A Group Key Management Supporting Individual Batch Rekeying and Fault Tolerance for Secure Multicast*

Hojae Hyun, Keechon Kim, and Sunyoung Han

Konkuk University, 1 Hwayang-dong, Gwangjin-gu, Seoul, Korea
Phone: (02) 450-3537
{hjhyun,kckim,syhan}@kkucc.konkuk.ac.kr

Abstract. One of the major problem areas of secure multicast is group key management [2]. In this paper, we present a novel approach for scalable group key management for secure multicast. Unlike previous work, based on centralized structure, our group key management uses distributed group security manager (GSM). Different GSMs are used to manage each local group, reducing the problem of concentrating the overhead on a single group manager. Using individual batch rekeying, it has advantage of reducing the frequency of rekeying from the size and membership dynamics of the group. Also, GSM supports fault tolerance as well.

1 Introduction

Recently, such Internet applications as multimedia conferences, teleconferences, pay per view and computer-supported collaborative work are expected to be more important. These applications are based upon group communications over Internet.

Multicast communication [1] provides efficient delivery of data from one sender to multiple receivers. It reduces sender transmission overhead, network bandwidth and receiving delay. Also, it supplements drawback of unicast and broadcast. Therefore, multicast communication is an efficient solution to group communication and applied for many of applications. However, multicast communication does not support restricted communication among specified group of users. i.e., any user can receive all data of a multicast group as any host join or leave a multicast group by sending IGMP(Internet Group Management Protocol) [3] messages to their local router. Due to the openness of multicast, the potential for an attack may be greater than in unicast [8].

A typical solution to protect the privacy of group communication is known as encryption algorithm. Encryption algorithm takes messages and encrypts it with a key. If it is applied for group communication, senders encrypt messages using group key that was already distributed to members of the group. Only those who know the group key will be able to decrypt encrypted message. Group key management is an important issue of secure multicast communication [7].

* This research was supported by University IT Research Center Project.

P.M.A. Sloot et al. (Eds.): ICCS 2003, LNCS 2659, pp. 485–494, 2003.

It is required that the group key changes after a new member has joined (so that the new member will not be able to access past multicast data) or a member of group has left (so that the old member will not be able to continue to access multicast data) [9].

In this paper, we describe the design and architecture of a scalable group key management that uses periodic batch rekeying for secure multicast. Our key management is a solution targeted at MBone related secure applications that involve a large number of members and dynamic membership.

To support scalability without regarding group size, our group key management uses distributed Group Security Manager (GSM). Using periodic batch rekeying, it has advantage of reducing the frequency of rekeying from the size and membership dynamics of the group. To support scalable rekeying, we propose *individual batch rekeying*. It is a batch rekeying driven by time interval of each local group independently.

Fault tolerance of key management is the key manager's ability to maintain performance even if service is halted due to system failure or failure due to high traffic. If the member detects fault of GSM, he selects new GSM for fault tolerance..

The remainder of this paper is organized as follows. The next section describes the architecture of proposed system. Section 3 presents an operational overview of system. Section 4 describes performance analysis. Finally, in Sect. 5 we present the conclusion and future research.

2 System Architecture

In the rest of this paper, we use the notation described in Table 1.

Table 1. Notation

GSM : GSM manager mode
member : GSM member mode
MAA : Multicast Address Allocator
Km : individual key of member, Kr : random key
Klg : local group key , Kpg: parent group key
ACL : Access Control List
Klg-ver: Klg revision number
Kpg-ver: Kpg revision number
reg-no: registration number

The proposed group key management offers scalability using distributed hierarchical *GSM* based on *Iolus*[12] that manage dividing big group to several small group as shown in (Fig. 1) so that the changes in a local group does not influence in whole. *MAA* [4][5] allocates (group address, local group address) to group initiator. Group initiator becomes the highest-level GSM. MAA allocates local group address to child GSM. Parent GSM manages local members. Child GSM knows three multicast group addresses (group address, parent group address, local group address).

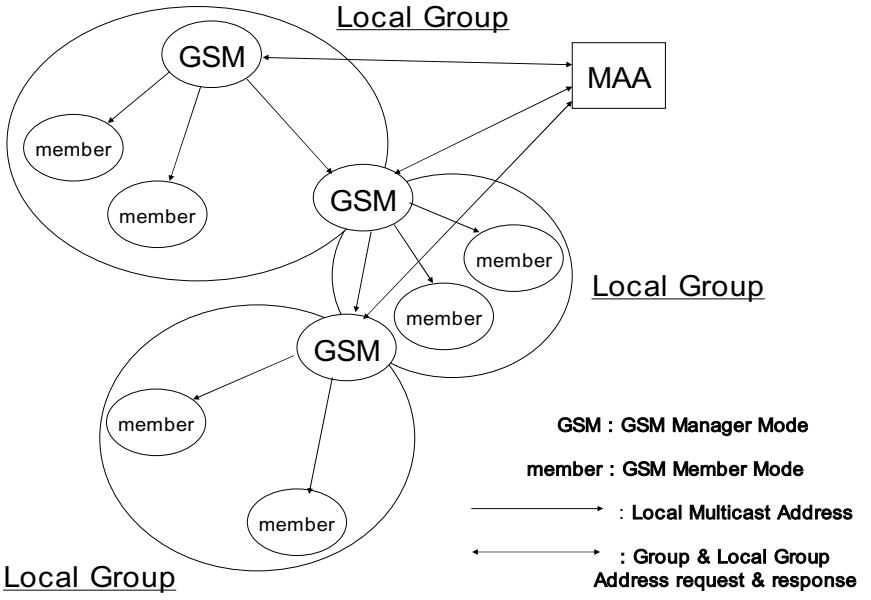

Fig. 1. GSM system architecture

2.1 GSM Component

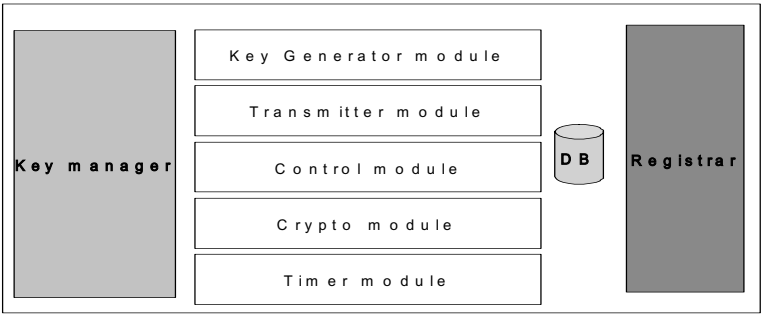

Fig. 2. GSM component

GSM is composed of *key manager*, *registrar*, *DB* and *5 function modules* (Fig. 2). One GSM in a local group collects member (or other GSM)'s requests during a *rekey interval* and processes the requests in a batch.
If a GSM does not exist in local group, joining member converts

– GSM manager mode: sending *alive alarm* message during *alarm interval timer* to *member.*

Otherwise, joining member acts

– GSM member mode: if member's timer for a *alive alarm* expires, he detects fault of local GSM (or parent GSM).

Key manager action

– *Key generator module* generates local group key (Klg).
– Key manager searches individual key(Km) of member from *DB*.
– *Crypto module* encrypts Klg with Km.
– *Transmitter module* multicasts {Klg}Km to members if current rekey expired. Where {Klg}Km denotes key Klg encrypted with key Km.

Registrar action

– Upon receiving request of each member, Registrar checks ACL and decides whether to grant or deny the request. If a request granted, registrar generates individual key and response message including local group address, parent group address and individual key through secure unicast channel. Registrar stores individual key of member in DB. GSM becomes a bottleneck when the member registration rate is high at once. To solve this problem, *registrar* can be added or removed dynamically.

GSM's DB is composed of *state, ACL*, *group ID, parent group ID, local group ID, group address, parent group address, local group address, parent group key, local group key, Kpg-ver, Klg-ver, PG-rekey interval, LG-rekey interval, member list, individual keys of local group, individual key for parent group, registration number of local group* and *registration number of parent group*.

When a member (or child GSM) is registered in local GSM (or parent GSM), *registration number* is stored in DB.

Key generator module action

– Random value should be hashed to generate individual or local group key of member (or child). Hash function [21] is a transformation that takes an input *m* and returns a fixed-size string. A hash function H is said to be one-way if it is hard to invert. Because given a hash value X hard to invert, it is computationally infeasible to find some input Y.

$$X = H(Y) \quad X: \text{key}, Y: \text{random value}, H: \text{encryption function} \tag{1}$$

The choice of the encryption function, H is dictated by the security requirements of the application. Any function such as DES, DES3, MD5 and IDEA can be used.

Transmitter module action

– *Parent group transmitter* multicast message to parent group encrypted with parent group key. Only child GSM use parent group transmitter.
– *Local group transmitter* multicasts messages (key update, alive alarm, data) to local group.

Control module generates a request message (*find_GSM, join, leave, NAK*) of member.

Crypto module encrypts (or decrypts) message (or encrypted message) of group with appropriate key.

Timer module cans startup (or reset) timers.
- *rekey interval timer* in GSM measures lifetime of valid key.
- *alive alarm timer* detects fault of GSM.
- *wait timer* in member detects loss of key update message.

3 Operational Overview of System

3.1 Group Initiation

Group initiator who wishes to create multicast group receives (*group address, local group address*) from MAA. The initiator can use this information to advertise the group session using the Session Announcement Protocol (SAP) [6]. Firstly, initiator acts as the highest-level GSM.

3.2 Member Join

A member can find local GSM using the Expanding Ring Search (ERS) mechanism [17]. He multicasts a *FIND_localGSM* packet with a small Time To Live (TTL) value to *group address*. TTL value defines a range for local group.

- GSM manager mode: After a moment, if there was no response, the member takes the place of local GSM. He multicasts a *FIND_parentGSM* to group address. TTL value defines a range greater than local group. if the member receives *REPLY_FIND_parentGSM* packet, He checks it same email and get *GSM IP address* from it. He sends *join* request to parent GSM and receives message that is composed of *ACL*, *parent group address*, *individual key for parent group, registration number of parent group, PG-rekey interval* from parent GSM through secure unicast channel. He joins parent group. After current rekey interval, member receives *key update message* from parent GSM through multicast channel and decrypts it and stores *Kpg* in DB.

- GSM member mode: if the member receives *REPLY_FIND_localGSM* packet, He checks it same email and get *GSM IP address* from it. He sends *join* request to local GSM and receives message that is composed of (*parent group address:* if parent GSM exist), *local group address*, *individual key for local group, registration number of local group, LG-rekey interval* from local GSM through secure unicast channel. He joins local group. After current rekey interval, member receives *key update message* from local GSM through multicast channel and decrypts it and stores *Klg* in DB.

3.3 GSM Action

GSM generates new group. Upon receiving *find_GSM (local, parent)* request, he responses message including *GSM IP address* and *email of requester*.

He acts in a network that supports multicast and authenticates each member using an authentication protocol, such as SSL 3.0 [15]. He receives member's request and responses through secure unicast channel (SSL 3.0).

After member joins, he starts rekey interval timer. After current rekey interval, he multicasts key update message to member. If receiving leave request of member, registrar deletes member info from DB.

3.4 Key Update

GSM increases *Klg-ver* of key update message step 1. *Add-no* field includes the smallest *reg-no* (except for leaver's *reg-no*) of new member during current rekey interval. GSM updates the *local group key* after current rekey interval. He distributes the new *local group key* to members in local group. We use *key update message* (Fig. 3) by group-oriented rekeying strategy [11].

HDR	Klg-ver	Add-no	{Klg'} Klg	{Klg'} Km1	….	{Klg'} Kmn

Fig. 3. Key update message

If a member receives *key update message* then he checks updated key by *Klg-ver*. New member uses (*reg-no – add-no* + 5)th field to find *local group key* in the *key update message*. If no member leaves, only the remaining member that has *reg-no* smaller than *add-no* can update their *local group key* by decrypting *4th field* in the *key update message*. Otherwise, remaining (or new) members use (reg-no + 4)th field to find *local group key* in the *key update message.* In this case, two fields(*Add-no, {Klg'}Klg)* are not used.

3.5 Leave & Fault Tolerance

If a member wants to leave, he sends a *leave* request to the local GSM. If the GSM receives it, he deletes information concerning the member. When GSM receives a join request after member leaves, it sends leaver's *reg-no* to new member for reusing empty spaces in DB, key update message.

GSM periodically sends *alive alarm* message including *parent group address, member list* for notifying own alive to local members.
(Fig 4) describes a pseudo-code about member's fault detection and tolerance.

```
while(alive alarm timer not expire){
 receive message from local GSM
if (message == alive alarm) reset alive alarm timer
time is increased automatically synchronized with system clock }
get remaining member's info of  low reg-no on member list
if ( member's info of low reg-no == my info) set mode = GSM manager
```

Fig. 4. GSM's fault detection and tolerance

If the child GSM (or parent GSM) leaves from group then he checks remaining member of local group. If the member exists then they announce the member that has lower reg-no of member list as a new GSM. New GSM joins parent GSM through parent group address and takes place of fault GSM.

If GSM wants to expel a member, it processes *key update* and *leave* immediately to disallow the leaving member's continued participation in the local group. Member (or child GSM) resets *wait timer* to 0. GSM resets *rekey interval timer* to 0.

3.6 Retransmission Key Update Message

Each member starts up *wait timer* to recognize problem of lost *key update message*. Member decides *wait timer* by average time of receiving *key update message* from local GSM. If *wait timer* has expired, member sends *NAK* packet(Fig 5) with lost *Klg-ver* through multicast channel [13], [14]. If member receives *NAK packet*, he has two methods. One, if the member has lost same key update message then he resets *wait timer* and does not send NAK packet to avoid *NAK-implosion*. Two, if the member has receivd the *key update message* correctly then he multicasts the *key update message*.

If GSM receives NAK packet, he multicasts the key update message. otherwise, he deletes the *key update message* from buffer.

```
Struct Nak_hdr {
 double packetID;
 double Message ID;
 double Nak_no;
 double Length;
 double Klg-ver;} /* lost key update message */
```

Fig. 5. NAK header

3.7 Data Distribution

A member generates *Kr* and encrypts data with it. He encrypts it with *Klg* and multicasts *{data}Kr , {Kr}Klg* to local group. Upon receiving {Kr}Klg, child GSM decrypts it and encrypts it with *Kpg*. The child GSM multicasts *{data}Kr, {Kr}Kpg* to parent group through *parent group transmitter*.

Decrypting and re-encrypting the whole data is reduced to simply decrypting and re-encrypting *Kr* to reduce the overhead of GSM.

4 Performance Analysis

In order to analyze the performance of our system, we implemented a simple program that uses our key management. The program consists of the GSM and client application. We evaluated performance tradeoffs between our individual batch rekeying and individual rekeying of Iolus.

All tests were performed on a Linux kernel 2.4.18 running on a 500Mhz Pentium II. We assumed that interval of rekey was 20sec and number of remaining member was 30. We measured processing performance between GSA and GSM on member's join request increase from 0 to 60sec step 4 in a local group. We used DES3 as encryption algorithm and 128bit key as local group key.

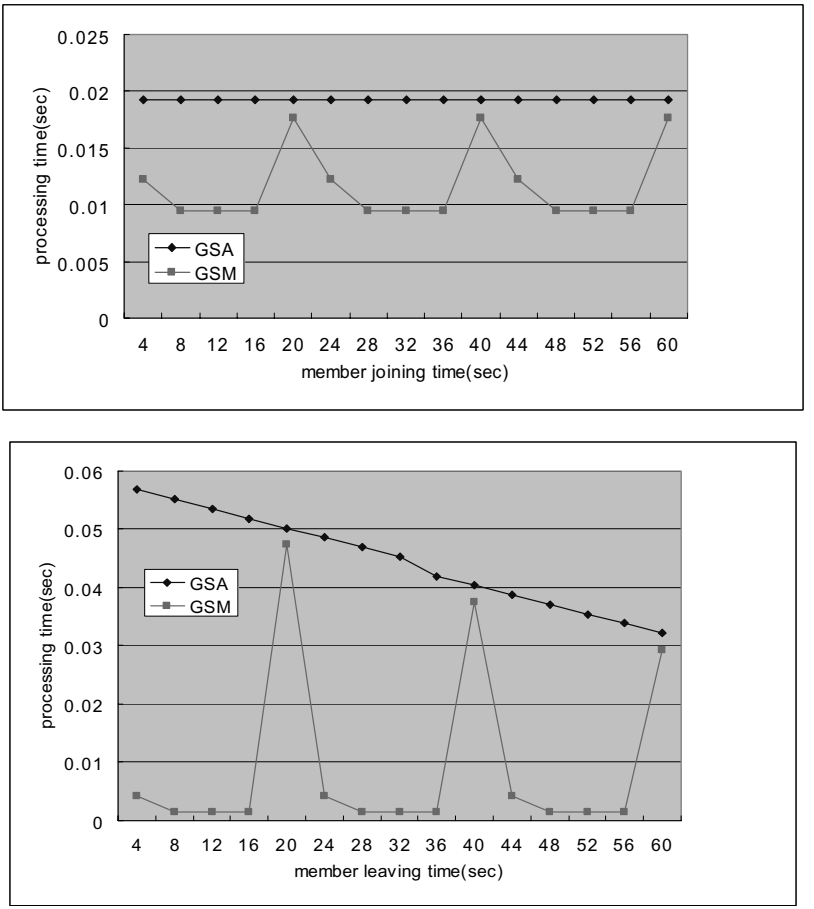

Fig. 6. Processing time of GSM and GSA for member join & leave

We show the processing time of local group manager per join (or leave) request for different rekeying policy (Fig. 6). Frequency of rekeying would become a bottleneck for local group manager. The advantage of our system is to collect requests during a rekey interval in a local group and update key in batch. The variable of performance is membership dynamics during a rekey interval.

5 Conclusions and Future Work

In this paper, we have proposed a novel approach that has distributed hierarchical GSM for secure multicast. A large multicast group is split into smaller local groups and each GSM is used to manage each local group. It minimized problem of concentrating the overhead on a GSM. Thus, exposure of key affects a local group are not reflected to other groups. Also, if a GSM fails, only its local group is affected.

Using individual batch rekeying, it has advantage of less number of rekeying from dynamic membership of the local group. In order to guarantee reliability of individual batch rekeying, our system supports retransmission mechanism.

We compare the performance of our system with Iolus. As a result, our system reduces the overhead of group manager through collects request during the rekey interval and batching key update.

Our key management increases the performance of data transmission through multithread of transmission and crypto module. Also, it supports fault tolerance of GSM for fully distributed structure.

Future works includes research about dynamic GSM for load balancing and IPv6 deployment.

References

1. S.E. Deering.: Host Extensions for IP Multicasting. RFC1112, August 1989
2. D. Wallner, E. Harder, R. Agee.: Key management for multicast: Issues and architectures. RFC 2627, June 1999
3. W. Fenner.: Internet Group Management Protocol, version2. RFC2236, November 1997.
4. D. Thaler, M. Handley, D. Estrin.: The Internet Multicast Address Allocation Architecture. RFC 2908, September 2000
5. P. Radoslavov, D. Estrin, R. Govindan, M. Handley, S. Kumar, D. Thaler: The Multicast Address-Set Claim (MASC) Protocol. RFC 2909, September 2000
6. M. Handley, C. Perkins, E. Whelan.: Session Announcement Protocol. RFC2974, October 2000
7. T. Hardjono, B. Cain, N. Doraswamy.: A Framework for Group Key Management for Multicast Security. Internet Draft, draft-ietf-ipsec-gkmframework-03.txt, August 2000
8. T. Hardjono, R. Canetti, M. Baugher, P. Dinsmore.: Secure IP Multicast: Problem Areas, Framework and Building Blocks. Internet Draft, draft-irtf-smug-framework-01.txt, September 2000
9. H. Harney, E. Harder.: Multicast Security Management Protocol (MSMP) Requirements and Policy. Internet Draft, draft-harney-sparta-msmp-sec-00.txt, March, 1999
10. T. Hardjono, B. Cain, I. Monga.: Intra-Domain Group Key Management Protocol. Internet Draft, draft-irtf-smug-intragkm-00.txt, September 2000

11. Chung Kei Wong, Mohamed Gouda, Simon S. Lam.: Secure group Communication Using Key Graphs. In Proceedings of ACM SIGCOMM '98, Vancover, B.C. September 1998
12. Mittra.: Iolus: A Framework for Scalable Secure Multicasting. In ACM SIGCOMM,volume 27,4 of Computer Communication Review, pages 277–288, New York, September 1997
13. S. Floyd, V. Jacobson, C. G. Liu, S. McCanne, L. Zhang.: A reliable multicast framework for light-weight session and application level framing. IEEE/ACM Transaction on Networking, 5(6):784–803, December 1997
14. D. Towsley, J. Kurose, S. Pingali.: A comparison of sender-initiated reliable multicast and receiver-initiated reliable multicast protocols. IEEE Journal on Selected Areas in Communications, 15(3):398–406, 1997
15. Alan O. Freier, Philip Karlton, Paul C. Kocher.: The SSL Protocol Version 3.0. Work in progress, Netscape Communications, November 1996
16. B. Preneel.: Analysis and Design of Cryptographic Hash Functions. Phd thesis, Katholieke University, Leuven, January 1993
17. D. Boggs.: Internet Broadcasting, XEROX Palo Alto Research Center, Technical Report CSL-83-3, 1983