# Secure Group Key Management Scheme for Multicast Networks

R. Srinivasan, V. Vaidehi, R. Rajaraman, S. Kanagaraj,
R. Chidambaram Kalimuthu, and R. Dharmaraj
*(Corresponding author: R. Srinivasan)*

Department of Electronics Engineering, MIT Campus of Anna University
Chennai 600 044. India (Email: srini0402@yahoo.com)
*(Received Oct. 19, 2006; revised and accepted June 12, 2007)*

## Abstract

This paper proposes a scheme to provide security to dynamic multicast VoIP systems efficiently. Security is usually provided by encrypting the media packets sent from a user to other users with the help of a shared key called the session encryption key. The most time consuming process in a dynamic multicast VoIP environment is the group key management. Whenever there is a change in the group membership, the key needs to be updated and the updated key has to be sent to all active members in the group. Hence, by decreasing the number of update messages required for an updated key, the performance of the system can be improved considerably, thus making the scheme more efficient. The proposed secure multicast key management scheme combines the advantages of logical-key tree structure and Chinese remainder theorem to achieve an effective scheme. This paper compares the efficiency of the proposed scheme with the existing schemes and the comparison shows that the proposed scheme performs better than the existing schemes in terms of reduction in key update messages.

*Keywords: Group controller, group key management, multicast, unicast, session encryption key, sub group controller*

## 1 Introduction

Multicasting refers to the transmission of a message from one sender to multiple receivers or from multiple senders to multiple receivers. If the same message is to be sent to different destinations, multicast is preferred to multiple unicast. The advantage of multicast is that, it enables the desired applications to service many users without overloading a network and resources in the server.

### 1.1 Security Issues in Multicast

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 [6, 7, 9] 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. Forward secrecy implies that whenever a member of a group leaves the group, he should not be able to hear further conversations in that group. Backward secrecy implies that a new member joining the group should not be able to access the previous conversations in that group.

Multicasting includes both real time applications and non-real time applications. If these issues are not efficiently addressed, they could cause a severe bottleneck especially in real time applications such as VoIP systems. Thus it is absolutely essential that a security scheme used for providing security in a multicast environment should not only be secure but also very efficient in order to minimize these bottlenecks.

One of the most important issues in multicast security is the group key management. Several schemes [1, 4, 5, 10, 11, 13, 15] have been proposed for secure multicasting, which can be generally classified into two basic types, The centralized scheme and the distributed scheme. In the centralized scheme, group key management is done by the Group Controller (GC) and there are fewer burdens for the users of the group. In the distributed scheme, each user performs the necessary operations for group key management; hence there are more burdens on the users.

In unicast communication, security is provided by encrypting the message with the help of a key at the sender end and decrypting with the same key at the receiver end. In securing multicast group, a similar approach is taken. The entire group shares a key called the Session Encryption Key (SEK) which is known only to valid members of the group and which is used to encrypt any message that will be sent within the group. All members of this

group use this SEK to encrypt session messages meant for the group. Whenever there is a change in the group membership, the SEK needs to be updated to prevent new members from deciphering older conversations and prevent members that have left from deciphering future conversations. The new SEK is to be intimated to the currently valid members of the group. For this purpose, Key Encryption Key (KEK) is used. This KEK is used to encrypt and distribute the new SEK. The key management should take care of secure distribution of SEK to only valid members of the group. Key update communication and key storage are the two important overheads in key management.

The rest of the paper is organized as follows. The advantages and disadvantages of the existing schemes in the literature followed by the description of the proposed scheme for secure key management are dealt. The section that follows compares the efficiency of the proposed scheme with that of the existing schemes and finally the concluding remarks are provided.

# 2 Key Management Techniques

There are various proposed schemes for key management in the multicast groups from the simple minimal key storage scheme to the complex hybrid tree key distribution scheme, with their own advantages and disadvantages.

## 2.1 Minimal Key Storage Scheme

The minimal key storage scheme [8] is a very trivial scheme where each member $M_i$ is allocated a unique KEK $K_i$ where $i$ is the member index. In this scheme each member of the multicast group has to stores two keys, its KEK and the common SEK. When there is a change in membership in the group, GC has to encrypt the new SEK individually with $K_i$'s of the remaining $N-1$ members. Therefore, the communication overhead for updating the members with the new key is $O(N)$. To minimize the GC storage, a pseudo-random function $gr$ is used with a random seed $r$ as an index to generate the key $K_i$ as $K_i = g_r(i)$. The GC has to store only two keys, the SEK and the random seed $r$. Thus this method has constant key storage.

## 2.2 Logical Key Hierarchy

An improved scheme for key management is the logical key hierarchy [2] in which a logical tree of KEK's is constructed for a given group. In the tree, each leaf node is assigned a member, thus fixing the number of leaves to be the group size $N$. Every node of the tree is assigned a KEK. A member at a leaf node is assigned the set of keys that form the path from the root node. For example, member $M_1$ is assigned the $KEK_s$ $\{K_0, K_{11}, K_{21}$, and $K_{31}\}$. Thus the member storage required is the height of the tree and the overhead is $O(\log N)$. Since a member shares the root key and all the intermediate KEKs with other users, all the keys possessed by the member except the one at the leaf node have to be updated when the member leaves the group. For example, when member $M_1$ leaves the group, the keys $K_0$, $K_{11}$, $K_{21}$ have to be updated. The number of key update messages required is of the order of $O(\log N)$. But for this hierarchy, the GC has to store all the keys in the tree and hence the key storage overhead is of the order of $O(N)$.

## 2.3 Hybrid Tree Distribution

A better scheme for key management is the hybrid tree distribution [14]. The minimal key storage scheme has constant key storage but communication overhead is $O(N)$. The logical tree structure has communication overhead as $O(\log N)$ but has storage overhead as $O(N)$. The hybrid tree structure takes advantage of both the schemes. In this scheme the entire members of the group are divided into clusters of size $M$ with every cluster allocated to a single leaf node. Then there will be $N/M$ clusters and also $N/M$ leaves, we need to build a tree of depth $\log_d(N/M)$, where $d$ is the degree of the given tree. To illustrate the scheme let us consider, a group of 24 members. These members are clubbed to form clusters of size $M = 3$. In this scheme the user storage is of the order of $O(\log(N/M))$. The key update overhead is of the order of $O(M + \log(N/M))$. So as long as $M$ is not too large, the key update overhead is not severe. Thus the hybrid structure takes advantage of both the models and hence an efficient model for key management.

## 2.4 Secure Lock Using Chinese Remainder Theorem

In this scheme, a secure lock is constructed using Chinese Remainder Theorem (CRT). The secure lock is used to lock the deciphering group session key. The single lock is transmitted with each encrypted message. Only users in the secure group can "unlock" the session key. The principle behind the secure lock lies in the mathematics of the CRT. The CRT states that for $N_1$, $N_2$, ..., $N_n$ positive, relatively prime integers and $R_1$, $R_2$, ..., $R_n$ positive integers, a set of congruous equations $X \equiv R_1$ mod $N_i$, $X \equiv R_2$ *mod* $N_2$, $X \equiv R_n$ *mod* $N_n$ have a common solution $X$ in the range of $[1, L-1]$ where $L = N_1{}^* N_2{}^* N_3{}^* \ldots N_n$ and $n$ is the number of participants in the group.

The property of CRT is used to generate $X$ where $R_i = Eeki(d)$ is the session key $d$ encrypted by the function $E$ using participant $u_i's$ public enciphering key $eki$ (part of a public key pair). The common lock $X$ is generated by the Initiator using each participant's public enciphering key. Each participant can recover the locked session key $d$ by applying the CRT. The participant computes $d$ using their secret deciphering key $dki$. Only participants whose enciphering keys are included in the calculation of $X$ can unlock $d$. The secure lock method is flexible towards the dynamic addition and deletion of group members. Using

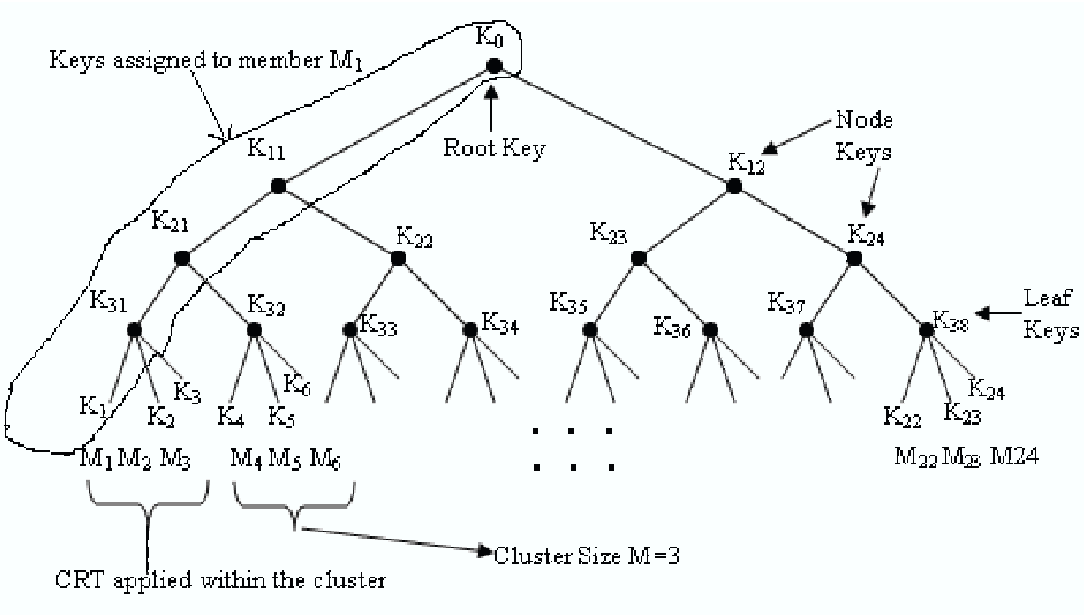



Figure 1: Proposed scheme for key management

the CRT, the Initiator can generate the common solution $X$ and rekey the group to include or exclude certain users from the group. Only those participants whose *eki* was used in the computation of $X$ can recover the session key. Because $X$ is common among all valid participants, the efficiency of the transmission of the lock (i.e) number of key update messages required is of the order of $O(1)$. Storage requirements at each participant site are limited to their public key pair.

# 3 Proposed Scheme for Key Management

All the previous schemes used only two entities namely the GC and the user/member. Here we introduce a new entity called the Sub-Group Controller (SGC). In this scheme, a logical tree structure is constructed and the total number of users $N$ is divided into clusters of size $M$. Each cluster is assigned to the leaf node of the logical tree. Logical key tree is used for inter-cluster key management and CRT scheme is used as intra-cluster key management. Since the group-controller alone cannot do the computation for finding the common solution $X$ for each cluster, the work is given to an entity what is called as a SGC. Each cluster is assigned to individual SGC, so that the SGC will only compute the common solution $X$ based on the session encryption key and the public key of the users within that cluster. After finding the common solution $X$ for that cluster, the SGC multicast the solution to all the users within that cluster. The scheme is described in the Figure 1 which is a modification of the hybrid tree scheme [14]. The number of update messages required in the proposed new scheme is of the order of $O(1 + \log(N/M))$. The key storage required for each user is very minimal because each user has to store its own public key and private key. Key storage required for each SGC is also very minimal because it has to store public keys of users within that cluster.

In this scheme, the GC accepts the request from users for joining a multicast group. After accepting the request, the GC authenticates the user and then requests the user for its public key. The user in response sends the public key to the GC. The GC constructs the logical tree structure according to the maximum number of users that can be supported by the system.

Then the GC sends the initial KEK to the appropriate users according to the logical tree structure. After sending KEK, the GC sends the public key of the valid users to SGC.

Then the GC encrypts the SEK with the KEK and sends it to the SGC. The SGC after receiving the SEK from the GC computes the secure lock (i.e) the common solution X using the SEK and public key of each user. Then it multicast the secure lock to all the users within the cluster. Each user within the cluster, after receiving the secure lock, applies CRT and decrypts it using its own private key to get the SEK. The user uses the SEK to encrypt any outgoing message to the group members or to decrypt any incoming message from any of the group members. Whenever there is a change in the group membership, the GC generates new SEK and encrypts it and sends it to the SGC. Then the secure lock is recomputed by all SGCs using the new SEK and public keys of the currently valid users in that cluster.

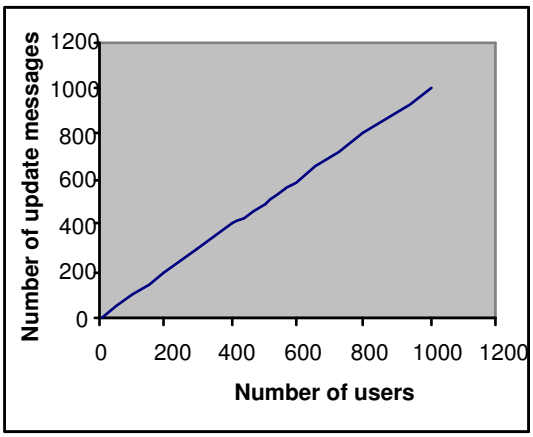



Figure 2: Key update complexities in minimal key storage scheme

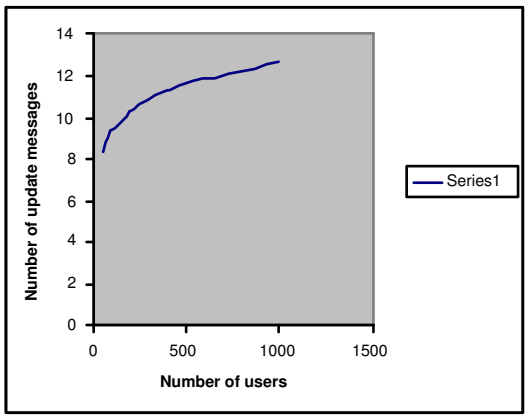



Figure 3: Key update complexities in hybrid key tree scheme

# 4 Analysis and Comparison

## 4.1 Comparison of Key Update Overhead

In the following section the overhead due to key updates are compared under various schemes:

- **Minimal Key Storage Scheme**
  The graph in Figure 2 shows the relation between number of update messages and the number of users in minimal key storage scheme. Since the complexity of the minimal key storage scheme is $O(N)$, we can see a linear relationship between the two variables.

- **Hybrid Key Tree Based Structure**
  The graph in Figure 3 shows the relation between number of update messages and the number of users in hybrid key tree structure. Since the complexity of the hybrid key tree based scheme is $O(M + \log(N/M))$ where $M$ is the cluster size ($M = 5$), we can see a logarithmic relationship between the two variables in the graph.

- **Proposed Scheme**
  The graph in Figure 4 shows the relation between number of update messages and the number of users in the proposed scheme. Since the complexity of the proposed scheme is $O(\log(N/M))$ where $M$ is the cluster size, we can see a logarithmic relationship between the two variables in the graph.

Thus comparing the hybrid key tree based scheme and the proposed scheme, the number of update messages required in the proposed scheme is reduced. This is due to the fact that the number of update messages in the proposed scheme is independent of the number of users within the cluster, whereas in the hybrid key structure the update messages depends on the number of users in the cluster.

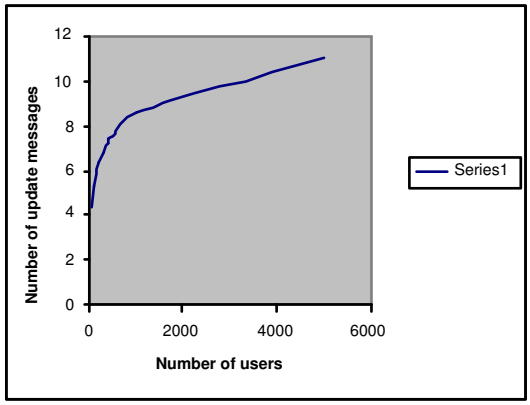


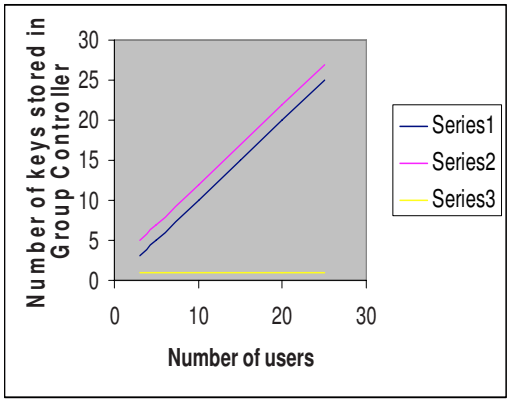

Figure 4: Key update complexities in the proposed scheme

## 4.2 Comparison of Key Storage

The key storage requirement in different schemes has been compared. This is plotted in Figure 5, the minimal key storage scheme has the lowest key storage complexity. The key storage in our scheme is only slightly higher than that of hybrid key tree structure which is the minimum cost for the reduction in update message.

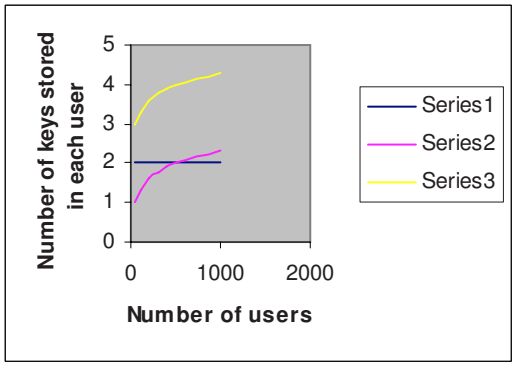
- **Key Storage in Group Controller**
  The graph in Figure 5 shows the relationship between the number of users and the number of keys stored in the GC. As the graph implies, there is no apparent increase in the number of keys stored as compared to the hybrid key tree based scheme.

- **Key Storage in User**
  The graph in Figure 6 shows the relationship between the number of users and the number of keys stored in each user. As the graph implies, there is there is a moderate increase in the number of keys stored in the users in the proposed scheme as compared to




Series 1 - Hybrid key tree based scheme
Series 2 - Our proposed scheme
Series 3 - Minimal key storage scheme

Figure 5: Key storage in GC for different schemes




Series 1 - Minimal key storage scheme
Series 2 - Hybrid key tree based scheme
Series 3 - Our proposed scheme

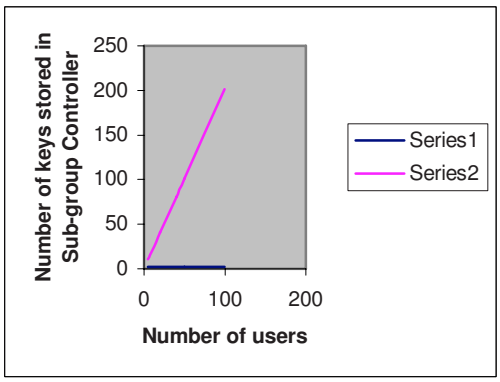
Figure 6: Key storage in user for different schemes

the hybrid key tree based scheme for a cluster size $M = 5$.

- **Key Storage in Sub-Group Controller**
  The graph in Figure 7 shows the relationship between the number of users and the number of keys stored in SGC. As the graph implies, there is almost a linear increase in the number of keys stored in the SGC as compared to the hybrid key tree based scheme for a cluster size $M = 5$. Since the key storage in the SGC is dependent on the cluster size, there is a variation in the key storage in the SGC.

## 5 Conclusion

In this paper a CRT based multicast secure key management scheme is presented. The performance of the proposed scheme is compared with the existing schemes in terms of update message and storage requirement when there is a change in group membership. The update complexity of the proposed scheme is found to be in the order of $O(1 + \log(N/M))$. The proposed scheme is found to provide a less update complexity in the network at the cost of moderate increase in the storage requirement.




Series 1 - Hybrid key tree based scheme
Series 2 - Proposed scheme

Figure 7: Key storage in SGC for different schemes

## References

[1] K. Becker and U. Wille, "Communication complexity of group key distribution," *Proceedings ACM Conference Computer and Commununications Security*, pp. 1-6, New York, NY, 1998.

[2] R. Canatti, J. Garay, G. Itkis, D. Micciancio, M. Naor, and B. Pankus, "Multicast security: A taxanomy and some efficient constructions," *IEEE Network*, vol. 2, pp. 122-128, Mar. 1999.

[3] K. C. Chan and S. H. G. Chan, "Key management approaches to offer data confidentiality for secure multicast," *IEEE Network*, vol. 17, no. 5, pp. 31-39, Oct. 2003.

[4] W. Chen, Y. Sun, and A. S. Pyzdrowski, "On the security cost of interval multicast," *International Journal of Network Security*, vol. 10, no. 3, pp. 204-212, 2010.

[5] S. Gharout, Y. Challal, and A. Bouabdallah, "Scalable delay-constrained multicast group key management," International Journal of Network Security, vol. 7, no. 2, pp. 160-174, 2008.

[6] P. Judge and M. Ammar, "Security issues and solutions in multicast content distribution: A survey," *IEEE Network*, vol. 17, pp. 30-36, Feb. 2003.

[7] P. S. Kruus and J. P. Macker, "Techniques and issues in multicast security," *MILCOM'98*, vol. 3, pp. 1028-1032, 1998.

[8] M. Li, R. Poovendran, and C. Bernstein, "Design of secure multicast key management scheme with communication budget constraint," *IEEE Communications Letters*, vol. 6, no. 3, pp. 108-110, Mar. 2002.

[9] M. Moyer, J. Rao, and P. Rohatgi, "A survey of security issues in multicast communications," *IEEE Network Magazine*, vol. 13, no. 6, pp. 12-23, Mar. 1999.
[10] M. M. N. Rasslan, Y. H. Dakroury, and H. K. Aslan, "A new secure multicast key distribution protocol using combinatorial boolean approach," *International Journal of Network Security*, vol. 8, no. 1, pp. 75-89, 2009.
[11] M. Steiner, G. Tsudik, and M. Waidner, "Diffie-Hellman key distribution extended to group communication," *ACM Computer and Communications Security*, pp. 31-37, 1996.
[12] D. M. Wallner, E. J. Harder, and R. C. Agee, *Key Management for Multicast: Issues and Architecture*, RFC 2627, June 1999.
[13] L. Wang and C. K. Wu, "Efficient key agreement for large and dynamic multicast groups," *International Journal of Network Security*, vol. 3, no. 1, pp. 8-17, 2006.
[14] J. Zhang, V. Varadharajan, and Y. Mu, "A novel dynamic key management scheme for secure multicasting," *ICON'03*, pp. 391-395, 2003.
[15] Q. Zhang and K. L. Calvert, "A peer-based recovery scheme for group rekeying in secure multicast," *International Journal of Network Security*, vol. 6, no. 1, pp. 15-25, 2008.

**R. Srinivasan** received his BE (ECE) from University of Madras, India, ME (Applied Electronics) from Madurai Kamaraj University, India and Ph.D from Anna University, India. He worked as Project Associate in the Microsatellite Project of Anna University from June 2003 to February 2006. He had published several papers in journals and conference proceedings. His areas of interest include multicast Networking, Network security and VoIP.

**V. Vaidehi** received BE (ECE) from College of Engineering Guindy, ME (Applied Electronics) and PhD from Madras Institute of Technology, Anna University. She had joined MIT in 1982 and currently she is a Professor in the Department of Electronics Engineering, MIT, Anna University. She had published several papers in journals and conference proceedings and had taken up several sponsored research projects. Her areas of interest include Networking, parallel processing and ADSP.

**R. Rajaraman, S. Kanagaraj, R. Chidambaram Kalimuthu, R. Dharmaraj** Final Year B.Tech students in the Department of Electronics engineering, Madras Institute of Technology Campus of Anna University, India.