Secure Group Key Management Scheme for Simultaneous Multiple Groups with Overlapped Memberships Using Binomial Key Tree

B.R. Purushothama, Kusuma Shirisha, and B.B. Amberker

Department of Computer Science and Engineering
National Institute of Technology Warangal
Andhra Pradesh-506004, India
{puru,bba}@nitw.ac.in, shirisha.kusuma@gmail.com

Abstract. The rich literature available for key management schemes for Secure Group Communication focuses on operating only a single group. We consider the operation of simultaneous/concurrent multiple groups in the Secure Group Communication model with overlapping memberships. Designing a secure key management scheme with efficient rekeying process in this scenario is a challenging task. We design an efficient secure group key management scheme for simultaneous multiple groups with overlapping memberships. We propose a new key structure called Binomial Key Tree Queue to manage the keys. Our scheme scales well as the overlapping memberships across the multiple groups increases. We compare the schemes with two schemes which have recently focused on key management protocol design for simultaneous multiple groups. The proposed scheme achieves significant reduction in rekeying cost, storage compared to these schemes. Interestingly, we achieve this efficiency in the rekeying cost without much increase in storage at user.

Keywords: Simultaneous multiple groups, Binomial Key Tree Queue, Overlapping memberships, Rekeying, Group Key Management.

1 Introduction

Secure Group Communication (SGC) refers to a scenario in which the group of users communicate securely among themselves such that confidentiality of the message is maintained among the group users in a way that outsiders are unable to get any information even when they are able to intercept the messages. Several applications like distributed interactive simulation, video conferences, collaborative work, teleconferences, white-boards, tele-medicine, real-time information services take advantage of the SGC model. The confidentiality of group communications is provided by encrypting group messages with a common shared secret, called *group key* among the group members in group communication.The confidentiality of the messages within the secure group are provided using encryption methods. The encryption of the message within the group is carried out using the common key called *group key*. The users possessing this *group key*

S.M. Thampi et al. (Eds.): SNDS 2012, CCIS 335, pp. 135–146, 2012.

can participate in the group communication securely. The group messages are shielded from the non-members of the group, as these users wont have access to the *group key*.

So, key management is a challenge in the SGC which is about methodology that enables distributing the *group key* securely among the group users. Often the groups are dynamic in which the users join and leave the group during the lifetime of the group. Whenever a member join/leaves a group the *group key* needs to be changed as the newly joining member should not be able to access the past communication (**backward access control**) and the leaving user should not be able to access the future communication (**forward access control**) of the continuing group communication. So, how to change the key, both efficiently and scalably, is a challenge.

The group key management schemes proposed in the literature follow different approaches. Centralized group key management schemes or protocol employ a trusted centralized entity called *Key Distribution Center (KDC)* for key management [1][2][3]. In decentralized group key management schemes the responsibility of managing the large group is divided among subgroup managers [4]. The group key is generated by all group members contribution in Distributed or Contributory group key management schemes [5][6] and the group members carry out the task of access control. Sandro et.al in [7] provide the classical survey on *group key management schemes*.

The rich literature focuses on key management in a single operating group. Recently, we have focused our research on group key management in the simultaneous (concurrent) multiple groups with overlapping membership. The challenge in this scenario is to make a user of a group to participate in multiple groups with less keys storage and rekeying cost.

1.1 Our Contribution

We have designed a SGC scheme for simultaneous multiple groups with overlapping membership using Binomial Key Tree Queue.

- We propose new key structure called Binomial Key Tree Queue for managing the simultaneous multiple groups with overlapping membership.
- We compare the proposed scheme with the schemes proposed in [8] and [9] for rekeying cost including number of encryptions, key changes and rekeying messages during membership change. We show that our proposed scheme is much efficient than the schemes in [8] and [9].
- Our keying scheme has the property that a user of a group can be in other multiple group communication sessions with only two keys per group apart from the keys which he has to hold for his own group and number of encryption and key changes are also significantly less upon group join and leave activities.
- In the proposed scheme, the efficiency in rekeying cost is achieved due to the non computation of auxiliary keys for a user when a user does have overlapping membership in other groups.

2 Simultaneous Secure Multiple Groups and Overlapping Membership in SGC Model

A secure group is the set of users communicating securely among themselves. The members of the secure group will have a *group key* using which they communicate securely. Secure multiple group is the collection of secure subgroups which are secure groups on their own. Each secure subgroup comprises of set of distinct users. The users of the subgroup communicate among themselves using their corresponding *group key.*

Definition 1 (Overlapping Membership in SMG). *Let $G_1, G_2, \ldots, G_m$ be the groups operating simultaneously. The members of the parent group G_i, for $i \in [1, m]$ want to communicate with the other groups G_j for $j \neq i$, $j < m$. Then, these members are said to have overlapping membership with the groups G_j.*

In simultaneous multiple groups with overlapping membership, the groups and the users of the groups are categorized as following. Let G_i be the group with users $\{u_1^i, u_2^i, \ldots, u_n^i\}$ and G_j be the group with users $\{u_1^j, u_2^j, \ldots, u_n^j\}$. Suppose k users of G_j have overlapping membership with group G_i. W.l.o.g let these users be $\{u_1^j, u_2^j, \ldots, u_k^j\}, k < n$. We categorize the users of the group G_i as

- **Parent Group and Parent Group Users**: For the users in $\{u_1^i, u_2^i, \ldots, u_n^i\}$, G_i is the parent group and these users are the parent group users of G_i. For the users $\{u_1^j, u_2^j, \ldots, u_n^j\}$, G_j is the parent group and these users are the parent group users of G_j.
- **Non-Parent Group and Non-Parent Group Users**: For the users in $\{u_1^j, u_2^j, \ldots, u_k^j\}$, for $k \leq n$, G_i is the non-parent group and these users are the non-parent group users of G_i,.

2.1 Example Illustrating the Overlapping Memberships in Simultaneous Multiple Groups

Consider Fig 1. There are three groups *Group A, Group B*, and *Group C*. For distinguishing the users of the various groups , members of the groups are colored red, green and blue for *Group A, Group B*, and *Group C* respectively. In Fig 1 there are 9 users colored red in *Group A*. For these users *Group A* is the **parent group**. Likewise, there are 10 and 8 users respectively in **parent group** *Group B* and **parent group** *Group C* colored green and blue respectively. The secure groups *Group A, Group B,* and *Group C* operate simultaneously. Therefore, these are termed as *simultaneous multiple groups.*

Overlapping membership is defined as the members of *Group i* for whom the *Group i* is the *parent group* and want to communicate with members of other groups *Group j*, where $i \neq j$ and $i, j = 1, 2, 3$ in Fig 1.

In Fig 1 , the overlapping memberships can be interpreted as described below.

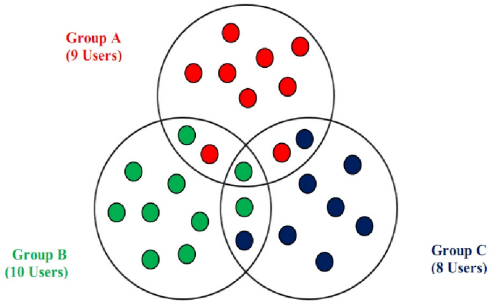

Fig. 1. Multiple Groups and Overlapping membership

1. The overlapping memberships of the group members of *parent group A*,
   - One of the member of *Group A* (colored red) wants to communicate with *(*Group B). This can be seen in the area $A \cap B$ of Fig 1. So, that user is said to have a overlapping membership with the *Group C*.
   - Likewise, a member of *Group A* in $A \cap C$ has overlapping membership with the *Group C*.
2. The overlapping membership of the group members of *parent Group B*,
   - A member of *Group B* in $(B \cap A) - (A \cap B \cap C)$ has a overlapping membership with *Group A*.
   - A member of *Group B* in $B \cap C - (A \cap B \cap C)$ has a overlapping membership with *Group C*.
   - A member of *Group B* in $B \cap A \cap C$ has a overlapping memberships with both the groups *Group A* and *Group C*.
3. Likewise, the overlapping membership of the group members of *parent Group C* can be interpreted.

The same illustration can be found in our previous work [8].

3 Notations and Definitions

- Let $U = \{u_1, u_2, \ldots, u_n\}$ be the set of users.
- $G_1, G_2, \ldots, G_m$ be the m groups with $n_1, n_2, \ldots, n_m$ distinct users respectively. No two users are same in the groups G_i, for $i = 1$ *to* m. In other words, $G_1, G_2, \ldots, G_m$ are disjoint.
- $\{M\}_K$: Encrypt the message M with key K. If $M = m_1, m_2, \ldots, m_n$, then encrypt each m_i, for $i, \ldots, n$ and send as one message or encrypt $m_1, \ldots, m_n$ and send. In the latter case, the receiver is assumed to know how to segregate the decrypted message.
- $Userset(K)$: Set of users possessing the key K.
- $u_j \rightarrow KDC : (J, G_i)$, Join request to KDC from a user u_j to join the group G_i. When the context is clear the single user u_j can be replaced with set of users $\{u_1, \ldots, u_l\}$.

Definition 2 (Binomial Tree). *The binomial tree S_h of height h is defined recursively [10]. The binomial tree S_0 has only one node. The binomial tree S_h has two binomial trees S_{h-1} that are connected as a one single tree. The root of one subtree is the leftmost child of the root of the other subtree. The Binomial tree satisfies the following properties. Let n be the number of nodes in a binomial tree.*

1. *If $n = 2^h$, the binomial key tree will be with*
   - *One binomial tree S_h and the degree of the root is h*
   - *Height h and 2^h nodes*
   - *Exactly $\binom{h}{i}$ nodes at depth i, such that $i = 0, 1, \ldots, h$*
2. *If n is not a power of 2 and $2^h + 1 \leq n \leq 2^h - 1$. Then,*
   - *The Binomial is a forest of binomial subtrees.*
   - *Consists atmost $h+1$ binomial subtrees whose height range from 0 to h.*
   - *There will be surely a binomial subtree S_h with height h and depending on n, there exists binomial subtrees with heights ranging from 0 to $h-1$.*
   - *If it exists, there will be only one binomial subtree S_i, for some $i \in [0, h]$*

Definition 3 (Binomial Key Tree). *Binomial Key Tree is essentially a binomial tree consisting of nodes representing the users $\{u_1, \ldots, u_n\}$ in U. Each node (user node) also represents the keys held by the corresponding user.*

Definition 4 (Binomial Key Tree Queue (BKTQ)). *It is a Queue of Binomial Key Trees or forest of Binomial Key Trees.*

- $BKTQ_G$: This denotes the representation of a secure group G using $BKTQ$.
- $BKTQ_G[h]$: It is a BKT with height h of the group G.
- $BKTQ_G[h, 1]$: BKT of height h of the parent group users in $BKTQ_G$
- $BKTQ_G[h, 2]$: BKT of height h of the non-parent group users in $BKTQ_G$
- $BKTQ_{PGU(G)}$: This denotes the representation of a secure group consisting of parent group users of G using $BKTQ$
- $BKTQ_{NPU(G)}$: This denotes the representation of a secure group consisting of non-parent group users of G using $BKTQ$

4 Proposed Binomial Key Tree Queue Structure for Secure Group

In this section, we define the *Binomial Key Tree Queue (BKTQ)* that is constructed by the KDC for managing the group keys in the scenario of multiple secure group communication with overlapping membership. For each secure group G the KDC constructs a $BKTQ_G$.

KDC constructs set of $BKT's$ for the parent group users in G as in [11]. W.l.o.g let these set of BKT's be $\{S_h, S_{h-1}, \ldots, S_0\}$ whose height is respectively $h, h-1, \ldots, 0$. It should be noted that h depends on the number of parent group users. The BKT $S_i, i \in [0, h]$ will have 2^i user nodes along with their corresponding keys. The KDC Constructs the $BKTQ_{PGU(G)}$ for parent group

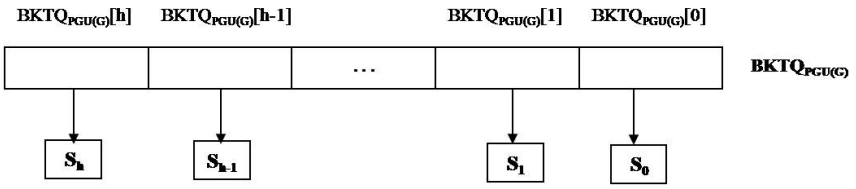

Fig. 2. Binomial Key Tree Queue Representation for Parent Group Users of Group G

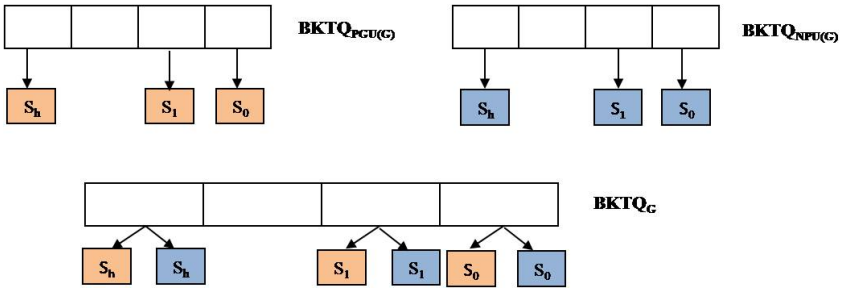

Fig. 3. Lazy Merge of BKTQ of Parent and Non-Parent Group Users of Group G

users in G as shown in Fig 2. As a notational convenience if any of the BKT S_i is empty we denote it as $BKTQ_{PGU(G)}[i] = \emptyset$.

KDC constructs the set of $BKT's$ for the non-parent group users of G followed by the $BKTQ_{NPU(G)}$ as shown in Fig 2. KDC will run the **Algorithm 1** to construct the final BKTQ of group G, $BKTQ_G$ by lazily merging $BKTQ_{PGU(G)}$ and $BKTQ_{NPU(G)}$. Finally, the BKTQ of group G as a result of execution of **Algorithm 1** looks as in Fig 3.

5 Secure Group Key Management Scheme Using *BKTQ*

In this section, we elaborate the group set up , join of parent and non-parent group users and leave of parent and non-parent group users. Readers should note that due to space limitations we are unable to give examples. We elaborate in detail the protocol and the ideas that helps in analysis of the protocols for rekeying cost. For binomial key tree basic join and leave operations one can refer [11][1]. We focus on the non-parent group user join and leave operations and show that our proposed scheme is efficient n comparison with the existing schemes.

Our scheme employs KDC to manage the simultaneous groups. Initially, the groups are empty. We assume that each user who joins the group will have a shared secret key with the KDC. We assume that the KDC authenticates the user before sending securely the shared key to the user.(Initially we assume the existence of the secure channel to give the shared key to user).

For every group G_i for $i = 1$ *to* m KDC forms the $BKTQ$ as explained in section 4. Each group will have a group key K_{G_i}.

Algorithm 1. Lazy Merge of Binomial Key Tree Queues: LazyMerge-BKTQ

Input :
- $BKTQ_{PGU(G)}$, Binomial Key Tree Queue of Parent Group users of G
- $BKTQ_{NPU(G)}$, Binomial Key Tree Queue of Non-Parental Group users of G

Output: $BKTQ_G$, Binomial Key Tree Queue of G

Let h_1 be the height of the leftmost BKT in $BKTQ_{PGU(G)}$;
Let h_2 be the height of the leftmost BKT in $BKTQ_{NPU(G)}$;
for $i \leftarrow 0$ *to* $min(h_1, h_2)$ **do**
  if $BKTQ_{PGU(G)}[i] \neq \emptyset$ *and* $BKTQ_{NPU(G)[i]} \neq \emptyset$ **then**
    - $BKTQ_G[i,1] = BKTQ_{PGU(G)}[i]$;
    - $BKTQ_G[i,2] = BKTQ_{NPU(G)}[i]$;

  end
  if $BKTQ_{PGU(G)}[i] = \emptyset$ **then**
    - $BKTQ_G[i,1] = \emptyset$;
    - $BKTQ_G[i,2] = BKTQ_{NPU(G)}[i]$;

  end
  if $BKTQ_{NPU(G)}[i] = \emptyset$ **then**
    - $BKTQ_G[i,1] = BKTQ_{PGU(G)}[i]$;
    - $BKTQ_G[i,2] = \emptyset$;

  end
end

5.1 Join of Parent Group User

When a new user u wants to join the parent group $G_i, i \in [1,m]$, KDC runs **Protocol 2**. The joining point for the new user is in the subtree corresponding to the parent group users of the corresponding group's BKTQ. The group key, the subtree key of the subtree corresponding to the joining point needs to be changed and communicated to the appropriate users.

5.2 Leave of Parent Group User

Suppose a parent group user u wants to leave parent group G_i, for $i \in [1,m]$. KDC runs **Protocol 3**. The leaving point is in the subtree of the parent group users corresponding to the leaving group's BKTQ. The group key, the subtree keys of the subtree of which the leaving user is part should be changed. The changed keys should be communicated to appropriate users. For the non-parent group users the KDC encrypts the changed group key using the subtree keys of the BKTQ corresponding to the non-parent group users.

Protocol 2. Join Protocol for the Parent Group

Input :
- Security parameter, k

1 Let K_{G_j} be the current group key of G_j represented by $BKTQ_{G_j}$;
2 Let n be the number of users in G_j;
3 $u \rightarrow KDC : (J, G_j)$;
4 $KDC \Longleftrightarrow u$: KDC Authenticate the user u and distribute K_u;
`// ` K_u `is the shared private key of the user` u `with` KDC;
5 KDC generates randomly a new group key $K^{'}_{G_j}$.;
6 Refer Join protocol in [1] or steps $4-14$ of Algorithm 1 in [11] ;
7 $KDC \rightarrow userset(K_{G_i}) : \{K^{'}_{G_j}\}_{K_{G_j}}$;
8 $KDC \rightarrow u : \{K^{'}_{G_j}\}_{K_u}$;

Protocol 3. Leave Protocol for the Parent Group

1 Follow the leave protocol in [11] and execute the next step.;
2 Send the new group key $K^{'}$ to the non-parent group users using their respective subtree keys.;

6 Join and Leave of Non-parent Group Users

Consider $G_1, G_2, \ldots, G_m$ groups represented as $BKTQ_{G_i}$, for $i = 1, \ldots, m$ such that $BKTQ_{G_i}[j, 1]$, for $j = 1, \ldots, h$ as the parent group users BKT's and $BKTQ_{G_i}[j, 2]$, for $j = 1, \ldots, h$ as non-parent group user BKT's. To achive efficiency in the rekeying process in the user join and leave operation we exploit the structure of Binomial Key Subtrees.

Suppose the users of G_i want to have overlapping membership with users of G_j. We propose the following process.

- We know that KDC has to rearrange the $BKTQ_{G_j}$ when a new user joins. KDC is assumed to make the users of G_i who wants to have overlapping membership with G_j as a part of one subtree. The users in this new subtree are non-parent group users of G_j.
- All the users who belongs to a subtree will have a common key (each subtree will have a common key). When these users join as non-parent group users to G_j there is no need to generate the intermediate/auxiliary keys for these users in G_j.
- These users are given only a single key of the new subtree that is formed by the KDC as part of G_j and the group key of G_j.
- With these the storage at the non-parent group user will be 2 keys apart from what he holds for his parent group.

The process of joining of a non-parent group user is given in Protocol 4.

Protocol 4. Join Protocol for the Non-Parent Group

Let $BKTQ_{G_i}$, for $i = 1, \ldots, m$ be the BKTQ's that contains the set of BKT's from G_i, for $i = 1, \ldots, m$ that wants to have overlapping memberships with G;
for *i=1 to m* **do**
$BKTQ_{NPU(G)} = MergeBKTQ(BKTQ_{G_i}, BKTQ_{NPU(G)})$ `// Refer Algorithm 6 for MergeBKTQ`
end
KDC generates keys for $BKTQ_{NPU(G)}[i]$, for $i = 0$ *to* h;
`// These are the keys for the subtrees constructed;`
KDC generates new group key $K^{'}$;
$KDC \rightarrow Userset(K) : \{K^{'}\}_K$;
`//` K `is the old group key. The existing non-parent group users of` G `will also get this new group key;`
for *i=0 to h* **do**
$KDC \rightarrow Userset(K_{BKTQ_{NPU(G)}[i]}) : \{K^{'}\}_{K_{BKTQ_{NPU(G)}[i]}}$;
end
`// This distributes the keys for the newly joined non-parent group users.`

Algorithm 5. Combine Binomial Key Subtree: CombineBKT

Input : S_i, S_j, Binomial Key Trees of same height h
Output: Binomial Key Tree of height, h or $h + 1$
if $S_i \neq \emptyset$ *and* $S_j \neq \emptyset$ **then**
return $< S_{i+1}, i + 1 >$; `// See Definition 2`
end
if $S_i = \emptyset$ **then**
return $< S_j, j >$
end
if $S_j = \emptyset$ **then**
return $< S_i, i >$
end

6.1 Leave of a Non-parent Group User

Protocol 7 gives the protocol wherein the users of a non-parent group leave. A new group key needs to be generated and the subtree keys should be changed for the subtree of which the leaving user is a part and the same should be communicated to appropriate users. The parent group users are distributed with the changed keys as in [11]. The non-parent group users are given the changed group keys using the new subtree keys generated.

7 Storage Cost Estimation

Suppose a user u is part of a group G_i with n users. Suppose the user is a part of BKT subtree $S_{\log_2 n}$. Then u will have $\log_2 n$ keys and the shared secret with

Algorithm 6. Merge of Binomial Key Tree Queues: MergeBKTQ

Input : $BKTQ_{G_i}$, $BKTQ_{G_j}$, Binomial Key Tree Queues of groups G_i and G_j
Output: $BKTQ_{G_{ij}}$, Merged Binomial Key Tree Queue

Let h_1 be the height of the leftmost BKT in $BKTQ_{G_i}$;
Let h_2 be the height of the leftmost BKT in $BKTQ_{G_j}$;
for $k \leftarrow 0$ *to* $min(h_1, h_2)$ **do**
    $< S, l >= CombineBKT(BKTQ_{G_i}[k], BKTQ_{G_j}[k])$; `// Refer Algorithm 5 for CombineBKT`
    if l=k **then**
        $< S, p >= CombineBKT(BKTQ_{G_{ij}}[l], S;); BKTQ_{G_{ij}}[p] = S$
    else
        $BKTQ_{G_{ij}}[l] = S$;
    end
end
for $k = min(h_1, h_2) \leftarrow$ *to* $max(h_1, h_2)$ **do**
    if $h_1 < h_2$ **then**
        $BKTQ_{G_{ij}}[k] = BKTQ_{G_j}[k]$;
    else
        $BKTQ_{G_{ij}}[k] = BKTQ_{G_i}[k]$;
    end
end

Protocol 7. Leave Protocol for the Non-Parent Group

1 Let S_i, for $i \in [0, h]$ be the subtree that contains the users who want to leave.;
2 Split S_i till the subtree of the users who leave is obtained.;
3 Remove the subtree containing leaving users;
4 Combine the remaining subtrees S_j, for $j \in [0, h]$ using Algorithm 5;
5 KDC will generate new keys for subtrees formed after combine. Also KDC generates the new group key;
6 Identify the subtrees in the combined tree where old subtrees before removal of the users are part.;
7 Distribute the new subtree keys and changed group key with their old subtree common keys.;
8 Distribute the changed group key to parent group users by following the leave protocol in [11].

KDC. Suppose u joins the group G_j, then he will be given the group key of G_j and the subtree key of the new BKT subtree of which he is part. So he will have additional 2 keys per overlapping membership. New intermediate keys need not be generated in G_j. When we make users of a group to have a overlapping membership as a subtree, the join and leave become efficient.

8 Comparison with the Existing Schemes

In this section, we analyze the proposed BKTQ based key managagement scheme for the simultaneous multiple groups with the schemes in [8] (given in Table 1)

Table 1. Comparison of the scheme in [8] and our proposed $BKTQ$ based scheme

	Scheme in [8]		Our Proposed Scheme	
	# Encryptions	# Key Changes	# Encryptions	# Key Changes
Join of a PGU	$2\lceil \log_2 n \rceil + 1$	$\lceil log_2 n \rceil$	2	1
Join of a NPGU	2	1	2	1
Leave of a PGU	$2\lceil \log_2 n \rceil + m - 2$	$\lceil log_2 n \rceil$	$\leq \log_2 n$	$\log_2 n + 1$
Leave of a NPGU	$\leq (m + 2^{\frac{log_2 n - 1}{2}})$	1	$\leq \log_2 m + \log_2 n - 1$	2

and [9] (not given in Table 1) please refer comparison section in [8]). Consider $m = 2^k, k > 0$ groups with each group having $n = 2^t, t > 0$ *parent group users*. In these m groups, every *parent group members* of every group has a overlapping membership with every other group. So in a group, there are $(m-1)n$ *non-parent group members* and n *parent group members*. The results of comparison based on this scenario are provided in Table 1. In Table 1, PGU is Parent Group User and NPGU is Non Parent Group User. As it is depicted in Table 1, our proposed $BKTQ$ based scheme out performs the scheme in [8] and [9].

9 Conclusion

We have proposed an efficient secure group key management scheme for managing the keys in simultaneous multiple groups with overlapping membership. We have proposed a new tree structure, Binomial Key Tree Queue. We have compared our scheme with the schemes in [8] and [9] for rekeying cost. Our results show that, the proposed scheme is efficient. The significant observation is that this efficiency achieved without much increase in storage at the user and KDC. In this scheme a group user will have to only store additional two keys per group for the groups with which he has overlapping membership. In complete paper, we will provide the protocol details with examples and provide the detailed analysis of the protocols for the rekeying cost.

Acknowledgments. This work was supported by Information Security Education and Awareness - ISEA Fellowship, Department of Information Technology, Ministry of Communications and Information Technology and Ministry of Human Resource Development, Government of INDIA.

References

1. Aparna, A., Amberker, B.B.: Secure group communication using binomial trees. In: Third International Symposium on Advanced Networks and Telecommunication Systems, IEEE ANTS (2009)
2. Blundo, C., De Santis, A., Herzberg, A., Kutten, S., Vaccaro, U., Yung, M.: Perfectly-Secure Key Distribution for Dynamic Conferences. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 471–486. Springer, Heidelberg (1993)

3. Wong, C.K., Gouda, M., Lam, S.S.: Secure Group Communication Using key Graphs. IEEE/ACM Transactions on Networking 8, 16–30 (2000)
4. Mittra, S.: Iolus: A framework for Scalable Secure Multicasting. In: ACM SIGCOMM 1997 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, vol. 27, pp. 277–288 (1997)
5. Amir, Y., Kim, Y., Nita-Rotaru, C., Schultz, J., Stanton, J., Tsudik, G.: Secure Group Communication using Robust Contributory Key Agreement. IEEE Transactions on Parallel and Distributed System 15, 468–480 (2004)
6. Burmester, M., Desmedt, Y.: A Secure and Efficient Conference Key Distribution System. In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, pp. 275–286. Springer, Heidelberg (1995)
7. Rafaeli, S., Hutchison, D.: A Survey of Key Management for Secure Group Communication. ACM Computing Surveys 35, 309–329 (2003)
8. Purushothama, B.R., Amberker, B.B.: Group key management scheme for simultaneous multiple groups with overlapped membership. In: Third International Conference on Communication Systems and Networks, COMSNETS, pp. 1–10 (2011)
9. Aparna, A., Amberker, B.B.: Key Management Scheme for Multiple Simultaneous Secure Group Communication. In: IEEE International Conference on Internet Multimedia Systems Architecture and Applications, IMSAA 2009, pp. 1–6 (2009)
10. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms. PHI Publishers, New Delhi (2006)
11. Aparna, A., Amberker, B.B.: A key management scheme for secure group communication using binomial key trees. International Journal of Network Management 20, 383–418 (2010)