Design and Performance Analysis of Scalable and Efficient Group Key Management Scheme [SEGKMS] for Group Communication in Multicast Networks

Saravana Kumar*, N.M. Purusothaman T[1], Lavanya S.[2]

*Department of CSE, Bannari Amman Institute of Technology, Erode, Tamil Nadu, India.
[1]Department of CSE & IT, Government College of Technology, Coimbatore, Tamil Nadu, India.
[2]Department of CSE, Bannari Amman Institute of Technology, Erode, Tamil Nadu, India
E-mail: saravanakumaar2008@gmail.com

Abstract: Key management plays a very important role in the data communications environment. **Problem** The lack of security services, communication overhead, computation overhead, etc enables us to concentrate on creating new innovative ideas. A key distribution algorithm in the reviewed protocols doesn't provide much security in group communication networks. **Proposed** SEGKMS proposes variable length user identity (UID) generation to reduce storage and time requirement. This paper uses one way hash function to generate group key and also it introduces an enhanced key distribution algorithm called proactive secret sharing scheme (PSSS) to ensure efficient group key distribution as a shares. **Result** SEGKMS produces a significant outcome which overcomes security issues through RSA cryptographic algorithm and Proactive Secret Sharing Scheme (PSSS). The concept of static group key reduces overall communication and computation overhead. This papers deals with the analysis of SEGKMS with respect to communication overhead, computation cost, etc. The analysis shows that SEGKMS comprises of the most reliable methods for key generations and hence, the data communication is also scalable. SEGKMS is compared to some of the other key management techniques and proved to be the better choice in this paper.
[Saravana Kumar, N.M. Purusothaman T, Lavanya S. **Design and Performance Analysis of Scalable and Efficient Group Key Management Scheme [SEGKMS] for Group Communication in Multicast Networks.** *Life Sci J* 2013;10(2):1740-1749]. (ISSN:1097-8135). http://www.lifesciencesite.com. 246

Key words: *Key Management,* multicast, public-private key, signature, user identity, rekeying

1. Introduction

Group key management plays an important role in group-oriented security. A group key should be shared among all members in the group in order to multicast information among a certain group securely. Before transmitting every data packages must be encrypted with a shared group key. Only the users with the shared key can decrypt these packages and get the data. If an illegal user receives the package then they cannot decrypt it without the key. Hence, the communication among the members in the group can be said to be secure. Re-keying is done in order to maintain forward secrecy and backward secrecy. Forward secrecy implies that whenever a member leaves the group, the key should be changed and he should not be able to encrypt or decrypt further messages in that group. Backward secrecy implies that a new member joining the group should not be able to encrypt or decrypt the previous messages in that group. Many techniques have been proposed and are in use in current networks.

This paper aims at providing convincing proofs for the scheme SEGKMS (Scalable and Efficient Group Key Management Scheme) in Multicast Networks [1], to be the one among the efficient key management and also performs better than some of the existing key management schemes. In SEGKMS, the group key once generated is not changed. Hence, it reduces the computation cost at any change in the network like member leave and member join. The multicast network is designed and updated in such a way that it supports multiple leaves and joins along with scalability of number of members. The scheme performs well even with a static group key with the help of a well-structured procedures to be followed to guarantee an efficient key management and hence the security of the communication.

2. Related work

Several key management schemes for the multicast networks such as Logical Key Hierarchy (LKH), Hybrid Tree Distributions etc. are listed and explained in [2]. Also they propose a new key management for provid-ing security in dynamic multicast networks that decreases the structure is cluster based and there is a sub-group con-troller that receives the public keys of the valid users from the group controller. A secure multicast key management scheme for cost optimization in case of single sender and multiple receivers is designed in [3]. A new algorithm is proposed to optimize with communication constraints. The scheme uses a hybrid tree scheme in which the storage and the update communication are functions of the cluster size. An elliptic curve cryptosystem-based group [4]

key management for secure group communications to provide security with a small key size is also presented. The scheme is made more efficient with a cluster structure of the communication network. This scheme well-support a single join and leave events. It explains rekeying processes undertaken in a periodic fashion. Secure Group Key Management Scheme for Multicast Networks using Number Theory for providing security to a dynamic multicast networks efficiently [5] makes use of the advantages of the LKH and Chinese Remainder Theorem to provide more effective key management. It also discusses about the security issues in multicast and also few key management techniques and then proposes a new key management scheme. Scalable and Reliable Cost Effective Key Agreement Protocol for Secure Group Communication [6] reduces the cost of computational overhead, number of messages needed during the time of key refreshing and the number of keys stored in servers and members. Computation-and-storage-efficient key tree man-agreement protocol for secure multicast communications [7] manages the key tree structure to maximize the efficiency of the computation and storage costs, and to minimize the increment of the communication cost. It uses Level-homogeneous key tree structure. A survey on key management for multicast is given in [8] which analyzes the problems in key management for multicast and reviews some typical schemes like simple key distribution center (SKDC), group key management protocol (GKMP), scalable multicast key distribution (SMKD), Iolus, and logical key hierarchy (LKH). A survey on group key management is given in [9] which introduces the security problems in multicast-oriented communication, centralized group key management protocols and analyzes the decentralization group key management. The protocols include member driven CGKMP like the GKMP protocol, LKH protocol, OFT protocol, centralized flat table key management (CFKM) protocol and time driven protocol. It explains the join and leave-procedures with respect to the IGKM protocol and time driven protocol. A hybrid scalable group key management approach for large dynamic multicast networks is proposed in [10], which tries to generate and distribute keys to the group members during leave or join of members by using key graph based Boolean minimization technique in order to improve scalability. It uses modified Huffman technique to generate UID for users in the group. It uses Petricks approach to deal with multiple leave of users. Another approach for group key management is given in [11], which utilizes Huffman and Petrick based approaches. Petricks method is utilized for the better performance in case of multi leaves. This work is proposed in order to overcome the large overhead in key distribution in multicast networks. A brief detail about the Diffie-Hellman key ex-change scheme can be obtained in [12] which solve the problem of sharing of a key by two communicating par-ties without an illegal user access to the key. Efficient Key Agreement for Large and Dynamic Multicast Groups is proposed in [13]. It details a scalable, efficient, authenticated group key agreement scheme for large and dynamic multicast systems which is identity-based that uses the bilinear map over the elliptic curves. The computations are explained at members join and leave conditions along with some security aspects. An efficient key management scheme for secure multicast in MANET is presented in [14]. It uses hybrid key management scheme and is proved to be secure way of key management. A survey on Group Key Management in MANETs is provided in [15]. It details the specific challenges to-wards key management protocols for securing multicast communications. It proposes a new key management scheme that is based on a sequential multi-sources model, and takes into account both localization and mobility of nodes, while optimizing energy and bandwidth consumptions. A new secure multicast key distribution protocol using combinatorial Boolean approach is proposed in [16]. It is based on Key Management using Boolean Function Minimization (KM-BFM) technique. This technique is compared with the other approaches and is proven to be efficient with respect to the communication overhead. A scalable and distributed security protocol for multicast communications is presented in [17]. It is based on the Iolus and the logical key hierarchy proto-cols. This approach is proven to reduce complexities in member leave and member joins. A key management scheme for high bandwidth secure multicast is presented in [18]. It concentrates on re-keying algorithms based on the Logical Key Hierarchy (LKH) and also reduces the longest sequence of encryptions and decryptions that need to be done in a re-keying operation.

3. Materials and methods

The multicast network in SEGKMS is a time-based cluster structure. Initially *Key Generation Center/Group Controller* (*KGC/GC*) assigns the number of sub-groups (cluster) to be constructed and their respective subscription span values based on which the members are grouped. Consider the number of sub-groups under *KGC/GC* to be 3 whose subscription spans are 30 days, 6 months and 1 year respectively is shown in Figure 1. Here *SGC1*, *SGC2*, and *SGC3* are sub-group controllers.

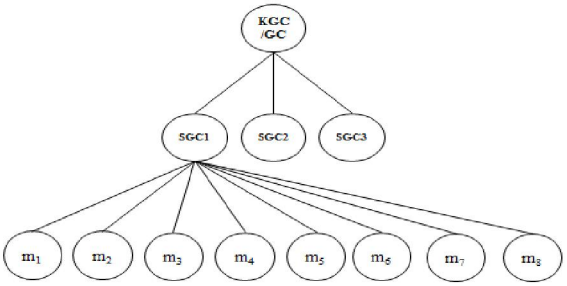

Figure 1. SEGKMS - Sample network

SEGKMS is of four steps: generation of *UID*s, Generation of group keys and sub-group keys, Generation of public keys and private keys along with signature for each member, and the communication between the members. There are two databases maintained by the *KGC/GC*: Existing member database that stores the data about members under each *SGC*. This data includes member *UID*s, subscription spans of each member; and Leaving members database that contains the data of the members which left previously. UID for each member in the sub-group is generated using Modified Huffman Coding technique [19].

This method is efficient only for a group where the user leave is predictable based on the probability values and does not support multiple leaves/joins in a group with variable key lengths. Initially, *KGC/GC* will randomly generate unique binary values to *SGC1*, *SGC2*, and *SGC3*.

Each *SGC* will assign *UID* values for its members. By combining the commonly received binary value from *KGC/GC* and the uniquely generated binary values from *SGC*, the final original UID is formed and assigned. SEGKMS also supports multiple leaves and joins with the same *UID* generation technique. The group key computation method used in [20] uses multi-party Diffie-Hellman and TGDH protocol to generate group keys. SEGKMS made the group key GK independent of sub-group key SGK. For the generation of SGK, each member under a sub-group sends the partial key, $f^{L_{i,j}}$ to *SGC1*, where $i = 1, 2, 3, 4, \ldots$ and $j = 1, 2, 3, \ldots$. The SGC generates its own partial keys and then uses these partial keys to compute the sub-group keys (*SGK*s). Here, f is the generator of the multiplicative group, Z_N^* which is the set $1, 2, \ldots, N-1$, N is the prime and L is a randomly chosen prime number for respective member. The resulting sub-group key of *SGC* is given by Equation 1.

$$SGK_1 = f^{L_{1,j}L_{2,j}L_{3,j}L_{4,j}\ldots\ldots\ldots L_{i,j}K_j} \quad (1)$$

The resulting *SGK* is sent to each member and is used for encryption and decryption of the message exchange among the members within the sub-group.

To generate a group key (GK), the *KGC/GC* collects the partial key of each sub- group. Consider Figure 1. Let partial keys of *SGC*s be f^{K_1}, f^{K_2} and f^{K_3} respectively. The *KGC/GC* receives $f^{K_1K_2K_3}$ and the *GK* is computed by *KGC/GC* by adding its own partial keys as shown in Equation 2.

$$GK = f^{K_1K_2K_3K_{GC}} \quad (2)$$

This Group key is broadcast to each sub-group which is used for decryption or encryption during the communication between different sub-groups under *KGC/GC*. The *SGK*s and *GK* are distributed in this network using proactive secret sharing scheme [21]. For each *GK* and *SGK* to be distributed to the sub-groups and the members, a time periods, T_{GK} and T_{SGK}, are set and divided into periods of time. Here, a proactive threshold scheme is applied, say $(r + 1, t)$, where t is the number of time periods and $r + 1$ is the number of locations, say routers on the way between the sender and receivers, to be compromised by the adversary, who tries to learn the *GK* or *SGK*, in a single time period which is difficult as at the end of each time period, the share become obsolete and has to be erased. It is even difficult to distrust the secret by the adversary as $t - r$ shares are to be corrupted in a single period of time.

Each user is given long-term public and private keys. The *KGC/GC* randomly chooses a secret key and the computes and publishes the corresponding public key. SEGKMS uses the idea of RSA to construct a private-public key pair, where the *KGC/GC* calculates (1) public key *(M, E),* where *M* is the product of any two large prime numbers, *a* and *b*, and *E* is the number prime with respect to *M* and (2) private key *(a, b, d, φ(M))*, where *d* is the part of private key of *KGC/GC* and is equal to e^{-1} *mod φ(M)*. The *KGC/GC* determines a primitive element *α* in *GF(a)* and *GF(b)*. Then it chooses a one-way hash function. Here, *(α, h())* is a public information where *h()* gives unique output for different input.

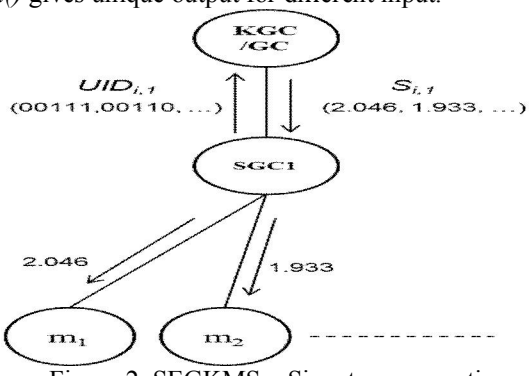

Figure 2. SEGKMS – Signature generation

Each *SGC* provides *UID*s of the member under it to the *KGC/GC* to obtain the signature $S_{i,j}$ for each $UID_{i,j}$ of a member $m_{i,j}$, where *i = 1, 2, 3, ...*, represents each member and *j = 1, 2, or 3* represents the *SGC*. If *KGC/GC* confirms the correctness and the relationship between $m_{i,j}$ and $UID_{i,j}$, then it calculates $S_{i,j}$ using Equation 3 and distributes $S_{i,j}$ to each *SGC* where each *SGC* distributes them to the respective members as shown in the Figure 2.

$$S_{i,j} = UID_i^d \bmod M \quad (3)$$

Both public-private keys pair and signatures are distributed using proactive secret sharing scheme. The communication between two members, $m_{i,j}$ and $m_{k,j}$, is as follows:

1) $m_{i,j}$ selects a random number $R_{i,j}$ and computes two public keys $x_{i,j}$ and $y_{i,j}$ as follows:

$$x_{i,j} = S_{i,j}.\alpha^{R_{i,j}} \bmod M \quad (4)$$

$$y_{i,j} = R_{i,j}^e \bmod M \quad (5)$$

2) $m_{i,j}$ uses a timestamp $T_{i,j}$ and the identification number $UID_{k,j}$ of the member $m_{k,j}$ for the operation of one-way function of $h(x_{i,j}, y_{i,j}, T_{i,j}, UID_{i,j})$, then computes

$$P_{i,j} = S_{i,j}.R_{i,j}^{h(x_{i,j},y_{i,j},T_{i,j},UID_{k,j})} \bmod M \quad (6)$$

3) $m_{i,j}$ sends $(UID_{k,j}, x_{i,j}, y_{i,j}, T_{k,j})$ to $UID_{k,j}$.

Similarly, member $m_{k,j}$ selects the random number $R_{k,j}$ and the timestamp $T_{k,j}$, then computes $x_{k,j}$, $y_{k,j}$, and $P_{k,j}$ i.e.

$$P_{k,j} = S_{k,j}.R_{k,j}^{h(x_{k,j},y_{k,j},T_{k,j},UID_{i,j})} \bmod M \quad (7)$$

and sends $(UID_{k,j}, x_{k,j}, y_{k,j}, T_{k,j})$ to $UID_{i,j}$. Each member sends this information through the *SGC*s. Before session key (*SK*) generation, $UID_{i,j}$ and $UID_{k,j}$ have to verify whether $(UID_{k,j}, x_{i,j}, y_{i,j}, T_{k,j})$ and $(UID_{k,j}, x_{k,j}, y_{k,j}, T_{k,j})$ are sent from members $m_{i,j}$ and $m_{k,j}$ respectively. It is done by checking

$$P_{k,j}^e = UID_{k,j}.y_{k,j}^{h(x_{k,j},y_{k,j},T_{k,j},UID_{i,j})} \bmod M \quad (8)$$

Consider $P_{k,j}$ from Equation 8. From Equation 7,

$$P_{k,j}^e = (UID_{k,j}^d \bmod M)^e.(R_{k,j}^{h(x_{k,j},y_{k,j},T_{k,j},UID_{i,j})} \bmod M)^e$$

Mathematically, $(G^x \bmod n)^y = (G^y \bmod n)^x = G^{xy} \bmod n$ and $(G^x \bmod n) \bmod n = (G^x \bmod n)$ because n is a very large number. According to RSA, $d = e-1 \bmod \varphi(n)$ and $d * e = 1 \bmod \varphi(n) = 1$

$$P_{k,j}^e = UID_{k,j}.y_{k,j}^{h(x_{k,j},y_{k,j},T_{k,j},UID_{i,j})} \bmod M$$

which is similar to Equation 9. Similarly, member verify at his end that

$$P_{k,j}^e = UID_{i,j}.y_{i,j}^{h(x_{i,j},y_{i,j},T_{i,j},UID_{k,j})} \bmod M \quad (9)$$

The communicating members compute a secret session key (*SK*). The computation of *SK*s is as follows: Consider the communication between two members $m_{i,j}$ and $m_{k,j}$. They compute the secret *SK*s $SK_{i,j}$ and $SK_{k,j}$ respectively as follows [22]:

$$SK_{i,j} = (\frac{x_{k,j}^e}{UID_{k,j}})_{k,j}^R \bmod M \quad (10)$$

$$SK_{k,j} = (\frac{x_{i,j}^e}{UID_{i,j}})_{i,j}^R \bmod M \quad (11)$$

$SK_{i,j}$ and $SK_{k,j}$ are the same for the communicating members. Hence,

$$SK_{i,j} = SK_{k,j} = \alpha^{e*R1*R2} \bmod M \quad (12)$$

Since, M is a very large, the above equation can be written as

$$SK_{i,j} = SK_{k,j} = \alpha^{e*R1*R2} \quad (13)$$

Using these session keys, the members communicate successfully with each other.

4. Rekeying

In SEGKMS, any member may leave or join the sub-group at any time. Whenever there is any change in the number of members in a sub-group, rekeying is done. When a member's subscription span is completed, the data about this member in the database of *KGC/GC* is removed and inserted in the leaving member database. The signatures, public-private keys and the group key of the other members remain same. Only the sub-group key is changed as explained in the previous sections.

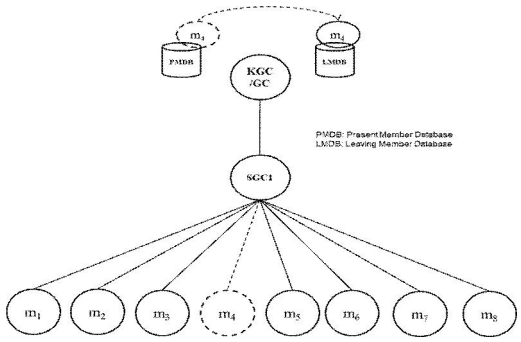

Figure 3. SEGKMS: Single-member leave

In Figure 3, member $m_{4,1}$ (m_4 of sub-group *SGC1*) leaves its sub-group and hence, the respective *SGC1* generates a new *SGK1*. All its data are shifted

to the leaving member database for future reference. If this member tries to communicate with existing members in the group, there are two cases.

1) When a member tries to communicate with the member in his previous sub-group, the member should have the valid sub-group key. But it is changed as soon as the member left the group. Hence, this communication fails

2) When a member tries to communicate with the member in other sub-group other than his previous sub-group, the member's request for communication should go through *KGC/GC* using *UID*. The *UID* does not exist in database and hence the request rejected. Hence, forward secrecy is achieved in SEGKMS. Further, the member who left can subscribe to the group again. But, the member is given new *UID* and the keys.

When a new member joins, his data including the UID is stored in the existing members' database at *KGC/GC*. Then the process of key generation, signature generation, and communication can be done as explained in previous sections. Let a new member $m_{9,1}$ join the sub-group *SGC1* (Figure 4). Then *SGC1* gives a new *UID* for $m_{9,1}$ i.e. $UID_{9,1}$ which is stored in the existing member database at *KGC/GC*.

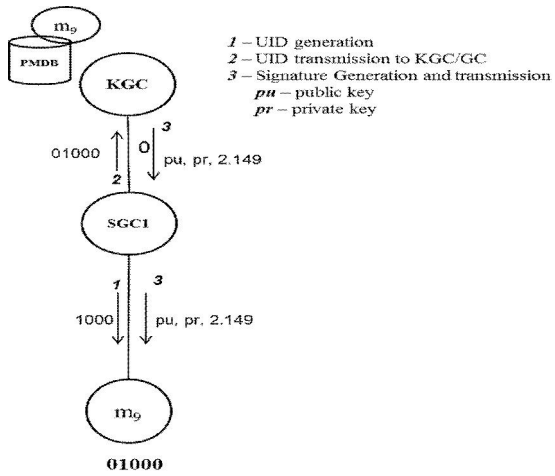

Figure 4. SEGKMS – Single member join

The new member may try to access the data exchanges before its joining. Here, as the *KGC/GC* has the knowledge of the subscription span of $m_{9,1}$, it compares the time of joining with the time of the data exchange that it is trying to access. The time does not fall under the member's subscription span. Hence, the request is rejected.

Two or more members may complete their subscription span at the same time. The data about these members are removed and placed in the leaving members' database at *KGC/GC*. Forward secrecy strategy is maintained as explained previously. The multiple leaves may be from the same sub-group or may be from different sub-group.

When multiple leaves take place in the same sub-group, only the members of that sub-group are given new *SGK*s and the other procedures follows as explained in previous sections. Let the members, $m_{1,1}$ and $m_{5,1}$, be the leaving-members as indicated in Figure 5.

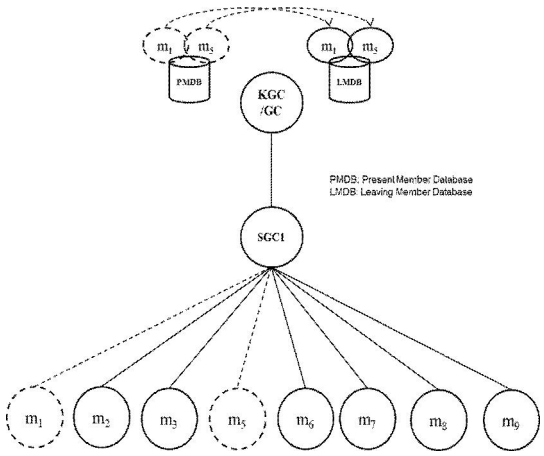

Figure 5. Multiple member leaves from single group

As there is a change in number of members under *SGC1*, it gets the partial keys from currently existing members and generates a new *SGK1* and distributes it to the group members.

When the members of different sub-groups leave at the same time, the keys will be changed for each sub-group where leave takes place. Let $m_{2,1}$ be the member leaving from *SGC1* and $m_{1,2}$ be the member leaving from *SGC2* as shown in the Figure 6.

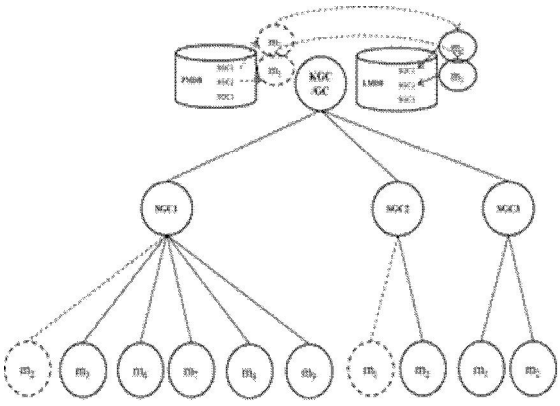

Figure 6. Multiple member leaves from multiple groups

Both the sub-groups notice the change in the number of members and hence *SGC1* and *SGC2*

generate a new *SGK*s such as *SGK1* and *SGK2* and distribute to the members.

The member join under a particular sub-group depends on the subscription span of a member. Backward secrecy strategy is maintained as explained previously.

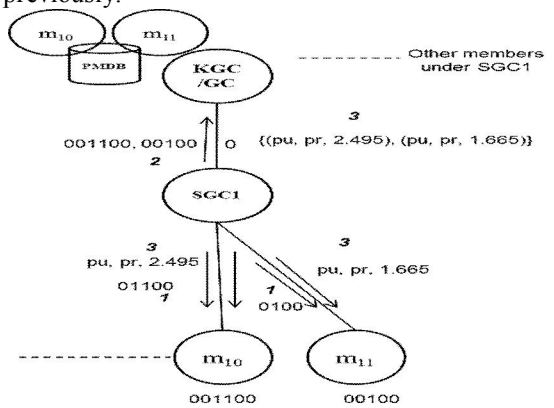

Figure 7. SEGKMS – Multiple joins in SGC1

When multiple members are to be joined, their subscription span may fall under same sub-group or different sub-groups. Two or more members may join the same sub-group at the same time. Let, members, $m_{10,1}$ and $m_{11,1}$ join *SGC1* at the same time (Figure 7).

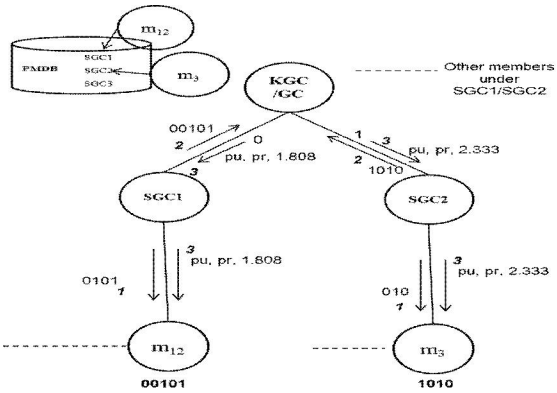

Figure 8. SEGKMS – Member joins in SGC1 and SGC2

SGC1 generates *UID*s for these members. These *UID*s and their subscription spans are stored in *KGC/GC* database. *SGC1* generates a new *SGK1* and distributes to the members under it. The group key, *GK* is given to the member.

Two or more members join the multicast group at the same time. Let members $m_{12,1}$ and $m_{3,2}$ join the sub- groups *SGC1* and *SGC2* at the same time as shown in Figure 8. Both SGCs generate a new *UID* for the newly joined member and then sends data about their *UID*s and the subscription spans to the *KGC/GC* which is stored in database.

KGC/GC then generates new public-private key and signatures for the newly joined members. Then the group key is sent to the newly joined members.

5. Analysis and comparison

Before the *UID* generation under any Sub-group, the subscriptions spans of all the members willing to join the group initially are to be sorted in increasing order. A simple sorting operation requires minimum *N* number of comparisons and maximum N^2 where *'N'* is the number of members. Further, *UID*s are generated using Modified Huffman Coding technique. It operates in *O(N log N)* times [19]. If the subscription spans are already sorted and then this coding technique applied, then it operates in *O(N)* time where the sorting takes *O(N log N)* times.

Let the number of Sub-group Controllers (SGCs) under Group Controller (GC) be *C* and number of members under any SGC be N_i, where, $i=1,2...,C$. The GC uses the partial keys received from each SGC and also its own partial key to generate *GK*. Hence it takes *O(log C+1)* operations for GK generation. In the same way, SGC uses partial keys of its members and also its own to generate SGK. Hence, it takes *O(log N+1)* operations for SGK generation.

SEGKMS uses RSA concept for public-private key generation. If *K* is the number of bits in modulus *M* then public key operations takes $O(K^2)$ steps and private key operations take $O(K^3)$ steps. Whenever a change occurs in the number of members under a sub-group, it may be because of a member-leave or member-join. SEGKMS supports multiple join and leaves at the same time ensuring both forward and backward secrecies. The group key once generated, will not be changed even when leaves and joins occur. Only the cluster key will be changed during leave and join events. Further, the *UID*s for the new members along with the public-private key pair and signature will be generated.

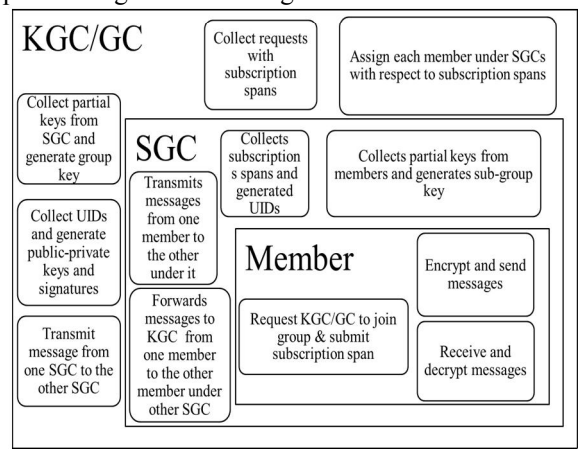

Figure 9. SEGKMS: Operations involved

Further the complexities and the overheads depend on the number of members present in the new sub-group. Operations involved in each component of SEGKMS are summarized in Figure 9.

The first scheme in comparison to SEGKMS is one-way function trees for key establishment in large dynamic groups. There are several points in one-way function trees (OFT) compared to which SEGKMS performs better.

- SEGKMS prefers top-down approach achieving advantages in both storage requirements and the key broadcasts. OFT can be used for both top-down and bottom-up references, where the former is used to reduce storage requirement of information and the latter is used to reduce rekeying broadcasts to about *lg n* keys.
- OFT uses binary tree structure and assigns randomly chosen keys for the users. SEGKMS UID tree can accommodate new nodes based on the subscription spans of the new user.
- In OFT there is a split in the leaf node of the tree making room for new member along with change in keys of the sharing sibling along with new member. In SEGKMS, when a new member joins, only the sub-group key of the other member falling under his group are changed to ensure forward and backward secrecies. All other keys of the existing users remain same.
- In OFT, when a member leaves, sibling of the leaving member is reassigned with new parent causing in change of the keys. In SEGKMS, when a member leaves, the siblings still remain under same parent and the sub-group controller causing change in only cluster key to ensure forward and backward secrecies.

When compared to LKH (Logical Key Hierarchy) [2], the following points can be noticed.

- In LKH, the degree of the LKH tree is constant and the number of members under each sub-group is constant. In case of arrival of new members with time span falling under any sub-group and the respective sub-group is full, it causes in the formation of new root node that results in the addition of new sub-group and also the possibility of doubling the number of members. In SEGKMS, this is achieved with the constant number of sub-groups. The usage of Modified Huffman Coding technique helps the addition of new members in the same group with the flexibility in the group size.
- In LKH, the group key and all the other keys for the members are regenerated at every member leave/join whereas in SEGKMS, the group key remains same for any change in number of members and also ensures backward and forward secrecies with the help of databases used.
- In LKH, each member stores the set of keys that store the path from the root node. Hence key storage is *O(log N+1)* where N is the sub-group size. In SEGKMS a member uses the cluster key and the session key for the communication.

SEGKMS is also compared to some of the other approaches: Group Key Management (GKM) approach for multicast cryptosystems , and energy efficient topology aware key management scheme (TAKM). Following are the points noticed in comparison with these approaches:

- PGKM operates on a static network structure with a limit on number of members under each controller; SGK uses a cluster-based network structure with an equal number of members in each cluster. The number of members is defined before dividing the network into cluster; in TAKM, members are grouped according to their physical distance. But it may have a disadvantage in grouping a single member who is existing in a farther distance from majority of members; In SEGKMS, the network structure is dynamic i.e. the number of members under each sub-group differ depending upon the number of join occurring in the network eventually.
- PGKM uses Chinese Remainder Theorem and a hierarchical graph. So, each members contains a key and a modulus; In TAKM all members compute their group key i.e. group key generation is the responsibility of the members; In SGK, since it uses CRT, the sub-group controller has to compute the common solution *X* which again has to be multicast to all the members under it; In SEGKMS, generation of group key is dome by the group controller and only once. There is less computation overhead on the members compared to other approaches.

In addition, the key storage value at server in PGKMP is *2n-1* and at member is *log 2n+1* which are *n+1* and *3* respectively for SEGKMS.

6. Results and Discussions

SEGKMS is compared to LKH and OFT and the following results were obtained.

Whenever a member leaves or joins, there is a change database which is notified to sub-group controller once for each change. Hence, only one message is sufficient for any change in the network. The number of messages exchanged at any change in a group 1. It contains data about the members left/joined and the current status of the sub-group. Then the computation cost goes up to *O(1)* for SEGKMS. Table 1 presents the communication costs of other approaches compared to SEGKMS.

Table 1. Communication Cost

Protocols	Join	Leave
OFT	$\log_2$ (n) +1	$\log_2$ (n)+1
LKH	2 $\log_2$ (n)-1	$\log_2$ (n)
SEGKMS	1	1

Figure 10 and 11 shows the statistical representation of the above analytical measures.

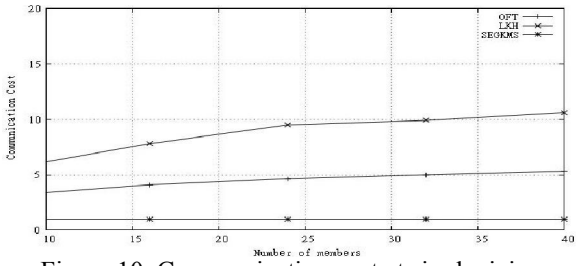

Figure 10. Communication cost at single joins

Like all other approaches, there is a rekeying process in SEGKMS, except that the group key remains same for any change in the number of members along with ensuring securities necessary. The rekeying is done only in the sub-group where the change takes place.

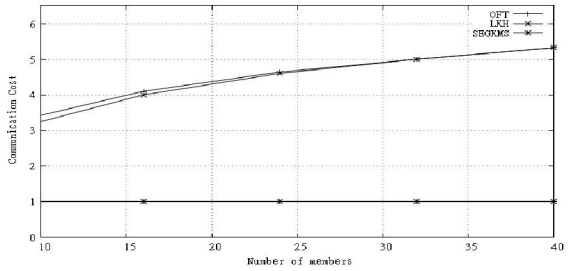

Figure 11. Communication cost at single leaves

Table 2 gives a brief comparison of computation costs for group key generation of OFT, LKH and SEGKMS for a single member join and leave. Figure 12 and 13 gives the statistical comparisons. In SEGKMS, the group key is not changed once it is generated. Whereas, in OFT and LKH, the group key is regenerated every time there is a change in the number of users.

Table 2. Computation Cost

Protocols	Join	Leave
OFT	$\log_2$ (n)+1	$\log_2$ (n)+1
LKH	2 $\log_2$ (n)-1	2 $\log_2$ (n)
SEGKMS	1	1

Here, *m*, is the sub-group size and *n* is the group size. When a member joins the group, the sub-group key is regenerated along with the UID, Public-privates keys pair and signature. For each newly joining member three new keys and one UID is generated. If there are *k* members joining at the same time, then *4k* computations are done. Here *K* is a constant for any number of members since it depends only on the members joining or leaving but not on the members in the sub-group.

Whenever a member leaves the group, only the *SGK* is regenerated. Hence, if *n* members are leaving the group, *n* times the *SGK* is generated. If they leave at the same time, only one *SGK* is regenerated. If joining and leaving are occurred at the same time, the number of computations done is only 1 where, *1* indicates the *SGK* generation.

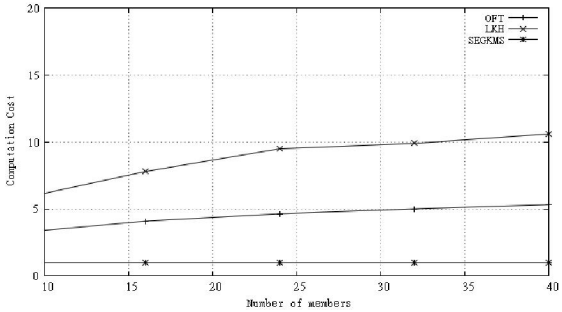

Figure 12. Computation cost at multi joins

Whenever a member joins or leaves, the number of computations done is given in previous sections.

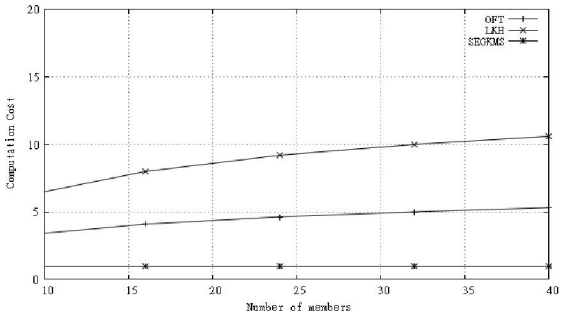

Figure 13. Computation cost at multi leaves

The comparison of number of rekey messages at single join and leave is given in Table 3. Figure 14 and 15 show the statistical analysis.

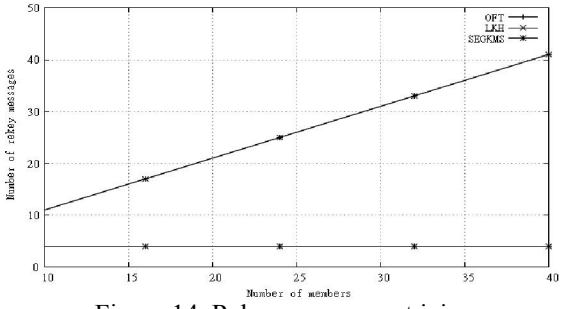

Figure 14. Rekey messages at joins

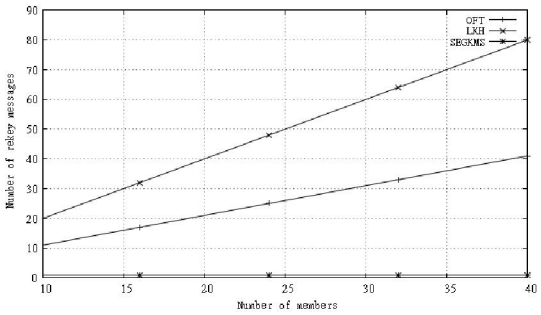

Figure 15. Rekey messages at leave operation

Table 3: Rekey Messages Needed

Protocols	Join	Leave
OFT	n+1	n+1
LKH	n+1	2n
SEGKMS	4	1

In SEGKMS, there are databases to maintain the details of both existing members and the past members.

Table 4: Key storage at join and leave

Protocol	Join	Leave
OFT	2n	$2\log_2(n)+1$
LKH	2n	$\log_2(n)+1$
SEGKMS	n+1	3

Considering there are *n* members which includes both existing and non-existing members, the key storage is given in Table 4.

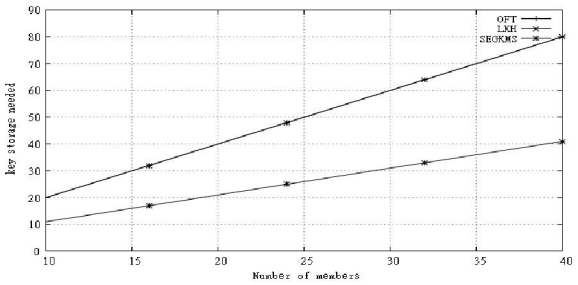

Figure 16: Key storage at join operation

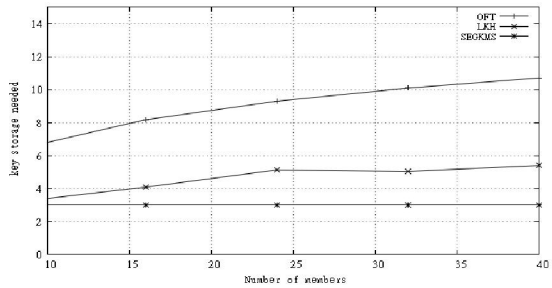

Figure 17: Key storage at leave operation

In SEGKMS, a member stores just three keys but the KGC/GC (server) stores *n+ 1* keys. Figure 16 and 17 show the statistical analysis.

7. Conclusion

SEGKMS is a key management system for multicast networks that results in security of the data exchanges. It is cost efficient with respect to the distribution of the keys between network elements. It is a systematic approach for the key management in a multicast network to achieve a great advantage in terms of scalability, forward secrecy, backward secrecy, key independence, etc. The number of sub-groups is constant and hence the group key remains same throughout. Hence, *GK* is generated and distributed only once. SEGKMS uses proactive secret sharing scheme which is proven to be efficient for distribution of the keys. The future work aims at enhancing SEGKMS by introducing the proactive variable secret sharing schemes for robustness of sharing and detection of adversaries. Compared to existing key management schemes, SEGKMS aims at providing a better key management technique with the use of the static group key. It uses a variable length signatures, and session keys which help in avoiding possible data access attacks. From the analysis and comparisons done in this paper, it is clear that the SEGKMS gives well organized key management for data communications in multicast networks.

References

[1] N. M. Saravana Kumar and T. Purusothaman, "SEGKMS: Scalable and Efficient Group Key Management Scheme in Multicast Networks". *European Journal of Scientific Research*, ISSN 1450-216X Vol. 89 No 3 October, 2012, pp.394-408.

[2] R. Srinivasan, V. Vaidehi, R. Rajaraman, S. Kanagaraj, 2010. Secure Group Key Management Scheme for Multicast Networks. International Journal of Network Security, Vol.11, No.1, PP.33-38.

[3] Mingyan, Li., R. Poovendran, C. Berenstein, 2002. De-sign of Secure Multicast Key Management Schemes With Communication Budget Constraint. IEEE Communications Letters, Vol. 6, No. 3.

[4] Pitipatana, S., Nirwan A., 2007. Elliptic Curve Cryptosys-tem-Based Group Key Management For Secure Group Communications. IEEE Military Communica-tions Conference. doi: 10.1109/MILCOM.2007.4455002.

[5] E.Munivel, J.Lokesh,"Design of Secure Group Key Man-agement Scheme for Multicast Networks using Num-ber Theory". CIMCA, IAWTIC, and ISE. 2008.

[6] S. Jabeen Begum and Dr.T. Purusothaman, 2011. A New Scalable and Reliable Cost Effective Key Agreement Protocol for Secure Group Communication. Journal of Computer Science 7 (3): 328-340.

[7] Xukai Zou, Byrav Ramamurthy, Spyros Magliveras, 2002. Efficient Key Management f or Secure Group Communications with Bursty Behavior. In Proceed-ings of the IASTED International Conference, Communications, Internet, and Information Technology.

[8] Shu-Quan Li, Yue Wu, "A Survey on Key

Management for Multicast". Second International Conference on Information Technology and Computer Science, 2010.
[9] Bibo Jiang, Xiulin Hu, "A Survey of Group Key Man-agement". International Conference on Computer Science and Software Engineering. 2008.
[10] T. Srinivasan, S.Sathish, R.Vi,jay Kumar, M.V.B.Vi,jayender, "A Hybrid Scalable Group Key Management Approach for Large Dynamic Mul-ticast Networks". The Sixth IEEE International Conference on Computer and Information Technology. 2006.
[11] Senthamil Ilango, Johnson Thomas, "Group Key Man-agement utilizing Huffman and Petrick based ap-proaches". Proceedings of the International Confe-rence on Information Technology: Coding and Com-puting (ITCC'04). 2004.
[12] Liming Wang, Chuan-Kun Wu, "Efficient Key Agree-ment for Large and Dynamic Multicast Groups". In-ternational Journal of Network Security, Vol.3, No.1, PP .8–17, July 2006 http://isrc.nchu.edu.tw/ijns/.
[13] J. Lakshmana perumal, K.Thanushkodi, N.M.Saravanakumar, K.Saravanan, G.Vigesh T.Purusothaman "Efficient Key Management Scheme for Secure Mul-ticast in MANET". IJCSNS International Journal of Computer Science and Network Security, VOL.10 No.11, November 2010.
[14] Mohamed-Salah Bouassida, Isabelle Chrisment, Olivier Festor, "Group Key Management in MANETs". In-ternational Journal of Network Security, Vol.6, No.1, PP .67–79, Jan. 2008.
[15] Mohamed M. Nasreldin Rasslan, Yasser H. Dakroury, Heba K. Aslan, "A New Secure Multicast Key Distri-bution Protocol Using Combinatorial Boolean Ap-proach".

17/5/2013

International Journal of Network Security, Vol.8, No.1, PP .75–89, Jan. 2009.
[16] Imane Aly Saroit, Said Fathy El-Zoghdy, Mostafa Matar, "A Scalable and Distributed Security Protocol for Multicast Communications". International Journal of Network Security, Vol.12, No.2, PP.61{74, Mar. 2011.
[17] Roberto Di Pietro, Luigi V. Mancini, Alessandro Mei, "Key management for high bandwidth secure multi-cast". Journal of Computer Security, 693–709, 2004.
[18] Anil Kapil, Sanjeev Rana, "Identity-Based Key Manage-ment in MANETs using Public Key Cryptography". International Journal of Security (I,JS), Vol(3), Is-sue(1), 2009.
[19] A.Herzberg, S.Jarecki, H.Krawczyk, M.Yung, "Proactive Secret Sharing or How to COpe with Perpetual Lea-kage". CRYPTO '95 Proceedings of the 15th Annual International Cryptology Conference on Advances in Cryptology, pp-339-352, Springer-Verlag.
[20] David A, McGrew and Alan T. Sherman, 2003. Key es-tablishment in large dynamic groups using one-way function trees. IEEE Transactions on Software En-gineering, doi: 10.1109/TSE.2003. 1199 073.
[21] M .V.Vijaya Saradhi and B H.Ravi Krishna 2009. A group key management approach for multicast cryp-tosystems. Journal of Theoretical and Applied Infor-mation Technology.
[22] Pavithira Loganathan and T. Purushothaman 2011. An Energy Efficient Topology Aware Key Management Scheme for Multicasting in Ad-hoc Networks. Inter-national Journal of Wisdom Based Computing.