In the first protocol instance, when U_i proves his knowledge of s_i to U_{i+1}, the attacker mounts the above attack against the Chaum et al. zero-knowledge discrete logarithm proof scheme by impersonating U_i to prove his knowledge of s_i to U'_{i+1} in the second protocol instance.

   In the second protocol instance, when U'_{i-1} proves his knowledge of s'_{i-1} to the attacker, the attacker mounts the above attack against the Chaum et al. zero-knowledge discrete logarithm proof scheme by impersonating U'_{i-1} to prove U'_{i-1}'s knowledge of s'_{i-1} to U_i in the first protocol instance.

In the first protocol instance U_i computes and broadcasts X_i as:

$$X_i = (Z'_{i+1}/Z'_{i-1})^{s_i}$$

The attacker intercepts this message and impersonates U_i to broadcast the same message in the second protocol instance.

3. In the second protocol instance, the users U'_j $(1 \leq j \leq n', j \neq i)$ computes the common session key. However, the attacker cannot compute the session key because he does not know s_i.

   The first instance will be terminated because the authentication messages between U_i and U_{i+1} are blocked by the attacker.

In the second protocol instance, the attacker succeeds in impersonating U_i to the members of the set S'.

3.2.4 Insider impersonation attack

In the above attack, suppose that the attacker is a legitimate participant in the second protocol instance, i.e. suppose that the attacker is U'_j in the set S' (U'_j is not required to be a member of the set S). Then U'_j can successfully impersonate U_i in the second protocol instance without any knowledge of the secret key of U_i. In this case, it is clear that U'_j can compute the session key agreed by all the participants of S', since U'_j is a legitimate member of S'.

This attack means that, even if authentication is implemented, a malicious participant can still impersonate another honest participant in a protocol instance.

3.3 Security vulnerabilities in the KY scheme

The KY scheme suffers from the following potential security vulnerabilities.

3.3.1 Insider impersonation attack

We show that, in some circumstances, any $n-2$ $(n \geq 3)$ malicious participants can collude to impersonate another participant. For simplicity of the description, we describe the impersonation attack in the three-party case. First note that, apart from the KY scheme, other group key agreement schemes which use neither a pre-distributed session identifier, as introduced in [5], nor a unique key identifier associated with each session key, as in the Internet Key Exchange protocol [12], also suffer from this vulnerability.

Without loss of generality, we assume that the three participants are $\{U_1, U_2, U_3\}$, and that U_2 is a malicious participant. The attack requires two instances of the protocol to be initiated simultaneously. U_2 mounts the attack as follows.

1. In the first protocol instance, U_1 chooses a random r_1 $(0 \leq r_1 < q)$ and broadcasts ID_{U_1}, 0, and r_1. U_2 chooses a random r_2 $(0 \leq r_2 < q)$ and broadcasts ID_{U_2}, 0, and r_2, and simultaneously blocks all the messages to and from U_3.

   U_2 initiates a second instance of the key agreement protocol among $\{U_1, U_2, U_3\}$, in which U_2 impersonates U_1. We use U_1' to represent the impersonated U_1. U_1' broadcasts ID_{U_1}, 0, and r_1. U_2 broadcasts ID_{U_2}, 0, and r_2. U_3 chooses a random r_3 $(0 \leq r_3 < q)$ and broadcasts ID_{U_3}, 0, and r_3.

   In the first protocol instance, U_2 impersonates U_3 to broadcast ID_{U_3}, 0, and r_3.

2. In both protocol instances all participants would set their state information as $\{(ID_{U_1}, r_1), (ID_{U_2}, r_2), (ID_{U_3}, r_3)\}$.

   In the first protocol instance, U_1 chooses a random s_1 $(0 \leq s_1 < q)$, and then computes and broadcasts ID_{U_1}, 1, Z_1, and σ_{11}, where

   $$Z_1 = g^{s_1},\ \sigma_{11} = Sign_{SK_{U_1}}(1||Z_1||nonce_1).$$

   U_2 chooses a random s_2 $(0 \leq s_2 < q)$, and then computes and broadcasts ID_{U_2}, 1, Z_2, and σ_{21}, where

   $$Z_2 = g^{s_2},\ \sigma_{21} = Sign_{SK_{U_2}}(1||Z_2||nonce_2).$$

   In the second instance, U_1' broadcasts ID_{U_1}, 1, Z_1, and σ_{11}. U_2 broadcasts ID_{U_2}, 1, Z_2, and σ_{21}. U_3 chooses a random s_3 $(0 \leq s_3 < q)$, and then computes and broadcasts ID_{U_3}, 1, Z_3, and σ_{31}, where

   $$Z_3 = g^{s_3},\ \sigma_{31} = Sign_{SK_{U_3}}(1||Z_3||nonce_3).$$

   In the first protocol instance, U_2 impersonates U_3 to broadcast ID_{U_3}, 1, Z_3, and σ_{31}.

3. In the first protocol instance, U_1 computes and broadcasts ID_{U_1}, 2, X_1, and σ_{12}, where

   $$X_1 = (Z_2/Z_3)^{s_1},\ \sigma_{12} = Sign_{SK_{U_1}}(2||X_1||nonce_1).$$

U_2 computes and broadcasts ID_{U_2}, 2, X_2, and σ_{22}, where

$$X_2 = (Z_3/Z_1)^{s_2},\ \sigma_{22} = Sign_{SK_{U_2}}(2||X_2||nonce_2).$$

In the second protocol instance, U_1' broadcasts ID_{U_1}, 2, X_1, and σ_{12}. U_2 broadcasts ID_{U_2}, 2, X_2, and σ_{22}. U_3 computes and broadcasts ID_{U_3}, 2, X_3, and σ_{32}, where

$$X_3 = (Z_1/Z_2)^{s_3},\ \sigma_{32} = Sign_{SK_{U_3}}(2||X_3||nonce_3).$$

In the first protocol instance, U_2 impersonates U_3 to broadcast ID_{U_3}, 2, X_3, and σ_{32}.

4. In both protocol instances, it is easy to verify that all the checks succeed and all participants compute the following session key:

$$K = g^{s_1 s_2 + s_2 s_3 + s_3 s_1}$$

In the first protocol instance, U_2 has succeeded in impersonating U_3; in the second protocol instance, U_2 has succeeded in impersonating U_1.

3.3.2 Insider different key attack under weak confirmation

We show that, even if weak key confirmation is implemented, any $n-2$ malicious participants can still make the other honest participants compute different keys at the end of the protocol.

For simplicity we describe the attack in three-party case. Suppose, in some past successful instance of the KY protocol among $\{U_1, U_2, U_3\}$, the key confirmation message sent by U_3 is $x_3^* = F_{K_3^*}(ID_{U_3})$ and $\sigma_{33}^* = Sign_{SK_{U_3}}(x_3^*)$. U_2 can initiate a new instance of the KY protocol among $\{U_1, U_2, U_3\}$ and mount a different key attack as follows.

1. Each user U_i $(1 \leq i \leq 3)$ begins by choosing a random r_i $(0 \leq r_i < q)$ and broadcasting ID_{U_i}, 0, and r_i.

2. After receiving the initial broadcast messages, U_i $(1 \leq i \leq 3)$ sets $nonce_i = ((ID_{U_1}, r_1), (ID_{U_2}, r_2), (ID_{U_3}, r_3))$ and stores it as part of its state information. Then U_i chooses a random s_i $(0 \leq s_i < q)$, computes $Z_i = g^{s_i}$ and the signature $\sigma_{i1} = Sign_{SK_{U_i}}(1||Z_i||nonce_i)$, and then broadcasts ID_{U_i}, 1, Z_i, and σ_{i1}.

3. When U_i $(1 \leq i \leq 3)$ receives the messages from other participants, he checks the messages as required by the protocol. It is easy to verify that all the checks will succeed.

U_1 computes and then broadcasts ID_{U_1}, 2, X_1, and σ_{12},where

$$X_1 = (Z_2/Z_3)^{s_1},\ \sigma_{12} = Sign_{SK_{U_1}}(2||X_1||nonce_1).$$

U_3 computes and then broadcasts ID_{U_3}, 2, X_3, and σ_{32}, where

$$X_3 = (Z_1/Z_2)^{r_3},\ \sigma_{32} = Sign_{SK_{U_3}}(2||X_3||nonce_3)$$

U_2 computes and sends ID_{U_2}, 2, X_2, and σ_{22} to U_3, where

$$X_2 = (Z_3/Z_1)^{s_2},\ \sigma_{22} = Sign_{SK_{U_2}}(2||X_2||nonce_2)$$

U_2 then waits until all the other messages have been received and computes the session key K in the normal way, i.e. as in step 3 of section 2.1. U_2 now sends ID_{U_2}, 2, X'_2, and σ'_{22} to U_1, where

$$X'_2 = X_2 \cdot \frac{K_3^*}{K},\ \sigma'_{22} = Sign_{SK_{U_2}}(2||X'_2||nonce_2)$$

Lemma 3.3. *As a result of the above steps, U_1 will compute the session key as K_3^*, and U_3 will computes the session key as K.*

Proof. This is immediate, since U_1 will compute the session key as $(Z_3)^{3s_1}(X_1)^2X'_2 = K \cdot \frac{K_3^*}{K} = K_3^*$, and U_3 will compute the session key as $(Z_2)^{3s_3}(X_3)^2X_1 = K$. □

Hence, as a result, U_2 shares the session keys K_3^* and K ($K \neq K_3^*$) with U_1 and U_3 respectively.

4. U_2 intercepts the confirmation messages between U_1 and U_3 and prevents them from reaching their indeed destinations. U_2 computes and sends the confirmation messages $x'_2 = F_{K_3^*}(ID_{U_2})$ and $\sigma'_{23} = Sign_{SK_{U_2}}(x'_2)$ to U_1, and then sends $x_2 = F_K(ID_{U_2})$ and $\sigma_{23} = Sign_{SK_{U_2}}(x_2)$ to U_3. Then U_2 impersonates U_3 to send x_{33}^* and σ_{33}^* to U_1.

   U_2 initiates a second instance of the key agreement protocol among the members of a set S'', which includes U_1 and U_2. In step 3 of the new instance, U_2 manipulates the communications and forces U_1 to compute the session key as K, and then obtains the confirmation message $(x_1 = F_K(ID_{U_1}), \sigma_{13} = Sign_{SK_{U_1}}(x_1))$ from U_1.

   U_2 impersonates U_1 to forward $(x_1 = F_K(ID_{U_1}), \sigma_{13} = Sign_{SK_{U_1}}(x_1))$ to U_3 in the current protocol instance.

5. It is easy to verify that all the key confirmation messages will be checked successfully by the various participants, and the attack will therefore succeed.

3.3.3 Insider impersonation attack with strong confirmation

We show that, even if strong key confirmation is implemented, any $n-2$ ($n \geq 3$) malicious participants can still collude to impersonate another participant. For simplicity, we describe the impersonation attack in the three-party case.

Without loss of generality, we assume that the three participants are $\{U_1, U_2, U_3\}$, and that U_2 is a malicious participant. The attack requires that two protocol instances are simultaneously initiated. U_2 mounts the attack as follows.

1. In the first protocol instance, U_1 chooses a random r_1 ($0 \leq r_1 < q$) and broadcasts ID_{U_1}, 0, and r_1. U_2 chooses a random r_2 ($0 \leq r_2 < q$) and broadcasts ID_{U_2}, 0, and r_2, and simultaneously blocks all the messages to and from U_3.

   U_2 initiates a second instance of the key agreement protocol among U_i ($1 \leq i \leq 3$), in which U_2 impersonates U_1. For simplicity, we use U_1' to represent the impersonated U_1. U_1' broadcasts ID_{U_1}, 0, and r_1. U_2 broadcasts ID_{U_2}, 0, and r_2. U_3 begins by choosing a random r_3 ($0 \leq r_3 < q$) and broadcasts ID_{U_3}, 0, and r_3

   In the first protocol instance, U_2 impersonates U_3 to broadcast ID_{U_3}, 0, and r_3.

2. In both protocol instances all participants would set their state information as $((ID_{U_1}, r_1), (ID_{U_2}, r_2), (ID_{U_3}, r_3))$.

   In the first protocol instance, U_1 chooses a random s_1 ($0 \leq s_1 < q$), and then computes and broadcasts ID_{U_1}, 1, Z_1, and σ_{11}, where

   $$Z_1 = g^{s_1},\ \sigma_{11} = Sign_{SK_{U_1}}(1||Z_1||nonce_1).$$

   U_2 chooses a random s_2 ($0 \leq s_2 < q$), and then computes and broadcasts ID_{U_2}, 1, Z_2, and σ_{21}, where

   $$Z_2 = g^{s_2},\ \sigma_{21} = Sign_{SK_{U_2}}(1||Z_2||nonce_2).$$

   In the second instance, U_1' broadcasts ID_{U_1}, 1, Z_1, and σ_{11}. U_2 broadcasts ID_{U_2}, 1, Z_2, and σ_{21}. U_3 chooses a random s_3 ($0 \leq s_3 < q$), and then computes and broadcasts ID_{U_3}, 1, Z_3, and σ_{31}, where

   $$Z_3 = g^{s_3},\ \sigma_{31} = Sign_{SK_{U_3}}(1||Z_3||nonce_3).$$

   In the first protocol instance, U_2 impersonates U_3 to broadcast ID_{U_3}, 1, Z_3, and σ_{31}.

3. In the first protocol instance, U_1 computes and broadcasts ID_{U_1}, 2, X_1, and σ_{12}, where

$$X_1 = (Z_2/Z_3)^{s_1},\ \sigma_{12} = Sign_{SK_{U_1}}(2||X_1||nonce_1).$$

U_2 computes and broadcasts ID_{U_2}, 2, X_2, and σ_{21}, where

$$X_2 = (Z_3/Z_1)^{s_2},\ \sigma_{21} = Sign_{SK_{U_2}}(2||X_2||nonce_2).$$

In the second protocol instance, U_1' broadcasts ID_{U_1}, 2, X_1, and σ_{12}. U_2 broadcasts ID_{U_2}, 2, X_2, and σ_{21}. U_3 computes and broadcasts ID_{U_3}, 2, X_3, and σ_{32}, where

$$X_3 = (Z_1/Z_2)^{s_2},\ \sigma_{32} = Sign_{SK_{U_3}}(2||X_3||nonce_3).$$

In the first protocol instance, U_2 impersonates U_3 to broadcast ID_{U_3}, 2, X_3, and σ_{32}.

4. In both instances, it is easy to verify that all the checks succeed and all participants compute the following session key:

$$K = g^{s_1 s_2 + s_2 s_3 + s_3 s_1}$$

In the first protocol instance, U_1 computes and broadcasts ID_{U_1}, 3, x_1, and σ_{13}, where

$$x_1 = F_K(ID_{U_1}),\ \sigma_{13} = Sign_{SK_{U_1}}(3||x_1||nonce_1).$$

U_2 computes and broadcasts ID_{U_2}, 3, x_2, and σ_{23}, where

$$x_2 = F_K(ID_{U_2}),\ \sigma_{23} = Sign_{SK_{U_2}}(3||x_2||nonce_2).$$

In the second protocol instance, U_1' broadcasts ID_{U_1}, 3, x_1, and σ_{13}, U_2 broadcasts ID_{U_2}, 3, x_2, and σ_{23}, and U_3 computes and broadcasts ID_{U_3}, 3, x_3, σ_{33}, where

$$x_3 = F_K(ID_{U_3}),\ \sigma_{33} = Sign_{SK_{U_3}}(3||x_3||nonce_3).$$

In the first protocol instance, U_2 impersonates U_3 to broadcast ID_{U_3}, 3, x_3, and σ_{33}.

5. It is easy to verify that all the received key confirmation messages will be checked successfully by the various participants, and thus U_2 will succeed in impersonating U_1 in the second instance. In fact U_2 will also succeed in impersonating U_3 in the first protocol instance.

4 Conclusions

In this paper we have shown that a number of security vulnerabilities exist in both the enhanced BD scheme and the KY scheme. In particular, we have shown that in the enhanced BD scheme the implementation of the authentication scheme does not meet the authentication requirement specified in [4].

Acknowledgements

The authors would like to express their deep appreciation for the valuable comments provided by Kenneth G. Paterson.

References

[1] M. Bellare and P. Rogaway. Entity authentication and key distribution. In D. R. Stinson, editor, *Advances in Cryptology – Crypto '93*, volume 773 of *Lecture Notes in Computer Science*, pages 110–125. Springer-Verlag, 1993.

[2] E. Bresson, O. Chevassut, D. Pointcheval, and J.-J. Quisquater. Provably authenticated group Diffie-Hellman key exchange. In *Proceedings of the 8th ACM Conference on Computer and Communications Security*, pages 255–264. ACM Press, 2001.

[3] M. Burmester and Y. Desmedt. A secure and efficient conference key distribution system. In A. D. Santis, editor, *Advances in Cryptology—EUROCRYPT '94*, volume 950 of *Lecture Notes in Computer Science*, pages 275–286. Springer-Verlag, 1994.

[4] M. Burmester and Y. Desmedt. A secure and efficient conference key distribution system. In A. D. Santis, editor, *Pre–Proceedings of Eurocrypt '94*, pages 279–290, 1994.

[5] R. Canetti and H. Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In B. Pfitzmann, editor, *Advances in Cryptology — Eurocrypt 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 453–474. Springer-Verlag, 2001.

[6] D. Chaum, J.-H. Evertse, J. Graaf, and R. Peralta. Demonstrating possession of a discrete logarithm without revealing it. In A. M. Odlyzko, editor, *Advances in Cryptology—CRYPTO '86*, pages 200–212. Springer-Verlag, 1987.

[7] L. Chen and C. Kudla. Identity based authenticated key agreement protocols from pairings. In *Proc. of the 16th IEEE Computer Security Foundations Workshop — CSFW 2003*, pages 219–233. IEEE Computer Society Press, 2003.

[8] K. Y. Choi, J. Y. Hwang, and D. H. Lee. Efficient ID-based group key agreement with bilinear maps. In F. Bao, R. Deng, and J. Y. Zhou, editors, *Proceedings of the 2004 International Workshop on Practice and Theory in Public Key Cryptography (PKC '04)*, volume 2947 of *Lecture Notes in Computer Science*, pages 130–144. Springer-Verlag, 2004.

[9] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, 1976.

[10] X. J. Du, Y. Wang, J. H. Ge, and Y. M. Wang. ID-based authenticated two round multiparty key agreement. Cryptology ePrint Archive: Report 2003/247, 2003.

[11] X. J. Du, Y. Wang, J. H. Ge, and Y. M. Wang. An improved ID-based authenticated group key agreement scheme. Cryptology ePrint Archive, Report 2003/260, 2003.

[12] D. Harkins and D. Carrel. The Internet Key Exchange (IKE). IETF RFC 2409, 1998.

[13] I. Ingemarsson, D. Tang, and C. Wong. A conference key distribution system. *IEEE Transactions on Information Theory*, 28(5):714–720, 1982.

[14] J. Katz and M. Yung. Scalable protocols for authenticated group key exchange. In D. Boneh, editor, *Advances in Cryptology — Crypto 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 110–125. Springer-Verlag, 2003.

[15] Y. Kim, A. Perrig, and G. Tsudik. Communication-efficient group key agreement. In *Proc. IFIP TC11 16th Annual Working Conference on Information Security*, pages 229–244, 2001.

[16] D. Steer, L. Strawczynski, W. Diffie, and M. Wiener. A secure audio teleconference system. In H. Krawczyk, editor, *Advances in Cryptology — Crypto '98*, volume 403 of *Lecture Notes in Computer Science*, pages 520–528. Springer-Verlag, 1998.

[17] W. Tzeng. A practical and secure-fault-tolerant conferenc-key agreement protocol. In H. Imai and Y. Zheng, editors, *Proceedings of Public Key Cryptography: Third International Workshop on Practice and Theory in Public Key Cryptosystems*, pages 1–13. Springer-Verlag, 2000.

[18] W. Tzeng. A secure fault-tolerant conference-key agreement protocol. *IEEE Transactions on Computers*, 51(4):373–379, 2002.

[19] F. G. Zhang and X. F. Chen. Attacks on two ID-based authenticated group key agreement schemes. Cryptology ePrint Archive, Report 2003/259, 2003.