Two Attacks on Dutta’s Dynamic Group Key Agreement Protocol

Hui Zhang, Chunxiang Xu and Abdur Rashid Sangi

Abstract. Ratna Dutta and Rana Barua proposed a dynamic group key agreement protocol with constant round referred to as DGKA protocol. They claimed that the DGKA protocol is dynamic, efficient and provably secure under DDH assumption. In this paper, we analyze the security of the DGKA protocol and discovered its vulnerable nature towards two attacks. The first attack relates to the fact that this protocol does not satisfy the key independence property which is crucial for dynamic group key agreement protocol. The second one is an impersonation attack which demonstrates that the DGKA protocol is vulnerable to replay attacks.

Keywords：attack, group key agreement, dynamic, impersonation

1 Introduction

A group key agreement protocol allows a group of users to communicate over an untrusted, public network to share a common secret value called a session key. The session key can be later used in other security services providing communication privacy and integrity. Therefore the group key agreement protocol is fundamental for the other security mechanisms in group applications and received particular attention. Based on public key infrastructure, a group key agreement with authentication mechanism [3, 7, 1, 8, 2, 11, 12, 17], allows group users to agree upon a common secret key even in the presence of active adversaries. In a dynamic group key agreement, users can join or leave the group at any time. Such schemes should ensure the freshness of session key while any membership changes, hence the subsequent sessions remain protected from the members who left and the previous sessions remain protected from newly joining members. In recent years, quite a number of dynamic group key agreement protocols [4, 5, 6, 15, 13, 14, 17] have been proposed.

In ISC 2005, Dutta et al. [9, 10] proposed a constant round authenticated group key agreement protocol (referred to as DGKA protocol) in dynamic scenario. They claimed that the DGKA protocol is dynamic and efficient. Compared with the authenticated group key agreement [12] the DGKA protocol requires less communication rounds.

In this paper, however, we discovered that this protocol is vulnerable to two attacks. The first attack relates to the fact that this protocol does not satisfy the key independence property which is crucial for dynamic group key agreement protocol. The second one is an impersonation attack which demonstrates that the authentication of the DGKA protocol is vulnerable to replay attacks.

This paper is organized as follows: In Section 2, we briefly review the DGKA protocol. Two attacks on DGKA protocol are described in Section 3. Finally, our conclusions are given in Section 4.

2 Review of DGKA protocol

This section briefly reviews the DGKA group key agreement protocol [10].

All group members $\{U_1, \cdots, U_n\}$ which will establish a common session key among themselves are logically ordered into a cycle, i.e., the indices are taken modulo n so that user U_0 is U_n and user U_{n+1} is U_1. All mathematical operations are performed in a cyclic group *G* of some large prime order *q* with *g* as a generator. It is assumed that the description of *G* is implicitly known to all users. The protocol also uses a standard digital signature scheme $DSig = (K, S, V)$ for authentication. *K* is the key generation algorithm which generates a signing key sk_i and a verification key pk_i for each user U_i, *S* is the signature generation algorithm and *V* is the signature verification algorithm. The protocol proceeds as follows.

Key Agreement Procedure:

Round 1: Each user U_i randomly chooses a secret value $x_i \in Z_q^*$, computes $X_i = g^{x_i}$ and $\sigma_i = S_{sk_i}(M_i)$ where $M_i = U_i \,|\, 1 \,|\, X_i$, then sends $M_i \,|\, \sigma_i$ to U_{i-1} and U_{i+1} (note that U_0=U_n and U_{n+1}=U_1)

Round 2: Each user U_i, on receiving $M_{i-1} \,|\, \sigma_{i-1}$ from U_{i-1} and $M_{i+1} \,|\, \sigma_{i+1}$ from U_{i+1}, verifies σ_{i-1} on M_{i-1} and σ_{i+1} on M_{i+1} using the verification algorithm *V* and the respective verification keys pk_{i-1}, pk_{i+1}; if verification fails, aborts; else U_i computes the left key $K_i^L = X_{i-1}^{x_i}$, the right key $K_i^R = X_{i+1}^{x_i}$, $Y_i = K_i^R / K_i^L$ and signature $\overline{\sigma}_i = S_{sk_i}(\overline{M}_i)$ where $\overline{M}_i = U_i \,|\, 2 \,|\, Y_i \,|\, d_i$ (d_i is the instance number generated by counter); then sends $\overline{M}_i \,|\, \overline{\sigma}_i$ to the rest of the users.

(Note that $K_i^R = K_{i+1}^L$ for $1 \le i \le n-1$, $K_n^R = K_1^L$ and $K_{i+(n-1)}^R = K_i^L$.)

Key Computation: Each user U_i, on receiving $\overline{M}_j \,|\, \overline{\sigma}_j$ from U_j verifies $\overline{\sigma}_j$ on $\overline{M}_j$ using the verification algorithm *V* and the verification key pk_j ; if verification fails, abort; else, U_i computes $\overline{K}_{i+j}^R = Y_{i+j}\overline{K}_{i+(j-1)}^R$; U_i verifies if $K_{i+(n-1)}^R = \overline{K}_{i+(n-1)}^R$; if verification fails, aborts; else, U_i computes the session key $sk_{U_i}^{d_i} = \overline{K}_1^R \overline{K}_2^R \cdots \overline{K}_n^R$, the seed $x = H(sk_{U_i}^{d_i})$ and stores K_i^L, K_i^R. (*H* is a hash function $H : \{0,1\}^* \to Z_q^*$.)

The session key $sk = g^{x_1x_2 + x_2x_3 + \cdots + x_nx_1}$.

Join: (the set of $\{U_{n+1}, \cdots, U_{n+m}\}$ with secret values $x_{n+1}, \cdots, x_{n+m}$ want to join the group $\{U_1, \cdots, U_n\}$.)

It is assumed that after the *m* users $U_{n+1}, \cdots, U_{n+m}$ joined the group, the new cycle is $U_1, \cdots U_n, U_{n+1}, \cdots, U_{n+m}$, $U_{n+m} = U_0$ and $U_{n+m+1} = U_1$. During the join procedure, users $U_2, \cdots, U_{n-1}$ are considered to be one user *U* with the secret value *x*, and then the new group $\{U_1, U, U_n, U_{n+1}, \cdots, U_{n+m}\}$ executes the Key Agreement Procedure. Let U_2 computes and sends the message on behalf of *U* and the remaining users $U_3, \cdots, U_{n-1}$ just receive the messages sent to U . At the end of the procedure, all the *n*+*m* users are able to reach the new session key $sk' = g^{x_1x + xx_n + x_nx_{n+1} + \cdots + x_{n+m}x_1}$.

Leave:

Suppose $\{U_1, \cdots, U_n\}$ is a set of users with secret values $x_1, \cdots, x_n$ and an execution of Key Agreement Procedure has already been done. Let K_i^L, K_i^R, $1 \le i \le n$ be left and right keys respectively of U_i computed and stored in this session. And suppose the users $U_{l_1}, \cdots, U_{l_m} \in \{U_1, \cdots, U_n\}$ want to leave the group. Then the new user set would be $\{U_1, \cdots, U_{l_1-L}\} \cup \{U_{l_1+R}, \cdots, U_{l_2-L}\} \cup \cdots \cup \{U_{l_m+R}, \cdots, U_n\}$ where U_{l_i-L} and U_{l_i+R} are respectively the closest remaining left and right neighbors of the user which left i.e. $U_{l_i}, 1 \le i \le m$.

Leave Procedure:

Round 1: For each leaving user U_{l_i}, let $j_1 = l_i - L$, $j_2 = l_i + R$;

U_{j_1}, U_{j_2} respectively choose the secrets randomly x_{j_1}, $x_{j_2} \in Z_q^*$, computes $X_{j_1} = g^{x_{j_1}}$, $X_{j_2} = g^{x_{j_2}}$ and $\sigma_{j_1} = S_{sk_{j_1}}(M_{j_1})$, $\sigma_{j_2} = S_{sk_{j_2}}(M_{j_2})$ where $M_{j_1} = U_{j_1} | 1 | X_{j_1} | d_{j_1}$, $M_{j_2} = U_{j_2} | 1 | X_{j_2} | d_{j_2}$

U_{j_1} sends $M_{j_1} | \sigma_{j_1}$ to U_{j_1-1} and U_{j_2};

U_{j_2} sends $M_{j_2} | \sigma_{j_2}$ to U_{j_1} and U_{j_2+1};

Round 2: For each leaving user U_{l_i}, let $j_1 = l_i - L$, $j_2 = l_i + R$;

U_{j_1-1} and U_{j_2} on receiving $M_{j_1} | \sigma_{j_1}$ form U_{j_1} verifies σ_{j_1} on M_{j_1} using the verification key pk_{j_1};

U_{j_1} and U_{j_2+1} on receiving $M_{j_2} | \sigma_{j_2}$ form U_{j_2} verifies σ_{j_2} on M_{j_2} using the verification key pk_{j_2};

If any of these verifications fails, then aborts; else,

U_{j_1} modifies its left key $K_{j_1}^L = X_{j_1-1}^{x_{j_1}}$ and right key $K_{j_1}^R = X_{j_2}^{x_{j_1}}$;

U_{j_2} modifies its left key $K_{j_2}^L = X_{j_1}^{x_{j_2}}$ and right key $K_{j_2}^R = X_{j_2+1}^{x_{j_2}}$;

U_{j_1-1} modifies its right key $K_{j_1-1}^R = X_{j_1}^{x_{j_1-1}}$;

U_{j_2+1} modifies its left key $K_{j_2+1}^L = X_{j_2}^{x_{j_2+1}}$

Then, re-index the $n-m$ users by $V_1, \cdots, V_{n-m}$ and $\{\hat{K}_1^L, \cdots, \hat{K}_{n-m}^L\}$, $\{\hat{K}_1^R, \cdots, \hat{K}_{n-m}^R\}$ respectively be the set of corresponding left and right keys.

Each user V_i computes $Y_i = \hat{K}_i^R / \hat{K}_i^L$ and signature $\hat{\sigma}_i = S_{sk_i}(\hat{M}_i)$ where $\hat{M}_i = V_i | 2 | Y_i | d_i$;

V_i sends $\hat{M}_i | \hat{\sigma}_i$ to the rest of the users in $\{V_1, \cdots, V_{n-m}\}$;

Key Computation: Each user V_i on receiving $\overline{M}_j | \overline{\sigma}_j$ from V_j ($1 \le j \le n-m$, $j \ne i$), verifies $\overline{\sigma}_j$ on $\overline{M}_j$ using the verification algorithm V and the verification key pk_j;

If verification fails, then aborts; else, V_i computes $\overline{K}_{i+1}^R = Y_{i+1}\overline{K}_i^R$;

For each j, $2 \le j \le n-m-1$, V_i computes $\overline{K}_{i+j}^R = Y_{i+j}\overline{K}_{i+(j-1)}^R$;

V_i verifies if $\hat{K}_{i+(n-m-1)}^R = \overline{K}_{i+(n-m-1)}^R$, if verification fails, then aborts; else V_i computes the session key $sk = \overline{K}_1^R \overline{K}_2^R \cdots \overline{K}_{n-m}^R$, the seed $x = H(sk)$ and stores $\hat{K}_i^L$, $\hat{K}_i^R$.

3 The attacks on DGKA protocol

Two attacks on the DGKA protocol are given in this section.

3.1 Attack mounted by leaving user

The DGKA protocol is a dynamic group key agreement protocol and provides mechanisms to process member addition and deletion. However, there are some problems in the leaving mechanism. This attack shows that the DGKA protocol doesn't satisfy the key independence property [16] which encompasses the following requirements:

(1) Old, previously used group keys can not be discovered by new group member(s). In other words, a group member can not have knowledge of the keys used before it joins the group.

(2) New keys are required to remain out of reach from former group members.

Precisely, we find that the DGKA protocol does not meet the second requirement. That is, the leaving user can compute the newly generated group key after the remaining users execute the Leave Procedure.

We firstly choose the simplest scenario to demonstrate this attack. Suppose $P = \{U_1, \cdots, U_n\}$ be the set of n users. They have executed the protocol for group key agreement and obtained the session key *sk*. U_j is a malicious user, whose goal is to compute the new session key $sk^{'}$ after it leaves the group. To be concise, we suppose U_j is the only user who leaves the group and we do not re-index the users during the execution of the leave procedure.

As a malicious user, U_j makes the following preparations for computing the new session key $sk^{'}$. During the key agreement procedure, U_j stored all the right keys $\overline{K}_1^R, \overline{K}_2^R, \cdots, \overline{K}_n^R$ it computed. After U_j leaves the group, the rest n-1 users $P/\{U_j\}$ execute the leave procedure to obtain a new session key. At this moment, U_j eavesdrops the session among $P/\{U_j\}$ and obtains all the information $\hat{M}_i \,|\, \hat{\sigma}_i$ which will be sent out by U_i during Round 2 of the leave procedure ($U_i \in P/\{U_j\}$).

Up to now, U_j has the following information: all the right keys of the key agreement procedure before U_j leaves: $\overline{K}_1^R, \overline{K}_2^R, \cdots, \overline{K}_n^R$; all the Y_i, $Y_i = \hat{K}_i^R \big/ \hat{K}_i^L$, which can be extracted from $\hat{M}_i$ $(1 \le i \le n, i \ne j$, $\hat{K}_i^L$, $\hat{K}_i^R$ are the left keys and right keys corresponding to the leave procedure). According to the DGKA protocol, during the leave procedure only three users $(U_{j-2}, U_{j-1}, U_{j+1})$ have to change their right keys and three users $(U_{j-1}, U_{j+1}, U_{j+2})$ have to change their left keys. Therefore, we have $\overline{K}_i^R = \hat{K}_i^R$ ($i \ne j-2, j-1, j, j+1$;). U_j can compute the right keys of

$U_{j-2}, U_{j-1}, U_{j+1}$ in following ways: $\hat{K}^L_{j-2} = \hat{K}^R_{j-3} = \overline{R}^R_{j-3}$, $\hat{K}^R_{j-2} = Y_{j-2} \cdot \hat{K}^L_{j-2} = \hat{K}^L_{j-1}$, $\hat{K}^R_{j-1} = Y_{j-1} \cdot \hat{K}^L_{j-1} = \hat{K}^L_{j+1}$, $\hat{K}^R_{j+1} = Y_{j+1} \cdot \hat{K}^L_{j+1}$. Then U_j can get all the right keys of the new group users and compute the new session key $sk' = \hat{K}^R_1 \hat{K}^R_2 \cdots \hat{K}^R_{j-1} \hat{K}^R_{j+1} \cdots \hat{K}^R_n$.

In the illustration above, we assume there is only one user which leaves the group. Actually, if there are more users leaving at the same time, any malicious user can mount an attack and obtain the new session key successfully as long as there are more than two adjacent users which keep their secret values unchanged in the leave procedure. So, let more users change their secret values besides the neighbors of the leaving users and make sure that there are no two or more adjacent users whose secret values keep unchanged. Only in this way, this attack can be avoided.

3.2 Replay attack by two malicious users

In [10], the authors said they modified the Katz-Yung [12] technique to achieve authentication in the DGKA protocol. Compared with Katz and Yung's technique, the DGKA protocol does not use nonces as part of the signed message and that's why the DGKA protocol requires only 2 rounds. However, nonces (used in KY authentication technique) are essential to resist replay attacks. Without the nonces in the signed message, the users can not judge whether the message it received is a fresh or a replay one.

After analyzing the DGKA protocol, we find that any two malicious users whose logic indexes are not adjacent in the former execution of the protocol may mount a replay attack in new protocol executions. Through the attack, these two malicious users can make the other honest users believe that they have already gained a session key among the group. However, some of the users actually did not participate in the execution of the protocol but were impersonated by these two malicious users replaying some messages.

Suppose $P = \{U_1, \cdots, U_n\}$ be a set of *n* users who have executed the protocol for group key agreement and obtained the session key. U_i and U_j are two malicious users and $j > i+1$. In addition to normal actions, following preparations should be made by U_i and U_j for the replay attack during the execution of the protocol.

Round 1: Store the secret value x_i and x_j they selected.

Round 2: U_i stores its right key K^R_i, U_j stores its left key K^L_j, one of them stores all the messages $\overline{M}_k \mid \overline{\sigma}_k$ they receive, $i < k < j$

After finishing a regular key agreement, user U_i and U_j can mount a new one in which U_k $(i < k < j)$ are impersonated by some malicious users. These malicious users are in collusion with U_i and U_j or even may be U_i and U_j themselves.

The actions of U_i, U_j and the malicious users are as following during the new group key agreement:

U_i : Round 1: Read the stored secret value x_i, computes $X_i = g^{x_i}$ and $\sigma_i = S_{sk_i}(M_i)$ where $M_i = U_i \mid 1 \mid X_i$, then sends $M_i \mid \sigma_i$ to U_{i-1}.

Round 2: On receiving $M_{i-1} \mid \sigma_{i-1}$ from U_{i-1}, verifies σ_{i-1} on M_{i-1} using the verification algorithm *V* and the respective verification keys pk_{i-1}; if verification fails, aborts; else U_i computes the left key

$K_i^L = X_{i-1}^{x_i}$, read the stored right key K_i^R, computes $Y_i = K_i^R \big/ K_i^L$ and signature $\overline{\sigma}_i = S_{sk_i}(\overline{M}_i)$ where $\overline{M}_i = U_i \mid 2 \mid Y_i \mid d_i$; then sends $\overline{M}_i \mid \overline{\sigma}_i$ to the rest of the users.

Key Computation: Act as normal.

U_j : Round 1: Read the stored secret value x_j, computes $X_j = g^{x_{ji}}$ and $\sigma_j = S_{sk_j}(M_j)$ where $M_j = U_j \mid 1 \mid X_j$, then sends $M_j \mid \sigma_j$ to U_{j+1}.

Round 2: On receiving $M_{j+1} \mid \sigma_{j+1}$ from U_{j+1}, verifies σ_{j+1} on M_{j+1} using the verification algorithm V and the respective verification keys pk_{j+1}; if verification fails, aborts; else U_i computes the right key $K_j^R = X_{j+1}^{x_j}$, read the stored left key K_j^L, computes $Y_j = K_j^R \big/ K_j^L$ and signature $\overline{\sigma}_j = S_{sk_j}(\overline{M}_j)$ where $\overline{M}_j = U_j \mid 2 \mid Y_j \mid d_j$; then sends $\overline{M}_j \mid \overline{\sigma}_j$ to the rest of the users.

Key Computation: Act as normal.

The j-i-1 malicious users who impersonate $U_{i+1}, \cdots U_{j-1}$:

Round 1: Do nothing.

Round 2: Each fake U_k $(i < k < j)$ who have already gotten the message $\overline{M}_k \mid \overline{\sigma}_k$ from U_i or U_j sends $\overline{M}_k \mid \overline{\sigma}_k$ to the rest of the users.

Key Computation: Act as a normal legitimate user.

The remaining legitimate users cannot distinguish a replay attack from a normal key agreement. The only messages they receive from these fake users are $\overline{M}_k \mid \overline{\sigma}_k$ $(i < k < j)$, each $\overline{\sigma}_k$ is definitely a valid signature for $\overline{M}_k$ which is signed by user U_k. However, as the message $\overline{M}_k = U_k \mid 2 \mid Y_k \mid d_k$ does not contain any information to keep it fresh, the honest users can't judge whether $\overline{M}_k$ a replayed message is. To avoid this attack, in our opinions, the mechanism of nonces [12] should be adopted in the protocol.

4. Conclusions

In this paper, we analyzed the security of the dynamic group key agreement protocol proposed by Dutta et al. in ISC 2005 and later published in IEEE Transactions on Information Theory in 2008. We gave two attacks on this protocol and demonstrated the serious flaw in its leave procedure and its vulnerability to replay attack and we also provided some suggestions for revision.

References

[1] R.Barua, R.Dutta, P.Sarkar. Extending Joux Protocol to MultiParty Key Agreement. Indocrypt2003. Also available at http://eprint.iacr.org/2003/062.

[2] C. Boyd and J. M. G. Nieto. Round-Optimal Contributory Conference Key Agreement, Public Key Cryptography, LNCS vol. 2567, Y. Desmedt ed.,

Springer-Verlag, pp. 161-174, 2003.

[3] E. Bresson and D. Catalano. Constant Round Authenticated Group Key Agreement via Distributed Computing. In Proceedings of PKC'04, LNCS 2947, pp. 115-129, 2004.

[4] E. Bresson, O. Chevassut, and D. Pointcheval. Provably Authenticated Group Diffie-Hellman Key Exchange - The Dynamic Case. Advances in Cryptology - Asiacrypt 2001, LNCS vol. 2248, C. Boyded., Springer-Verlag, pp. 290-309, 2001.

[5] E. Bresson, O. Chevassut, and D. Pointcheval. Dynamic Group Diffie-Hellman Key Exchange under Standard Assumptions. Advances in Cryptology - Eurocrypt '02, LNCS 2332, L. Knudsen ed., Springer-Verlag, pp. 321-336, 2002.

[6] E. Bresson, O. Chevassut, A. Essiari and D. Pointcheval. Mutual Authentication and Group Key Agreement for low-power Mobile Devices. Computer Communication, vol. 27(17), pp. 1730-1737, 2004. A preliminary version appeared in Proceedings of the 5th IFIP-TC6/IEEE , MWCN'03.

[7] E. Bresson, O. Chevassut, D. Pointcheval, and J. J. Quisquater. Provably Authenticated Group Diffie-Hellman Key Exchange. Proc. 8th Annual ACM Conference on Computer and Communications Security, ACM, pp. 255-264, 2001.

[8] M. Burmester and Y. Desmedt. A Secure and Efficient Conference Key Distribution System. In A. De Santis, editor, Advances in Cryptology EUROCRYPT '94, Workshop on the theory and Application of Cryptographic Techniques, LNCS 950, pages 275-286, Springer-Verlag, 1995.

[9] R. Dutta, R. Barua. Provably Secure Constant Round Contributory Group Key Agreement in Dynamic Setting. IEEE Transactions on Information Theory (TIT) 54(5):2007-2025, 2008.

[10] R. Dutta, R. Barua. Constant Round Dynamic Group Key Agreement. ISC 2005:74-88, 2005.

[11] R. Dutta, R. Barua and P. Sarkar. Pairing Based Cryptographic Protocols : A Survey. Cryptology ePrint Archive, Report 2004/064, available at http://eprint.iacr.org/2004/064.

[12] J. Katz and M. Yung. Scalable Protocols for Authenticated Group Key Exchange, In Advances in Cryptology - CRYPTO 2003.

[13] Y. Kim, A. Perrig, and G. Tsudik. Simple and Fault-tolerant Key Agreement for Dynamic Collaborative Groups. In S. Jajodia, editor, 7th ACM Conference on Computation and Communication Security, pages 235-244, Athens, Greece, Nov. 2000.

[14] Y. Kim, A. Perrig, and G. Tsudik. Tree based Group Key Agreement. Report 2002/009, http://eprint.iacr.org, 2002.

[15] H. J. Kim, S. M. Lee and D. H. Lee. Constant-Round Authenticated Group Key Exchange for Dynamic Groups. In the Proceedings of Asiacrypt'04, to appear.

[16] M. Steiner, G. Tsudik, M. Waidner. Key Agreement in Dynamic Peer Groups. IEEE Trans. Parallel Distrib. Syst. (TPDS) 11(8):769-780, 2000.

[17] M. Steiner, G. Tsudik, M. Waidner. Diffie-Hellman Key Distribution Extended to Group Communication, ACM Conference on Computation and Communication Security, 1996.