Secure Conference Scheme for Mobile Communication by Computing -Efficient Rekeying for Multicast Key Distribution.

Nilesh M. Shidurkar[1] , Prof. Mangesh M.Ghonge[2]
Student : Computer Science and Engineering ,jcet Yavatmal, India ,shidurkar.nilesh@gmail.com,
Assistant Professor : Computer Science and Engineering, jcetYavatmal, India,mangesh.cse@gmail.com,

Abstract: An important problem for secure group communication is key distribution. Most of the centralized group key management schemes employ high rekeying cost. Here we introduce a novel approach for computation efficient rekeying for multicast key distribution. This approach reduces the rekeying cost by employing a hybrid group key management scheme (involving both centralized and contributory key management schemes). The group controller uses the MDS Codes, a class of error control codes, to distribute the multicast key dynamically. In order to avoid frequent rekeying as and when the user leaves, a novel approach is introduced where clients recompute the new group key with minimal computation. This approach ensures forward secrecy as well as backward secrecy and significantly reduces the rekeying cost and communication cost. This scheme well suits wireless applications where portable devices require low computation.

Keywords: Erasure decoding, Key Distribution, MDSCodes, Rekeying, Multicast.

1. INTRODUCTION

Many group-oriented and distributed applications need security services which includes key management. Such applications need a secure group key to communicate their data. This brings importance to key distribution techniques For group-oriented applications, multicast is an essential mechanism to achieve scalable information distribution. Multicast describes communication where information is sent from one or more parties to a set of other parties. In this case, information is distributed from one or more senders to a set of receivers, but not to all users of the group. The advantage of multicast is that, it enables the desired applications to service many users without overloading a network and resources in the server.

Security is essential for data transmission through an insecure network. There are several schemes to address the unicast security issues but they cannot be directly extended to a multicast environment. In general, multicasting is far more vulnerable [4, 5, 6] than unicast because the transmission takes place over multiple network channels. A more difficult and challenging issue arises due to the multicast group membership being dynamic. Users can leave and join the groups, thus making the issue of group management more difficult in large-scale systems. Also we need to provide Forward Secrecy and Backward Secrecy. One of the most important issues in Multicast Security is the Group Key Management. Group key management, which is concerned with generating and updating secret keys, is one of the fundamental technologies to secure such group communications. Key management facilitates access control and data confidentiality by ensuring that the keys used to encrypt group communication are shared only among legitimate group members. Thus, only legitimate group members can access group communications. The shared group key can also be used for authentication. When a message is encrypted using the group key, the message must be from a legitimate group member. To prevent these problems, the following two security criteria are important for the group key distribution in secure multicast communication. Forward secrecy: If a person has left a group, the departed member cannot decrypt encrypted messages transmitted after the leaving. Backward secrecy: If a person joins a group, he cannot decrypt encrypted messages transmitted before the joining. The process for achieving forward and backward secrecy requires redistributing the group key. This process is called group rekeying [7][13]. There are three types of group key management schemes. In centralized key management, such as, group members trust a centralized server, referred to as the key distribution center (KDC), which generates and distributes encryption keys. In decentralized schemes, the task of KDC is divided among subgroup managers. In contributory key management schemes, group members are trusted equally and all participate in key establishment [8][12][14]. In this paper, we study how a multicast group key can efficiently be distributed in computation. In this a centralized key management model is used where session keys are issued and distributed by a central group controller (GC), as it has much less communication complexity, when compared to distributed key exchange protocols. The group controller uses the communication,Computation and storage resources for distributing the session key to the group members. The main problem here is how the resources can be used to distribute the session key. This is referred to as group key distribution problem. There are two approaches that are generally used for distributing the session key to the group of n members. The first approach is that the group controller GC shares an individual key with each group member. That key is used to encrypt a new group session key. In the second approach the group controller shares an individual key with each subset of the group, which can then be used to multicast a session key to a designated subset of group members. This approach has less communication, computation and storage complexity when compared to the other approach.

For a multicast group with large number of members key-tree-based approach is used. This approach decomposes a large group into multiple layers of subgroups with smaller sizes. Using this approach communication complexity is reduced, but the storage and computation complexity is increased.

In this paper, the main aim is to reduce the rekeying cost. A new novel approach for computation efficient rekeying for multicast key distribution is introduced. This approach reduces the rekeying cost by employing a hybrid group key management scheme and also maintains the same security level without increasing the communication and storage complexity. In this scheme, session keys are encoded using error control codes. In general encoding and decoding using error control code reduces the computation complexity. Thus, the computational complexity of key distribution can be significantly reduced.

2. LITERATURE REVIEW AND RELATED WORK

A. Computation-Efficient Multicast Key Distribution

An important problem for secure group communication is key distribution. In this paper, a new multicast key distribution scheme[10] is introduced whose computation cost is significantly reduced. This scheme employs MDS Codes, a class of error control codes, to distribute multicast key dynamically. This reduces the computation load of each group member. When this scheme is used with key-tree-based schemes, it provides much lower computation complexity which also maintains low and balanced communication complexity and storage complexity for secure dynamic multicast key distribution.

In key distribution scheme, a basic operation is to distribute a piece of secret data to a small group of n members, where each shares a different key with the GC. In the existing schemes, this is done by n encryptions, followed by n unicasts. In the new scheme, this is done by using one erasure decoding of certain MDS code, followed by one multicast to all n members. This is the basic key distribution scheme of key distribution that is used in this paper. This scheme can be integrated into any key distribution scheme, especially the schemes based on key trees, to reduce the computation cost. The multicast group that is used can have n members.

B. Iolus: A Framework for Scalable Secure Multicasting

Iolus approach [2] proposed the notion of hierarchy subgroup for scalable and secure mulitcast.In this method, a large communication group is divided into smaller subgroups. Each subgroup is treated almost like a separate multicast group and is managed by a trusted group security intermediary (GSI). GSI connect between the subgroups and share the subgroup key with each of their subgroup members. GSIs act as message relays and key translators between the subgroups by receiving the multicast messages from one subgroup, decrypting them and then remulticasting them to the next subgroup after encrypting them by the subgroup key of the next subgroup. The GSIs are also grouped in a top-level group that is managed by a group security controller (GSC). Although Iolus has improved the scalability of the system, because the member join or leave only affect their subgroup only while the other subgroup will not be affected. It has the drawback of affecting data path. This occurs in the sense that there is a need for translating the data that goes from one subgroup, and thereby one key, to another. This becomes even more problematic when it takes into account that the GSI has to manage the subgroup and perform the translation needed. The GSI may thus becomes the bottleneck.

C. *Logical Key Hierarchy*

The logical key hierarchy (LKH) [11] is an efficient approach that supports dynamic group membership. This method was proposed by Wallner et al. and Wong et al. individually. Waller et al. discussed binary trees and Wong et al. discussed the generalized case - key graphs, but the implicated ideas in their method is identical – to convert the cost of communication from linearly to logarithm with the group size of n. In this approach, the group controller (GC) maintains a logical key tree where each node represents a key encryption key (KEK). The root of the key tree is the group key used for encrypting data in group communications and it is shared by all users. The leave node of the key tree is associated with a user in the communication group. Each user secretly maintains the keys related to the nodes in the path from its leaf node to the root. We call the set of keys that a member knows the key path. Figure 1 shows a sample of key tree. When a member leaves the group, all the keys that the member knows, including the group key and its key path, need to be refreshed. When a member joins the group, GC authenticates the member and assigns it to a leaf node of the key tree. The GC will send the new member all the keys from his/her corresponding leaf node to the root. The main reason for using such a key tree is to efficiently update the group key if a member joins or leaves the group.

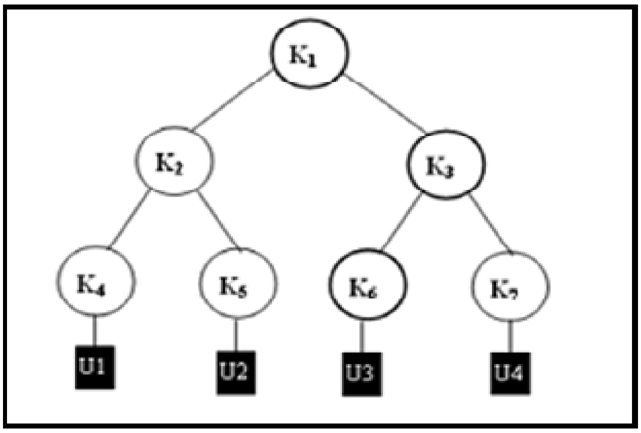

Fig.1:Logical Tree Structure.

D. Secure Group Communication Using Key Graphs

The key graph approach [3] assumes that there is a single trusted and secure key server, and the key server uses a key graph for group key management. Key graph is a directed acyclic graph with two types of nodes: u-nodes, which represent users, and k-nodes, which represent keys. User u is given key k only when there is a directed path from u-node u to k-node k in the key graph. Key tree and key star are two important types of key graph. In a key tree, the k-nodes and u-nodes are organized as a tree. Key star is a special key tree whose tree degree equals the group size. Key star is the basic key graph approach. In a key star, every member has two keys: the individual key and the group key. In a key tree, the root is the group key, leaf nodes are individual keys, and the other nodes are auxiliary keys.

A key tree is a rooted tree with the group key as root. A key tree contains two types of nodes: u-nodes containing user's individual keys, and k-nodes containing the group key and auxiliary keys. A user is given the individual key contained in its u-node as well as the keys contained in the k-nodes on the path from its u-node to the tree root. Consider a group with 9 users. In this group, user u9 is given the three keys on its path to the root: k9, k789, and k1−9. Key k9is the individual key of u9, key k1−9is the group key that is shared by all users, and k789 is an auxiliary key shared by u7, u8, and u9.

E. Batch Rekeying For Secure Group Communication

Batch rekeying [1] usually falls into two categories: rekeying after a fixed time period or rekeying after a fixed number of members have joined/departed the group. A point to be noted is that batch rekeying provides a trade-off between performance and security. Since the GC does not perform rekeying immediately, a departing member will remain in the group longer, and a joining member has to wait longer to be accepted to the group.

Join/depart requests that are collected during a period of interval, called the rekey interval, and they are rekeyed in a batch. Doing so not only alleviates the above issues, but it also reduces the number of group rekeying events. Furthermore, the number of rekeying messages that needs to be multicast to the group can be much smaller than the number of rekeying messages that would be generated if each membership change is to be processed individually, due to the overlapping in paths from the leaf nodes to the root.

3. ANALYSIS OF PROBLEM

We show that Hwang's conference key distribution protocol [9] is insecure against eavesdropping once a participant joins a conference in progress. Before that, we examine the representation of PI by Q, y and R. Each r_i is a sum of two 256-bit numbers r_{i1} and r_{i2}. Therefore, r_i is a 257-bit number and PI = K_c + lcm(r0,….,rm) has at most 257(m+1) bits, not 256(m+1) bits. Now, since Q and R are both 256-bit numbers, there will be cases when PI cannot be represented by Q, y, and R as claimed in the paper. However, since the representation of PI has no bearing on the security of the protocol, we suppose that PI is broadcast as it is to the participants, and we show that the protocol is insecure nonetheless.

For ease of notation and without loss of generality, suppose that U1, U2, and U3 are already in conference using the session key K_c when U4 joins in the conference. When U4 joins, NC sends PI = K_c +r4 to U4, which is observed by U1, U2, and U3. Now, U1, U2, and U3 all know K_c and hence can obtain r4 by computing PI - K_c. If U3, say, should quit the conference after U4 joined, NC would broadcast a new session key $K_c{}'$+ lcm($r_0{}'. r1 + t', r2 + t', r4 + t'$) and t' .However, U3 would then be able to compute $K_c{}'$ because he knows t' and r4 and hence would be able to eavesdrop on the conference that he has just left. Similarly, a new participant U5 may also follow the same line of attack by waiting for a participant to join before joining the conference himself and leaving immediately while continuing to be able to eavesdrop. This attack works because the new participant's secret is only masked by an entity K_c, which is known to all other current participants. One way of stopping this attack is to send $K_c + r_i . k_i$ for some random number k_i , so that r_i is never known by another participant.

4. PROPOSED WORK

In the earlier approach, the rekeying is done at every member join or leave. The new group key is multicasted to the group members each time by the group controller through multicasting. Using this group key, the group controller establishes a secure multicast channel with the authorized group members. In this, the group controller GC has to communicate with the group members each time when member leaves the group. The complexity of the rekeying operation changes because rekeying is done at every member join or leave the group. This makes the computational complexity very high. In our approach, the computational complexity can be much more reduced. The computational complexity can also be further reduced by reducing the no of rekeying operations.

Consider a group of n members. Usually when a member leaves the group rekeying operation should be performed to compute the new group key. This increases the burden on the server to recompute the group key and once again multicast to all the members of the group. As the nature of the members in group communication is dynamic, several rekeying has to happen. This is the major drawback in the earlier approach and inorder to overcome that we introduce a new novel approach which makes the computation complexity much more efficient and makes the rekeying cost more significant.

For a dynamic multicast group, a session key is issued by Group Controller (GC). The Group Controller uses this session key to establish a secure multicast with the authorized group members. When new members join or leave the group, the GC reissues the new session key to the authenticated group members. This ensures security of the current session and that of the old sessions. That is the newly joined members cannot recover the communications can not access the current session. Thus the forward secrecy and backward secrecy is maintained for the group

communication is maintained.

The complexity of the rekeying operation changes when new members join the group and old members leave the group. When a new member join the group, the GC multicast the new session key encrypted by the current session key to all the current members, followed by a unicast to the new member to send the new session key encrypted by a predetermined encryption key shared between the GC and the new member. Thus, with low computation cost and communication cost a new member can join the group. However, when an old member leaves the group , the current session key cannot be used to convey the new session key securely, since it is known to the old member.

In key distribution scheme, a basic operation is to distribute a piece of secret data to a small group of n members, where each shares a different key with the GC. In the existing schemes, this is done by n encryptions, followed by n unicasts. In the new scheme, this is done by using one erasure decoding of certain MDS code, followed by one multicast to all n members. This is the basic key distribution scheme of key distribution. This scheme is integrated into any key distribution scheme, especially the schemes based on key trees, to reduce the computation cost. The multicast group that is used can have n members.

The basic scheme[10] consists of three phases:

1. Generation of MDS codes and encoding the session key by the group controller
2. Multicasting the session key encoded MDS code, and
3. Retrieving the session key from the MDS code by individual members of the group.

a) Generation of MDS codes and encoding the session key by the group controller

In this functions that is used to create MDS codes by the group controller is initialized by the group controller and new member joins the group.

1) Initializing functions that is used to create MDS codes by the group controller[10]

The group controller makes both the MDS Codes C and the one-way hash function H public.

2) Member Initial Join[10]

When a new member i is authorized to join the multicast group for the first time, the GC sends it a pair (ji , si) using a secure unicast, where si is a random element and ji is a positive integer satisfying ji ≠ jk for all k’s , where k is a current member of the multicast group. The pair (ji , si) is called as seed key denoted by Si and is kept in the GC’s local database.

3) Rekeying[10]

Whenever some new members join or some old members leave a multicast group, the Group Controller needs to distribute a new session key to all the current members. After an old member leaves, the GC needs to distribute a new key to n remaining members to achieve both forward and backward secrecy of the session key. In this the group controller GC executes the rekeying process and sends the key to the client and when the authorized member of the group receives a message from the group controller, it can decode the key that is send to it by the group controller.

1. The GC randomly chooses a fresh element r in F, which has not been used to generate previous keys.

2. In the remaining group of n members, for each member i of the current group with its seed key (ji , si), the GC constructs an element Cj , in GF(q) : Cji = H(si + r), where + is simple combining operation in F.

3. Using all Cj ‘s in the above step, GC construct code word c of the (L,n) MDS code C: Set the ji th symbol of the code word to be Cji.. Using an erasure decoding algorithm for C , calculate the n corresponding message symbols m1, m2,... mn.

4. The GC sets the new session key k to be the first message symbol m1: k=m1.

5. The GC multicasts r and m2, m3..... mn.

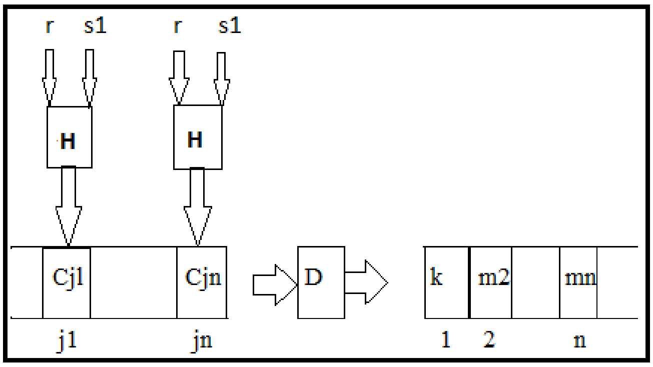

Fig. 2: Rekeying-GC’s Operations.

b) Multicasting the session key encoded MDS code

Here the group controller multicasts 'r' and m2, m3..... mn to the current group of 'n' members.

c) Retrieving the session key from the MDS code by individual members of the group

Upon receiving r and m2, m3..... mn from the GC, an authorized member i of the current group executes the following steps to obtain the new session key.

1. Calculate Cj =H(si + r) with its seed key (ji , s i).

2. Decode the first message symbol m1 from the (n-1) message symbols m2, ... mn, together with its codeword symbol Cji.

3. Recover new session key k, k = m1.

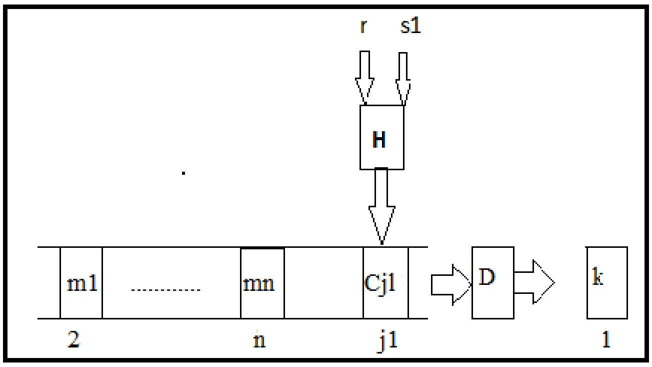

Fig. 3:Rekeying- Operation of members.

When authorized user receives the value of r and m2, m3.....mn from the group controller GC , each of the authorized user of the group again calculates the Cj . then again the code word is generated. Then an efficient erasure decoding algorithm is used to recover the new session key. This recovery process is done by each authorized user of the group. After applying the erasure decoding algorithm certain set of codes are formed from which the new session key that is send by the group controller is recovered. In this recovery process that uses MDS codes that is, theRS codes, are linear. The single codeword in this is the linear combination of all the n original message symbols. This decoding process is mainly used for solving the linear equation with one unknown. This is equivalent to an encoding operation with much lower computation than a general erasure decoding operation for multiple unknowns.

A.Novel approach for Computation-Efficient Rekeying

A set of dummy user are introduced by the server inorder to protect the size of the group (which plays a critical role in our approach). The dummy users introduced by the server randomly join or leaves the group. Now at anypoint of time the members in the group will be as GrpSizeold= uj +dj – (ul + dl), where uj and ul is user join and user leave and d j and d l is dummy user join and dummy user leave. Inorder to protect the group key information even when a user leaves, we consider the group size as the critical factor. It is understood that in group communication member join and member leave is a dynamic process. When a member leaves the group key should be redistributed and so computation cost becomes more tedious.

To calculate the new group key, the authenticated group member executes the following steps:

1. Initially, the GC computes the group key GrpKey and distributes to users by using the MDS Codes[10].

2. When uj no of user leaves the group, server randomly introduces djnew and dlleave. The user uj who left the group cannot predict the group size changes that has made in the group after he leaves.

3. Now the group size will be GrpSizenew= GrpSizeold + uin + din – (uout + dout) where uin is the no of members joining the group, uout is the no of members leaving the group, din is the no of dummy users joining the group and dout is the no of dummy users leaving the group.

4. The new group key is calculated as GrpKeynew = GrpSizenew + GrpKey.

5. Now a new value j is calculated such that j= GrpSizenew mod 64.

6. The new group key GrpKeynew is updated by undergoing a cyclic shift of GrpKeynew.

The steps 2,3,4,5,6 continues when the user leaves the group. Thus a new group key is calculated by each group members and rekeying is done This makes the computation cost less and the rekeying is more significant. But, in the earlier approach the computation cost is more because the multicasting is done at every rekeying process.

For security reasons, the rekeying using MDS codes has to be done in some interval. The frequency of rekeying is much lesser than earlier case when rekeying is done for every user leave. This subsequently reduces the rekeying cost and significantly improves the security.

Moreover the group dynamic membership information such as group size ,number of user joining, number of user leaving is unknown to any user.

5. OBJECTIVES

1) Reduces the rekeying cost.
2) Allowance for any active participant to join or to exit a conference.
3) Service many users without overloading a network.
4) Privacy of conversation content.
5) Reduce the computational complexity.

6. CONCLUSION

In this paper a novel approach is used which makes the computation cost much more efficient and the rekeying cost is significantly reduced. The group key is multicasted by the GC to the group members using the MDS Codes. Frequent rekeying is avoided when the user leaves, where clients recompute the new group key with minimal computation. This also makes the computation complexity greatly reduced. It also provides low and balanced communication complexity and storage complexity for dynamic group key distribution.

REFERENCES

[1] X.S. Li, Y.R. Yang, M.G. Gouda, and S.S. Lam, "Batch Rekeying for Secure Group Communications," *Proc. 10th Int'l World Wide Web Conf.* (WWW '01), pp. 525-534, May 2001.

[2] S. Mittra, "Iolus: A Framework for Scalable Secure Multicasting," *Proc. ACM SIGCOMM '97*, pp. 277-288, Sept. 1997.

[3] C. Wong, M. Gouda, and S. Lam, "Secure Group Communication Using KeyGraphs," *IEEE/ACM Trans. Networking*, vol. 8, pp.12-23, Feb. 2000.

[4] Peter S. Kruus and Joseph P. Macker, "Techniques and issues in multicast security," MILCOM98, 1998.

[5] Paul Judge and Mostafa Ammar**,** "Security Issues and Solutions in Multicast Content Distribution: A Survey", IEEE Network, February 2003, pp 30 – 36.

[6] M. Moyer, J. Rao and P. Rohatgi, "A Survey of Security Issues in Multicast Communications", IEEE Network Magazine, Vol. 13, No.6, March 1999, pp. 12-23.

[7] Yan Sun, and K.J. Ray Liu, "Securing Dynamic Membership Information in Multicast Communications," *IEEE INFO CONFERENCE 2004.*

[8] F.J. MacWilliams and N.J.A. Sloane, The Theory of Error Correcting Codes. North- Holland Math. Library, 1977.

[9] M. S. Hwang, "Dynamic participation in a secure conference scheme for mobile communications," *IEEE Trans. Veh. Technol.*, vol. 48, pp. 1469–1474, Sept. 1999.

[10]Lihao Xu, Cheng Huang, "Computation Efficient Multicast Key Distribution," *IEEE Trans. Parallel And Distributed Systems*, Vol 19, No. 5, May 2008.

[11]Ran Canatti, Juan Garay, Gene Itkis, Daniel Micciancio, Moni Naor, and Benny Pankus, "Multicast Security: A taxanomy and some efficient constructions", IEEE Network, March 1999, pp 122-128.

[12]H. Harney and E. Harder, Logical Key Hierarchy Protocol, IETF Internet draft, work in progress, Mar. 1999.

[13] T.M. Cover and J.A. Thomas, Elements of Information Theory. John Wiley & Sons, 1991.

[14]A.J. Menezes, P.C. van Oorschot, and S.A. Vanstone, Handbook of Applied Cryptography, fourth ed. CRC Press, 1999.