사용자 순서 재조정을 통한 그룹 키 생성 트리 프로토콜

홍성혁*

Re-Ordering of Users in the Group Key Generation Tree Protocol

Sunghyuck Hong*

요 약 인터넷 응용프로그램을 통해 그룹 통신의 사용이 증가하면서 보안의 대한 요구사항인 메시지 무결성, 사용자 인증, 기밀성이 중요시 되고 있다. 보안 요구 사항을 위해서 그룹 통신 시에 암호 키를 생성하여 메시지 기밀성을 유지하는데, 키 생성을 효율적이면서도 안전하게 키 생성 트리를 이용하여 모바일 컴퓨팅 환경의 사용자도 쉽게 키를 생성하도록 사용자 재배치를 통해 키 생성 효율을 증대 시키기 위해 본 연구의 목표가 있다.

주제어 : 키 관리, 프로토콜 디자인, 네트워크 보안, 인증, 그룹 통신, 그룹 동의

Abstract Tree-based Group Diffie-Hellman (TGDH) is one of the efficient group key agreement protocols to generate the GK. TGDH assumes all members have an equal computing power. As one of the characteristics of distributed computing is heterogeneity, the member can be at a workstation, a laptop or even a mobile computer. Therefore, the group member sequence should be reordered in terms of the member's computing power to improve performance. This research proposes a reordering of members in the group key generation tree to enhance the efficiency of the group key generation.

Key Words : Protocol design, network security, authentication, group communication, agreement protocols, group key management.

1. Introduction

Android phones and iPhone are used a lot by users. More than 70% of cell phone users are smart phones which are able to communicate with other smart phone users. Group communications are pervaded over the network such as video conferences and on-line chatting programs, games, and gambling. Security plays an important role in these instances of group communication. According to [11], user authentication processes and key distribution are just at the beginning of the group communication [11]. The CGK generation, on the other hand, takes a relatively long time to complete. For achieving a high level of security, the CGK should be changed after every user joins and leaves so that a former group member has no access to current communications and a new member has no access to previous communications [11] To improve the group communication efficiency, the CGK generation needs to be optimized by the improved key generation algorithm. Accordingly, group key agreement protocol focuses on the CGK generation. The function for generating CGK in the group key agreement is a modular exponentiation. In order to calculate the CGK using modular exponentiations, the adaptation of key trees is needed to reduce the computational overhead. Modular exponentiation is the computationally most expensive operation in TGDH [2]. The number of exponentiations for membership events depends on the number of

*소속 직책: 백석대학교 정보통신학부 조교수
논문접수: 2012년 6월 27일, 1차 수정을 거쳐, 심사완료: 2012년 7월 23일

group members. The algorithm efficiency of TGDH is O ($\log 2^n$), where n is the current number of users, so it is efficient as long as the key tree is perfectly balanced. However, there is overhead to maintain a perfect key tree balance. If it does not consistently maintain balance, it could compute the CGK worse than a normal sequence structure. Therefore, a group key management protocol needs to evaluate group key generation efficiency.

1.1 Goal and Contribution

Secure and efficient key management is a critical issue for the secure group communication [10] and they are big challenges because security and efficiency must be considered at the same time. In this paper, we assume that the secure communication is guaranteed during the group communication, thus this paper's overall goal is focusing on increasing the efficiency of the group key generation.

The major goals in the paper are as follows:

- Design the efficient group key agreement algorithm
- Prove the efficiency of the new algorithm

2. Re-ordering of Users in the group key generation Tree Protocol

According to previous section, TGDH has a major disadvantage when computing the CGK because of the lack of consideration for users' computing powers. In this Section 2, the new protocol is introduced that how new protocol can improve the performance of the CGK generation.

The GC (Group Controller) is required to manage the group key generation tree and control the overall processes to compute the group key. The GC reassigns users to the structure of the tree whenever membership changes.

2.1 The Protocol Definition

The four basic protocols are introduced in this section: join and leave [12].

All protocols follow the common features:

- Only approved members contribute an equal share to the group key. Non authorized members stay while the group key process is being done.
- Whenever membership changes, all members must generate a secret key and a blind key.
- The secret key is never revealed.
- Each member is required to know all blind keys in the entire key tree.

2.2 Join protocol

Suppose that the group size is n. The group members are M_1, M_2, M_3,…, M_{n-1}, M_n (n < 100).

The key generation tree in the group key agreement protocol (GKAP) is organized by the following features. As part of the protocol, the group controller (GC) in the GKAP initiates to build the group key generation tree. The GC is assigned to the root node <l,v> (l=0, v=0). In this case, l is the level of the tree and v-th node at level l in the tree.

When a member joins the group, then GC assigns the member to the right most node, <l+1,v+1>. The GC reassigns the tree structure and repositions his location <l,v> to <l+1,v>. The new member is assigned to <l+1,v+1> = <1,1> (l=0, v=0).

Figure 1 shows an example of new member Mn+1 joining a group where the GC (Mn) performs the following actions:

Step 1: The new member broadcasts a request to join with $T_c(M_{n+1})$.

Step 2: Current members computes [a private key, a bkey] pair and measure $T_c(M_n)$ time for a key pair.

Step 3: The group controller

- updates key tree by adding new member node

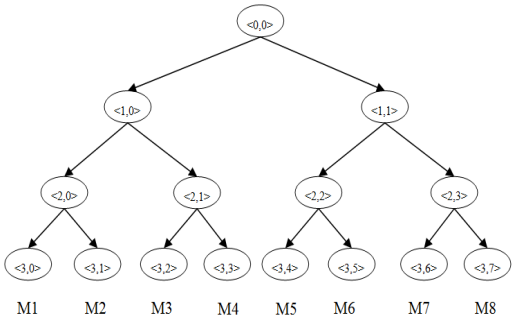
and new intermediate node.

- removes all keys and bkeys on the key-path.
- determines members who will join to next level to compute the group key processes by comparing each member's $T_c(M_n)$.
- broadcast updated tree T.

Step 4: Only members who have been selected by the GC compute the group key using new tree T.

2.3 Leave protocol

As a member leaves the group, the group key must be computed again. If the approved member leaves, the GC will determine the back up member and allow him to join the group key generation processes. The GC will delete the leaving member's leaf node. If the leaving member has a sibling node, then the former sibling node will be promoted to replace the leaving member's parent node. The GC computes all [key, bkey] pairs on the key path up to the root and broadcasts updated key set of bind keys. In this case, only approved members contribute to compute the group key.



〈Fig. 1〉 Key Generation Tree

Step 1: The group controller

- updates key tree by removing the leaving member node and the relevant parent node.
- removes secret keys and blind keys from the leaf node related to the leaving member node.
- Selects the new member for participating in generating the group key if the leaving member is a group key generating member. If the leaving member is not a group key generating member, then the GC doesn't need to select the new group key generating member.
- generates new share and computes all key pairs on the key path.
- broadcasts the updated tree T.

Step 2: Selected members start computing the group key using new tree T.

As shown in Figure 1, key node <0,0> is taking more computation than any other nodes' computations. Figure 1 illustrates the reordering of member in the key tree. Leaf nodes are members (M1, M2, M3, M4, M4, M5, M6, M7, and M8). Key pairs are <M1, M2>, <M3, M4>, <M5, M6>, and <M7, M8>. Each member M_i calculates $g^{ki} \bmod p$ and measures the time to calculate each member's key, and then each member starts to use the Diffie-Hellman key exchange, so they exchange the values M_1 $(g^{K_1} \bmod p)$ and M_2 $(g^{K_2} \bmod p)$ with DES (Data Encryption Standard) [8]. Each key pair gets a sub-group key $k = g^{k1k2} \bmod p$. Each member in a key pair checks the time measurement to calculate $g^{Ki} \bmod p$ and decides which member goes to the next CGK computation processes. Assuming the faster members are M2, M4, M6, and M8, then they should advance to the next key generation process until they get the final CGK. The rest of the members will wait until the processes have been completed. Next subgroup key pairs will be <M2, M4> and <M6, M8>. The faster members calculate subgroup key again. Finally, the fastest member M8 (according to assumption in Figure 1) calculates CGK and distributes it to the rest of the members using Data Encryption Standard (DES).

3. Discussion

The total number of messages is shown in Table 1.

The proposed protocols use a divide and conquer algorithm with a binary tree or queue structure to reduce the complexity of generating a group key. In the first level of the group key generation process, each member must participate in computing a blind key in each membership operation, whereas in the next level only half of the members are authorized to participate in the process.

Table 1. Communication and Computation Cost Comparison

Protocol		Communication		Computation Costs
		Rounds	Messages	Exponentiations
Proposed Protocol	Join	2	2n - 2	3h / 2
	Leave	1	2n - 2	3h / 2
	Partition	1	2n - 2	3h
	Merge	2	2n - 2	3h / 2
GDH	Join	4	n + 3	n + 3
	Leave	1	1	n - 1
	Partition	1	1	n - p
	Merge	m + 3	n + 2m + 1	n + 2m + 1
TGDH	Join	2	3	3h / 2
	Leave	1	1	3h / 2
	Partition	min (log2 p, h)	2h	3h
	Merge	$\log_2 k+1$	2k	3h / 2

Compared to current efficient TGDH protocol, The proposed group key agreement protocol does not need to participate all members. Therefore, unnecessary delays are reduced. The comparison table is followed in Table 1.

4. Conclusion

Group communication is getting more pervaded because of smart phone users. The well-known weakness of smart phone is limited computing power because of a portability. Thus, the distributed computing must consider the variety of users. Mobile computers are getting more popular and network clusters are communicating with conventional servers. Therefore, the group key agreement protocol needs to be reordered for sequencing of key generation depending upon user's computing power in order to unnecessary delays over networks. This reordering will then give users a proper role in generating CGK and as a result, the new algorithm will contribute to maximize the efficiency of the CGK generation.

References

[1] Y. Kim, A. Perrig, and G. Tsudik, (2001), Communication-efficient group key agreement, In 17th International Information Security Conference (IFIP SEC'01)

[2] Y. Kim, A. Perrig, and G. Tsudik, (2004), Tree-based Group Key Agreement, ACM Transaction on Information and System Security

[3] M. Steiner, G. Tsudik, and M.Waidner.(1996), Diffie-Hellman key distribution extended to group communication, ACM Conference on Computer and Communication Security, New Delhi, India. 31 - 37

[4] I., D. Tang, and C. Wong,(1982), A conference key distribution system, IEEE Transactions on Information Theory, 28(5), 714 - 720

[5] W. Diffie and M. E. Hellman.(1976), New directions in cryptography, Transactions on Information Theory, IT-22(6), 644-654.

[6] Y. Kim, A. Perrig, and G. Tsudik,(2000), Simple and fault-tolerant key agreement for dynamic collaborative groups, In S. Jajodia, editor, 7th ACM Conference on Computer and Communications Security, 235 - 244, Athens, Greece, ACM Press.

[7] D. Wallner, E. Harder, and R. Agee,(1997), Key management for multicast: Issues and architecture, Internet-Draft draft-wallner-keyarch -00.txt,

[8] A. Menezes, P. van Oorschot,(1996), and S. Vanstone, Handbook of Applied Cryptography,

CRC Press, 250.

[9] Y. Amir, Y. Kim, C. Nita-Rotaru and G. Tsudik, (2002), On the Performance of Group Key Agreement Protocols, In Submission, Theoretical comparison and actual measurement over LAN/WAN of five group key management protocols integrated with Spread group communication system.

[10] Y. Amir, Y. Kim, C. Nita-Rotaru, J. Schultz, J. Stanton, (2004), Secure Group Communication Using Robust Contributory Key Agreement, IEEE Transaction on parallel and distributed systems, 15(4)

[11] C. Wong, M. Gouda, and S. Lam,(2000), Secure Group Communications Using Key Graphs, IEEE / ACM Transactions on Networking, 8(1)

[12] Y. Kim,(2002), GROUP KEY AGREEMENT: THEORY AND PRACTICE, Ph.D. Dissertation.

[13] Tripathi, S.; Biswas, G.P. (2009), Design of efficient ternary-tree based group key agreement protocol for dynamic groups, Communication Systems and Networks and Workshops, COMSNETS 2009. 1-6, 5-10

홍 성 혁(Sunghyuck Hong)

・received the Ph.D. degree from Texas Tech University in August, 2007 major in Computer Science. Currently, he works at Division of Information and Communication in Baekseok University as an assistant professor. Before he joined Baekseok, he worked at International affairs in Texas Tech University as a senior programmer/analyst, and his jobs were development of ASP.NET web applications and maintenance of PC/Server. He is a member of editorial board in the Journal of Korean Society for Internet Information (KSII) Transactions on Internet and Information Systems. His current research interests include Secure Wireless Sensor Networks, Key Management, and Networks Security.

・E-Mail:shong@bu.ac.kr