Security Weakness of Flexible Group Key Exchange with On-Demand Computation of Subgroup Keys

Qingfeng Cheng, Chuangui Ma

Zhengzhou Information Science and Technology Institute,
Zhengzhou, P. R. China
qingfengc2008@sina.com

Abstract. In AFRICACRYPT 2010, Abdalla et al. first proposed a slight modification to the computations steps of the BD protocol, called mBD+P. Then they extended mBD+P protocol into mBD+S protocol. In this paper, we show that both of mBD+P and mBD+S protocols are vulnerable to malicious insiders attack. Further, we propose a simple countermeasure against this attack.

Key words: Group key exchange; Malicious insider attack; Random oracle model; Key confirmation.

1 Introduction

Group key exchange (GKE) enables three or more parties to agree upon a common secret session key in the open network for secure group communication. However, GKE protocols is currently less well understood than the case of two-party key exchange protocols. Many security attributes have so far been ignored for the case of GKE protocols.

In 2009, Manulis proposed flexible GKE protocols [1] utilizing the well-known parallel Diffie-Hellman key exchange (PDHKE) technique in which each party uses the same exponent for the computation of peer-to-peer (p2p) keys with its peers. Further, Manulis investigated possible optimizations of these protocols allowing parties to re-use their exponents to compute both group and p2p keys, and showed that not all such GKE protocols could be optimized, which included the original Burmester-Desmedt (BD) GKE protocol [2].

Recently, Abdalla et al. used the more generalized and flexible approach than Manulis's scheme to propose two GKE protocols: mBD+P and mBD+S [3], which are based on the well-studied BD GKE protocol. The mBD+P protocol is modified for obtaining the secure merge of BD and PDHKE. The mBD+S protocol as the extension of the mBD+P protocol gets the ability to compute an independent session key for any possible subgroup of the initial GKE users. In addition, the authentication procedure in their protocols is similar to the general authentication technique from [4] and both of mBD+P and mBD+S protocols are proven the security in the random oracle model. In this paper, we

will show that their protocols are vulnerable to malicious insider attack. Under our attack, malicious insiders can disrupt establishment of a common group session key among all group members. Furthermore, we improve their protocols and use key confirmation technique to overcome this secure flaw.

The rest of this paper is organized as follows. In Section 2, we briefly review Abdalla et al.'s protocols. In Section 3, we show that their protocols can't resist malicious insiders attack. In Section 4, we propose our improvement to repair this secure flaw. Finally, the conclusions will be given in Section 5.

2 Review of mBD+P and mBD+S Protocols

In this section, we briefly review mBD+P and mBD+S protocols proposed by Abdalla et al. in 2010. In Table 1, we list the abbreviations and notations used in mBD+P and mBD+S protocols. For more details, we refer to [3].

Table 1. The notations

Notations	Description
q	A large prime
τ	Security parameter
G	A cyclic additive group of order q
H_g, H_p, H_s	Random oracles from $\{0,1\}^*$ to $\{0,1\}^\tau$
H	Random oracle from G to $\{0,1\}^\tau$
n	The number of users
$U_1, U_2, ..., U_{n-1}, U_n$	Users
$Sign$	A digital signature scheme
sk_i	Signature private key
pk_i	Verification public key

2.1 mBD+P Protocol

In this subsection, we briefly review the mBD+P protocol, which includes two stages: group stage and p2p stage. On the correctness of key computation and the security analysis of the mBD+P protocol refer to [3].

Group Stage Let the group users be defined by **pid**$=(U_1, ..., U_n)$. In the following description we assume that user indices form a cycle such that $U_0 = U_n$ and $U_{n+1} = U_1$.

[Round 1]. Each U_i computes $y_i = g^{x_i}$ for some random $x_i \in_R Z_q$ and broadcasts (U_i, y_i).
[Round 2]. Each U_i proceeds as follows:

- lets $sid_i = (U_1|y_1, ..., U_n|y_n)$,

- computes $k'_{i-1,i} = y_{i-1}^{x_i}$ and $k'_{i,i+1} = y_{i+1}^{x_i}$,
- $z'_{i-1,i} = H(k'_{i-1,i}, sid_i)$ and $z'_{i,i+1} = H(k'_{i,i+1}, sid_i)$,
- $z_i = z'_{i-1,i} \oplus z'_{i,i+1}$,
- $\sigma_i = Sign(sk_i, (U_i, z_i, sid_i))$,
- broadcasts (U_i, z_i, σ_i).

[Group Key Computation]. Each U_i checks whether $z_1 \oplus ... \oplus z_n = 0$ and whether all received signatures σ_j are valid and aborts if any of these checks fails. Otherwise, U_i proceeds as follows:

- iteratively for each $j = i, ..., i+n-1$, computes $z'_{j,j+1} = z'_{j-1,j} \oplus z_j$
- accepts $k_i = H_g(z'_{1,2}, ..., z'_{n,1}, sid_i)$ as the group session key.

P2P Stage

[P2P Key Computation]. On input any user identity $U_j \in pid_i$ the corresponding user U_i proceeds as follows:

- computes $k'_{i,j} = y_j^{x_i} = g^{x_i x_j}$,
- accepts $k_{i,j} = H_p(k'_{i,j}, U_i|y_i, U_j|y_j)$ as the two-party session key.

2.2 mBD+S Protocol

In this subsection, we briefly review the mBD+S protocol, which also includes two stages: group stage and subgroup stage. Since the group stage of the mBD+S protocol is same as that of the mBD+P protocol, here we omit the details. On the correctness of key computation and the security analysis of the mBD+S protocol refer to [3]. Next, we only introduce the subgroup stage.

Subgroup Stage On input any user identity $spid \subset pid$ the corresponding users perform the following steps. We assume that $spid = (U_1, ..., U_m)$ with $m < n$ and that $U_0 = U_m$ and $U_{m+1} = U_1$.

[Round 1]. Each $U_i \in spid$ proceeds as follows:

- extracts $ssid_i = (U_1|y_1, ..., U_m|y_m)$ from sid_i,
- computes $k'_{i-1,i} = y_{i-1}^{x_i}$ and $k'_{i,i+1} = y_{i+1}^{x_i}$,
- $z'_{i-1,i} = H(k'_{i-1,i}, sid_i)$ and $z'_{i,i+1} = H(k'_{i,i+1}, sid_i)$,
- $z_i = z'_{i-1,i} \oplus z'_{i,i+1}$,
- $\sigma_i = Sign(sk_i, (U_i, z_i, ssid_i))$,
- broadcasts (U_i, z_i, σ_i).

[Subgroup Key Computation]. Each U_i checks whether $z_1 \oplus ... \oplus z_m = 0$ and whether all received signatures σ_j are valid and aborts if any of these checks fails. Otherwise, U_i proceeds as follows:

- iteratively for each $j = i, ..., i+m-1$, computes $z'_{j,j+1} = z'_{j-1,j} \oplus z_j$
- accepts $k_{i,J} = H_s(z'_{1,2}, ..., z'_{m,1}, ssid_i)$ as the subgroup session key.

3 Insider Attack on mBD+P and mBD+S Protocols

In this section, we propose our attack to the group stage of their protocols. Our attack is similar to Lee and Lee's cryptanalysis [5] on Jung's scheme [6]. Under our attack, two malicious insiders can victim a user to agree a different group session key from other users. We note that this attack also can be mounted to the subgroup stage in the similar way.

Suppose that users U_{i-1} and U_{i+1} are two malicious insiders. They are going to deceive U_i into believing that U_i shares a common group session key with other users after execution of the group stage of the mBD+P protocol or the mBD+S protocol, while in fact U_i does not have the common group session key. All group users honestly execute the protocol during setup phase. In the group stage, two malicious insiders U_{i-1} and U_{i+1} try to disrupt the protocol as follows:

[Round 1]. Each U_l (for $1 \le l \le n$) computes $y_l = g^{x_l}$ for some random value $x_l \in_R Z_q$ and broadcasts (U_l, y_l).
[Round 2]. Each U_j (for $1 \le j \ne i-1, i+1 \le n$) proceeds as follows:

- lets $sid_j = (U_1|y_1, ..., U_n|y_n)$,
- computes $k'_{j-1,j} = y_{j-1}^{x_j}$ and $k'_{j,j+1} = y_{j+1}^{x_j}$,
- $z'_{j-1,j} = H(k'_{j-1,j}, sid_j)$ and $z'_{j,j+1} = H(k'_{j,j+1}, sid_j)$,
- $z_j = z'_{j-1,j} \oplus z'_{j,j+1}$,
- $\sigma_j = Sign(sk_j, (U_j, z_j, sid_j))$,
- broadcasts (U_j, z_j, σ_j) (for $1 \le j \ne i-1, i+1 \le n$).

Malicious insider U_{i-1} proceeds as follows:

- lets $sid_{i-1} = (U_1|y_1, ..., U_n|y_n)$,
- computes $k'_{i-2,i-1} = y_{i-2}^{x_{i-1}}$ and $k'_{i-1,i} = y_i^{x_{i-1}}$,
- $z'_{i-2,i-1} = H(k'_{i-2,i-1}, sid_{i-1})$ and $z'_{i-1,i} = H(k'_{i-1,i}, sid_{i-1})$,
- $z_{i-1} = z'_{i-2,i-1} \oplus z'_{i-1,i} \oplus r_M$, where $r_M \in_R Z_q$ chosen by U_{i-1} and U_{i+1}.
- $\sigma_{i-1} = Sign(sk_{i-1}, (U_{i-1}, z_{i-1}, sid_{i-1}))$,
- broadcasts $(U_{i-1}, z_{i-1}, \sigma_{i-1})$.

Malicious insider U_{i+1} proceeds as follows:

- lets $sid_{i+1} = (U_1|y_1, ..., U_n|y_n)$,
- computes $k'_{i,i+1} = y_i^{x_{i+1}}$ and $k'_{i+1,i+2} = y_{i+2}^{x_{i+1}}$,
- $z'_{i,i+1} = H(k'_{i,i+1}, sid_{i+1})$ and $z'_{i+1,i+2} = H(k'_{i+1,i+2}, sid_{i+1})$,
- $z_{i+1} = z'_{i,i+1} \oplus z'_{i+1,i+2} \oplus r_M$, where $r_M \in_R Z_q$ chosen by U_{i-1} and U_{i+1}.
- $\sigma_{i+1} = Sign(sk_{i+1}, (U_{i+1}, z_{i+1}, sid_{i+1}))$
- broadcasts $(U_{i+1}, z_{i+1}, \sigma_{i+1})$.

[Group Key Computation]. Each U_l checks whether $z_1 \oplus ... \oplus z_n = 0$ and whether all received signatures σ_j are valid and aborts if any of these checks fails. Otherwise, all group members except victim U_i proceed as follows:

- iteratively for each $j = l, ..., l+n-1$, computes $z'_{j,j+1} = z'_{j-1,j} \oplus z_j$

- accepts
  $k_l = H_g(z'_{1,2}, ..., z'_{i-2,i-1}, z'_{i-1,i} \oplus r_M, z'_{i,i+1} \oplus r_M, z'_{i+1,i+2}, ..., z'_{n,1}, sid_l)$,
  where $1 \leq l \neq i \leq n$ as the group session key.

U_i proceeds as follows:

- iteratively for each $j = i, ..., i+n-1$, computes $z'_{j,j+1} = z'_{j-1,j} \oplus z_j$
- accepts $k_i =$
  $H_g(z'_{1,2} \oplus r_M, ..., z'_{i-2,i-1} \oplus r_M, z'_{i-1,i}, z'_{i,i+1}, z'_{i+1,i+2} \oplus r_M, ..., z'_{n,1} \oplus r_M, sid_i)$
  as the group session key.

Since H_g is a random oracle, it is obvious that the session key k_i computed by U_i is different from the group session key k_l (for $1 \leq l \neq i \leq n$) computed by other users.

4 Improvement of mBD+P and mBD+S Protocols

In this section, we propose an effective countermeasure against malicious insider attack. The main idea to prevent the malicious insider attack is that we add an additional round for key confirmation to the group stage of the original mBD+P and mBD+S protocols. In the improvement of mBD+P and mBD+S protocols descriptions, we add two random oracles: $H_g^{'}$ is a random oracle from $\{0,1\}^*$ to $\{0,1\}^{2\tau}$ and H_{kc} is a random oracle from $\{0,1\}^*$ to $\{0,1\}^{\tau}$. Next, we describe the details of our improvement.

[**Round 1**]. Each U_i computes $y_i = g^{x_i}$ for some random $x_i \in_R Z_q$ and broadcasts (U_i, y_i).
[**Round 2**]. Each U_i proceeds as follows:

- lets $sid_i = (U_1|y_1, ..., U_n|y_n)$,
- computes $k'_{i-1,i} = y_{i-1}^{x_i}$ and $k'_{i,i+1} = y_{i+1}^{x_i}$,
- $z'_{i-1,i} = H(k'_{i-1,i}, sid_i)$ and $z'_{i,i+1} = H(k'_{i,i+1}, sid_i)$,
- $z_i = z'_{i-1,i} \oplus z'_{i,i+1}$,
- $\sigma_i = Sign(sk_i, (U_i, z_i, sid_i))$,
- broadcasts (U_i, z_i, σ_i).

[**Group Key Computation**]. Each U_i checks whether $z_1 \oplus ... \oplus z_n = 0$ and whether all received signatures σ_j are valid and aborts if any of these checks fails. Otherwise, U_i proceeds as follows:

- iteratively for each $j = i, ..., i+n-1$, computes $z'_{j,j+1} = z'_{j-1,j} \oplus z_j$
- computes $(k_i, k_i^{kc}) = H_g^{'}(z'_{1,2}, ..., z'_{n,1}, sid_i)$.

[**Key Confirmation Message**]. Each U_i proceeds as follows:

- computes

$$M_i = H_{kc}(k_i^{kc}, sid_i),\ \sigma_i^{kc} = Sign(sk_i, (U_i, M_i, sid_i))$$

– broadcasts $(U_i, M_i, \sigma_i^{kc})$.

[Round 3]. Each U_i checks whether $M_i = M_j$ (for $1 \leq j \neq i \leq n$) and whether all received signatures σ_j^{kc} are valid and aborts if any of these checks fails. Otherwise, U_i completes the session by accepting k_i as the common group session key.

With this improvement, all group users can verify whether their group session key are computed in the same key material and find whether there exists malicious insiders. This simple countermeasure is also effective to the subgroup stage of mBD+S protocol.

5 Conclusion

The design of secure GKE protocols has been proved to be a non-trivial task. Many GKE protocols had appeared in the literature that subsequently were proved to be flawed. In this paper, we point out that Abdalla et al.'s protocols cannot satisfy a security goal, which is to make all group users share a common group session key. The group stage and subgroup stage of their protocols suffer from malicious insiders colluding attack. Two malicious insiders can cheat a user into accepting a different session key from other users. Further, we propose an improvement of their protocols with key confirmation to repair this security weakness.

References

1. M. Manulis, Group key exchange enabling on-demand derivation of peer-to-peer keys, The 7th International Conference on Applied Cryptography and Network Security (ACNS 2009), LNCS 5536, pp. 1-19, 2009.
2. M. Burmester and Y. Desmedt, A secure and efficient conference key distribution system, Advances in Cryptology-EUROCRYPT 1994, LNCS 950, pp.275-286, 1994.
3. M. Abdalla, C. Chevalier, M. Manulis, and D. Pointcheval, Flexible group key exchange with on-demand computation of subgroup keys, Advances in Cryptology-AFRICACRYPT 2010, LNCS 6055, pp.351-368, 2010.
4. J. Katz and M. Yung, Scalable protocols for authenticated group key exchange, Advances in Cryptology-CRYPTO 2003, LNCS 2729, pp. 110-125, 2003.
5. S.M. Lee and D. H. Lee, Analysis of an efficient group key agreement protocol, IEEE Comm. Lett., vol.10, no.8, pp. 638-639, 2006.
6. B.E. Jung, An efficient group key agreement Protocol, IEEE Comm. Lett., vol.10, no.2, pp.106-107, 2006.