International Journal of Emerging Engineering Research and Technology
Volume 5, Issue 4, 2017, PP 6-11
ISSN 2349-4395 (Print) & ISSN 2349-4409 (Online)
http://dx.doi.org/10.22259/ijeert.0504002

# Secure Group Communication with Low Communication Overhead using Effective Multicasting

**Mr. P. L. Srinivasa Murthy**

*Associate Professor, GRIET, Hyderabad, India*

***Corresponding Author:** *Mr.P.L.Srinivasa Murthy, Associate Professor, GRIET, Hyderabad, India*

***Received Date: 19-05-2017*** ***Accepted Date: 25-05-2017*** ***Published Date: 26-05-2017***

## ABSTRACT

*Multicast networking support is becoming an increasingly important future technology area for both commercial and military distributed and Group based applications. Secure Group communication deals with the exchange of information between authorized group members. Security of packets delivered from a source to a large group of receivers presents one of most challenging problem for the network architecture. A majority of proposals for scalable secure multicasting makes use of hierarchal key distribution trees. In this paper we proposed a protocol with a lower communication cost for secure group key management called "N-Sec Cumulative Rekeying Protocol". The protocol is based upon the use of key trees for secure group and periodic batch rekeying.*

**Keywords:** *Group communication, Multicast Security, Session Key, Boolean Minimization.*

## INTRODUCTION

Internet applications, such as online games, newscast, stock quotes, multiparty conferences, and military communications, can benefit from secure internet multicast communications. In most of these applications, users typically receive identical information from a single or multiple senders. Hence, grouping these users into a single multicast group and providing a common session encryption key to all of them will reduce the number of message units to be encrypted by the senders.

The messages shared in a group are protected by encryption using a chosen key called the session key, which is also referred as Group key. Session keys are symmetric, that is the same group key is used to encrypt and decrypt messages by all the members in the group. The dynamic nature of group membership requires that the keys have to be renewed to achieve two basic requirements called Forward secrecy, which ensures that a passive adversary (member or non-member) who knows a contiguous subset of old group keys cannot discover subsequent group keys and Backward secrecy, which ensures that a passive adversary who knows a contiguous subset of group keys cannot discover preceding group keys.

However keeping the keys secret is one of the hardest problems in cryptography. If a sloppy key management system is in place, it is a very easy to steal the key from the storage database than to perform actual attack on the cryptographic algorithms or protocols. This paper addresses the key generation and distribution problems associated with maintaining communication integrity in the presence of membership changes.

In many public-key systems, it is also assumed that a centralized Certification Authority, trusted by all users, exists in the domain. Furthermore, the lifetime of each key directly affects the security of each encrypting key. Longer the key has been in used; the more likely it is to get compromised by attackers. So the keys have to be refreshed periodically and it should be changed whenever there is a change in group membership. As the frequency of group membership changes increases, it becomes necessary to reduce the cost of key distribution operations.

The centralized server or group controller creates a new group key after every join and leave. After a join, the new group key can be sent via unicast to the new member (encrypted with its individual key) and via multicast to existing group members (encrypted with the previous group key). Thus, changing the group key securely after a join is not too much work. After a leave, however, the previous group key can no longer be used and the new group key must be encrypted for each remaining group member using its individual key. Thus, we see that changing the group key securely after a leave incurs computation and communication costs proportional to the same as initial group key distribution. To reduce the cost of key re-distribution batch rekeying mechanism is used. This paper focuses on "*cumulative member removal*" [3] algorithm which uses Boolean minimization techniques to minimize the number of messages required to distribute new keys to the existing group members in a secure manner.

## KEY MANAGEMENT STRATEGIES

Key management [1] plays an important role of enforcing access control on the group key. It supports the establishment and maintenance of key relationships between valid parties. Three major approaches to group key management [5] are given below

❖ Centralized Group Key Management Architectures

In this type of key management a single entity is employed for controlling the whole group. But the major problem of single point of failure exists.

❖ Distributed Key Management Architectures

In distributed key management there is no explicit KDC, and the members themselves do the key generation. For large groups collecting the contribution from every user is tedious and time consuming, due to this reason scalability criteria is not fulfilled.

❖ Decentralized Architectures

In the decentralized subgroup approach, the large subgroup is split into small subgroups. In this approach, more entities are allowed to fail before the whole group is affected.

A large number of re-keying methods are developed based on the architectures discussed above. They are listed in the following paragraphs:

### Evolution of Re-Keying Mechanisms

In Group key management protocol (GKMP) [4], a Group Key controller creates a Group Key Packet (GKP) that contains a Group Key Encryption Key (GKEK) and a Group Traffic Encryption Key (GTEK). When re-keying is needed the KDC generates a new GKP and encrypts it with the current GKEK.

The LKH [7] scheme KDC maintains a tree of keys. Each leaf holds a KEK associated with one member. The root holds the group key. When a member leaves a group, its parent node's KEK and all KEKs held by nodes in the path to the root are compromised and changed.

Iolus [9] proposes a framework with a hierarchy of agents that splits the large group into small subgroups. A Group Security Agent (GSA) manages each subgroup. The GSAs are also grouped in a top-level group that is managed by a Group Security Controller.

Group Diffie Hellmann Key exchange is an extension for the DH key agreement protocol that supports group operations [5].The DH protocol is used for two parties to agree on a common key.

In key graphs [8], a trusted centralized key server maintains a hierarchy of keys. Key server is responsible for maintaining a relation between user set and key set. It uses three different rekeying mechanisms namely User oriented rekeying, Key oriented rekeying and Group oriented rekeying.

This paper discusses how to reduce the communication cost and the computation cost at the sender's side. In order to reduce the communication cost, a batch rekeying technique is involved. The proposed algorithm is called as " N- Sec Cumulative Eviction ". This algorithm uses simple Boolean techniques to identify a new session key based on the evicted member's id. Depends on the size of the group either K-map or Quine McCluskey can be used.

## N-SECS CUMULATIVE EVICTION GROUP REKEYING

In order to send the group rekey message a separate architecture is followed at the sender's side and the receiver's side. Figure 1.1 shows the block diagram of Sender part. Initially the user has to register to the Group Controller. Then the group controller creates the key database. The initial keys i.e. session key and a set of auxiliary keys for the registered users are transmitted before the actual transmission starts.

### Group Controller Model Description

The group controller then creates an initial key structure that will accommodate all the users and broadcasts the program encrypted by the common key called session key. Group controller also handles users de-registration or removal. Removal is initiated by the users.

The controller accumulates all removal and then runs the N-Sec Cumulative Eviction Group Rekeying algorithm to find the key in order to ensure that only registered users are communicating in the group.

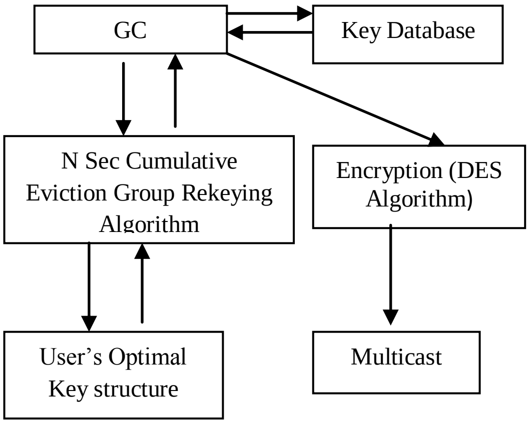
GC
Key Database
N Sec Cumulative Eviction Group Rekeying Algorithm
Encryption (DES Algorithm)
User's Optimal Key structure
Multicast

**Figure1.** *Group Controller (GC) part*

In short, the most important results of system or group induction are for an individual to establish a base system key known only to the individual and the system or group manager, and for the system or group manager to check the credentials of the individual.Operations processed by a group manager

- add member(s) to group
- remove member(s) from group
- evict member(s) from group

## Key Management Scheme

The important requirement for the implementation of the paper is that "No users outside the target set can decrypt the program". A unique user ID (UID) is given to every member of the group which is a binary string of length n. Consequently, a UID can be written as $X_{n-1}X_{n-2}....X_0$, where $X_i$ can be written as $\overline{x_i}$ or $x_i$ depending on whether $x_i$ is 0 or 1.The length of the UID depends upon the size of the group.

In order to join in a secure multicast session, either as a receiver or sender, a user has to request access to the group from the controller responsible for that session. If the user is authenticated person then he receives two different keys namely Session key and Auxiliary keys

The controller manages all the auxiliary keys, namely $\{k_0,\overline{k_o},k_1,\overline{k_1},.....k_{n-1},\overline{k_{n-1}}\}$. The keys possessed by different members in the group of size 8 are shown in Fig 2.1. For example, member $C_7$ with UID 111 possesses the auxiliary keys $k_2,k_1,k_0$ and session key SK shared by all users.

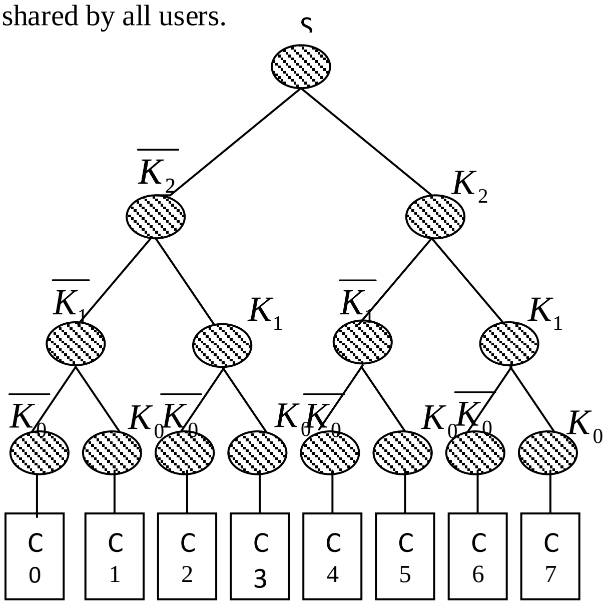

**Figure2.** *Key Distribution*

## Single Member Removal

Whenever a user of a group is to be expelled or voluntarily leaves group a new session key needs to be disseminated to every user except the one who left already. To facilitate the updating of session key SK , the controller has to compute the new session key $SK_{new}$ and this is encrypted by the keys that are "complementary" to the ones of the departing member.

Assume user 7 is leaving from the multicast group and his UID is 111. It possesses keys $k_2,k_1,k_0$. In order to achieve the forward secrecy, the session key has to be changed and it should be encrypted in such a way that the user who left the group should not be able to decrypt it. New session key is computed and it is sent as 3 different messages encrypted by the three different keys that are complementary to the evicted user.

Keys possessed by evicted user - $k_2,k_1,k_0$.

Complementary keys are - $\overline{k_2},\overline{k_1},\overline{k_0}$.

Messages are -
$\{SK(new)\}_{\overline{k_0}},\{SK(new)\}_{\overline{k_1}},\{SK(new)\}_{\overline{k_2}}$

The departing user also receives all the messages; it can not decrypt it, since every message is encrypted with a key that the departing member does not possess. It also guarantees that every other member of the group can decrypt at least one message. This differing key(s) can be used for decryption.

Figure 2.2 shows a visual representation of the rekeying method enlightened. In this figure the solid round nodes indicates the keys obsessed by the leaving member $c_7$. The hatched round nodes symbolize the complementary set that is the keys not obsessed by $c_7$. Every other branch has at least one hatched node. Now it is ensured that the new session key is encrypted with the complementary set of the leaving user, all members except the member who left the group will be able to decrypt at least one message. New session key is communicated to the remaining members in a secure manner.

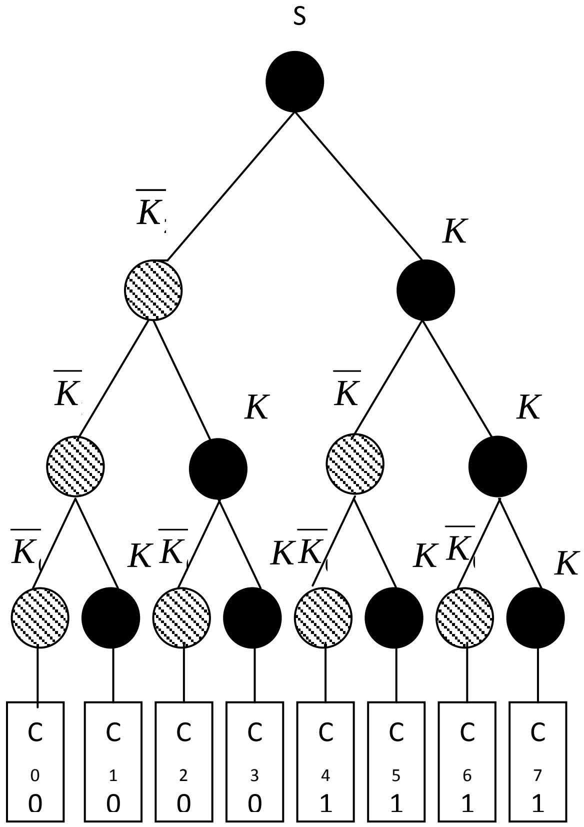

**Figure2.** *Departure of* $c_7$

This guarantees that only a member that is in possession of the new session key $SK(new)$ can obtain the updated auxiliary key $K_i(new)$. Since the departing member does not know the new session key $SK(new)$, it is excluded from the future updates of the session key.

## N-Sec Cumulative Member Removal

The key update process portrayed in the preceding section can be applied $K$ times consecutively to remove $K$ members from the group. However, a more proficient policy is to aggregate the removal of several members from the group.

A systematic approach to the problem of removing multiple members in the same round is called "Cumulative Member Removal "algorithm. Now consider a group of clients, $S = \{c_0, c_1, ..., c_{N-1}\}$, where $N = 2^n$. The user ID (UID) of a client $c$ is written, in binary form, as an $n$ - bit ID $u(c) = X_{n-1}X_{n-2}....X_0$,where $X_i, i = 0,1,...,n-1$ is either 0 or 1.

Rekeying can be accomplished by updating the session key and auxiliary keys or those members whose membership function $m(u) = 1$.Now consider the users $c_0, c_2, c_6$ with UID $000,010,110$ have to be removed from the group.

The rekeying is done by using only two different ways instead of $2 \times 3 = 6$ different ways. This is performed using Boolean minimization technique .Doing this way reduces the communication overhead.

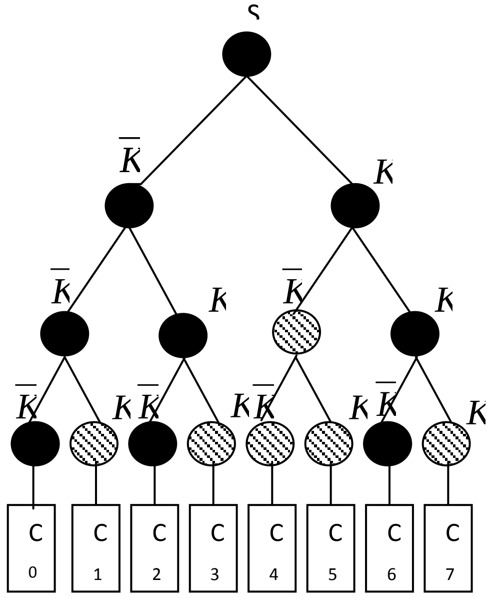

**Figure2.** *Users* $c_0, c_2, c_6$ *are leaving*

## PERFORMANCE ANALYSIS

The total storage area required is calculated using the formula given below

## Storage Complexity

n- Number of members in each subgroup

Total number of keys = $\log_2 n+1$.The storage complexity can be calculated as

Total storage = User storage + Controller storage

User storage = No of keys held by an individual user = $\log_2 n+1$

Controller storage = Total key space + session key = $2\times\log_2 n+1$

So total storage = $n\times(\log_2 n+1)+2\times\log_2 n+1$ = $(n+2)\log_2 n+n+1$

Thus the storage complexity increases in the order of $o(\log_2 n)$

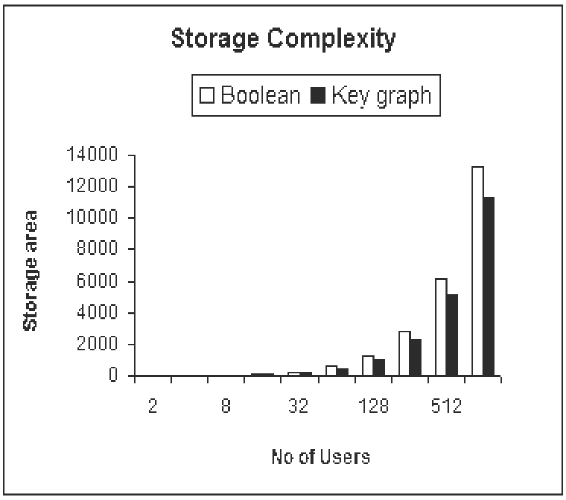

**Figure4.** *Total storage required for Boolean and Key Graph method at the user's side*

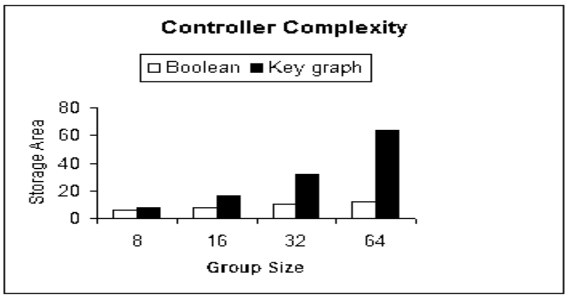

**Figure4.** *Controller storage complexities for Boolean and key graph method*

## Communication Complexity

### Eviction of a Single User

The communication complexity during a single member eviction is $o(\log_2 n)$.

## Eviction of Set of Users

Thus the communication complexity during a bulk removal is At best case it is less than $o(\log_2 n)$ and in worst case it is equal to n/2.

For Boolean minimization method: $(n+2)\log_2 n+n+1$

Key Graph method : $n(\log_2 n+3)-1$

For increasing number of users the storage required is reduced significantly in Boolean minimization method when compared with the key graph method. The controller complexity for both the Boolean and key graph method is calculated as,

For Boolean minimization method $2\times\log_2 n+1$

For Key graph method $2\times n-1$

## Communication Complexity Analysis

At the best case the performance achieved is $<o(\log_2 n)$. In worst case it is equal to $n/2$ where n is the no of users in the group. The communication complexity is measured in terms of its rekeying message.

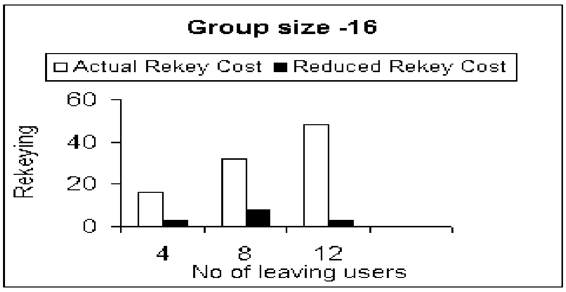

**Figure4.** *Communication Complexities for Group size -16*

**Table5.1** *Storage Complexity at Controller Side*

| Method | Complexity |
|---|---|
| Key Graph | $n(\log_2 n+3)-1$ |
| Boolean | $(n+2)\log_2 n+n+1$ |

Total storage required is calculated by Total Storage = total users storage + controller storage

**Table5.2 .** *Total Storage Complexity*

| Method | Complexity |
|---|---|
| Key graph | $o(n)$ |
| Boolean | $o(2\log_2 n)$ |

## REFERENCES

[1] S.Annadurai, T.Purusothaman "*Key Management for Internet pay sites*", Proceedings of National Conference on Modern Trends in Electrical and Instrumentation Systems, Anna University, Coimbatore, pp 273-279, 2007.

[2] C. Blundo, L.A. Frota Mattos, and D.R. Stinson, "*Generalized Beimel-Chor schemes for broadcast encryption and interactive key distribution*," Theoretical comput. Sci., vol. 200, no. 1-2, pp. 313-334, 1998.

[3] I.Chang, R. Engel, D. Kandlur, D. Pendarakis, and D. Saha., March, "Key management for secure Internet multicast using Boolean function minimization techniques", In *Proceedings of IEEE INFOCOM*, New York, pp.576-586.

[4] S. Mittra, Feb 2004, "IOLUS: Framework for scalable secure multicasting ", in Proc. ACM SIGCOMM'97, pp. 1 – 12.

[5] Sandro Rafaeli and David Hutchison, Sep 2003, "A Survey of Key Management for Secure Group Communication", ACM Computing Surveys, Vol.35, No.3, pp.309 – 329.

[6] C. K.Wong, M. Gouda, and S. S. Lam. , Feb 2004, "Secure group communications using key graphs", IEEE/ACM Transactions on Networking, pp.16–30.

[7] Yhavitt and Michel Abdalla, "Key Management for Restricted Multicast Using Broadcas Encryption," IEEE/ACM Transactions on Networking, vol. 8, no. 4, Aug 2000.

**Citation:** *P. Srinivasa Murthy, "Secure Group Communication with Low Communication Overhead using Effective Multicasting", International Journal of Emerging Engineering Research and Technology, vol. 5, no. 4, pp. 6-11, 2017.*