Scalable rekeying limited to subgroup using Hybrid key trees

Dr. V. Valli Kumari
Andhra University
AU College of Engineering
Visakhapatnam, AP, India

D.V. Naga Raju
Shri Vishnu Engg College for Women
Vishnupur
Bhimavaram, AP, India

Dr. K.V.S.V.N Raju
Andhra University
AU College of Engineering
Visakhapatnam, AP, India

ABSTRACT
Maintaining security is a critical issue in any group communication protocols. The objective of security in a group communication is to ensure the access only to the legitimate members of the multicast group. The entry and eviction of the members are the main criteria to change the group key and to give them more assurance of a secret communication, which is known as re-keying. Since it is a frequently performed activity during a communication, the group key updating need to be done in a scalable and efficient manner. Earlier, client–server paradigm is the most predominantly used technique for applications like conferencing, chat groups, interactive video gaming, etc which use the concept of unicast for the transmission of data. Present day advancements in the Internet technologies, especially the increase of bandwidth are definitely encouraging environment for new developments. Unlike the old communication models, where the delivery of the packets are to be carried out in an unicast model, multicasting technique provides an efficient delivery service to larger user-community with effective and efficient network resource utilization.

In the earlier schemes proposed for rekeying mechanism like LKH [4], FDLKH [7], DLPKH [9], the entire group will be disturbed with change in the membership. This paper proposes two new ideas: one with an objective of efficient re-keying and the other with an objective of disturbing only a subset of the group. Both the ideas don't need the secure channel for the distribution of the key material like [4],[7] and also does not reveal the private keys like [9]. The number of keys maintained at each member in this scheme, number of messages sent, size of the messages and number of encryption and decryptions are always constant unlike the other schemes which typically depends on the height of the tree.

General Terms
Management, Performance, Reliability, Security.

Keywords
Multicast group, re-keying, group communication, secure channel.

1. INTRODUCTION
History is filled with examples of how technology helped usher a new eras of prosperity. Efforts are on streamline technology widening the knowledge canvas. This demands innovation and precision at every juncture. Toeing the identical line of perfection this paper focuses on key distribution using hybrid key trees for reducing disturbed members. In the present competition of cutting edge technology, quality upgradation and improving accessibility are the demanding requisites. Any new idea ensuring viability reserves the glory of practicality in the world of computer science. With the increase of bandwidth along with several latest advancements in the internet technologies encouraging people for the development of new services like secure video conferencing, interactive gaming, stock quotes distribution which are based upon a group communications model where packets need to be delivered to a large community from one or more sources.

Earlier uni-casting is the prominently used technique for many of such applications. Network resources can be best utilized with multicasting for group communication. Providing security is a challenging issue with the dynamic nature of the membership in any group communication model. The activity of key refreshment must be performed very often to maintain the forward and backward secrecy. In particular, cost for key establishment and key renewing is usually relevant to the group size and subsequently becomes a performance bottleneck in achieving scalability.

Group key management is the most important issue in secure group communication. The existing Group key management protocols fall into three classes: centralized group key management, decentralized group key management, and contributory group key agreement. Yacine Challal et al [16], Sandro Rafaeli et al [17] conducted an excellent survey on these classes.

In Centralized group key Management, a single entity known as central server takes the responsibility of key generation and distribution. In this scheme each user is associated with two keys: one is the group key which one shares with the group members and an individual key which one shares with the central server. A higher degree of security and efficiency will be there with the centralized schemes but it suffers from a major disadvantage like dependency on a single entity, which can be a single point of failure. If the central server is compromised all the keys will be known to the adversaries. Single point of failure and key compromization are solved in the decentralized group key management protocols with the transfer of the burden of generation and distribution of keys to dynamically selected members. This scheme also suffers from the possibility generation of weak keys.By taking the share of the members in the generation of the key the key agreement protocols reduces the chances of producing the weak keys.

Any group communication model should fulfill the forward and backward secrecy, collusion freedom, key independence, low bandwidth overhead, 1-affects-n and minimum storage requirements efficiently and effectively. Forward secrecy in the sense, a member who comes out of the group should not be able to decrypt the future communications. When a member newly joins a group should not be allowed to know old the messages. Forward and backward secrecies can be achieved by refreshing the group key. Collusion freedom means that the current group key cannot be deduced by the combined work of the evicted members by sharing their individual information. According to key independence, one key can not be deduced from other keys .According to low bandwidth overhead, the no of rekey messages should be minimum. As per the 1-affects-n: all the other group members should not be affected with a single membership change and also minimum storage requirements. This paper proposes a new idea which concentrates on 1-affects-n property.

2. RELATED WORK

Harney and Muckenhirn[1, 2] propose a group key management protocol which uses the concept of pair wise keys. The major weakness of this scheme is that it requires O(n) rekey message in case of join. Chinese Remainder Theorem based scheme proposed by Chiou and Chen [3] suffers from the need of high computational resources on the central server. C. K. Wong, M. Gouda, and S. S. Lam [4] proposed Local Key Hierarchy (LKH) where a single entity known as Central server maintains a tree of keys. Each node is associated with a KEK (key encryption key) and the members who occupy at leaf node holds secret keys shared with other members of the group. Each member knows all the KEKs corresponding to the nodes in the path from its leaf to the root .The key corresponding to the root of the tree is the TEK (traffic encryption key).For a balanced binary tree, each member stores at most 1+log2 (n) keys, where n is the number of group members. This key hierarchy allows reducing the number of re-key messages to O (log n) instead of O (n) in GKMP. McGrew and Sherman [5] proposed an improvement over LKH, called One-way Function Trees (OFT). OFT allows reducing the number of re-key messages from 2log2 (n) to only log2 (n). DeCleene et al. [6] proposed Intra-domain Group Key Management Protocol (IGKMP) which also suffers from a single point of failure. Inoue and kuroda [7] have proposed the Fully Distributed Logical public key hierarchy (FDLKH), in which they used the concept of LKH without any central server. Moreover, in FDLKH the members will not have any individual keys unlike LKH. For each subtree the task of generating and distributing of key material will be assigned to group members known as captains. The captains use DH key agreement. Rafeli and Hutchison [8] proposed Hydra protocol. In this scheme for each sub group a controller (Hydra server) will be there which controls that subgroup. If a member joins a group or leaves from a particular group, the hydra server who is responsible for that subgroup in which the event has happened, takes the responsibility of generating and distributing to the other HS involved in that session. Rakesh Bobba,Himamshu Khurana[9] proposed DLPKH: Distributed Logical Public Key Hierarchy where they also followed the concept of Logical Key Hierarchy. In this scheme they used public key trees for the secure distribution of the updated keys which have the advantage of secure distribution of the keys without establishing any secure channel. In this scheme, each member who occupies at leaf node will know all the private and public keys of their ancestor nodes and also the public keys of the nodes that are siblings to the set of nodes on the path from the leaf to the root. The responsibility of generation and distribution of keys is given to the sponsors and co-sponsors. The main objectionable issue with this scheme is that the private keys of ancestor nodes will be revealed which is totally against the concept of public key cryptography.

3. TERMINOLOGY

(l, m) mth node at level l in a tree, where $0 \le m \le 2l - 1$

(l, m) ^{I}updated mth node at level l in a tree, where $0 \le m \le 2l - 1$

M (l, m) Member who occupies the node (l,m)

PK (l,m) Public Key associated with the node (l,m)

SK (l,m) Private key to be associated with the node (l,m)

PK^{I} New public Key associated with the node (l, m)

SK^{I} New private Key associated with the node (l, m)

T(l,m) Sub tree rooted at the node (l,m)

E(PK, X) Encryption of data X using a public key PK

E(SK, X) Encryption of data X using a private key SK

GK Group key

RSGC Right subgroup controller

LSGC Left subgroup controller

n Number of members in a group

p,g EIGmal group parameters

SGKP-1 technique one

SGKP-2 technique two

4.0 SECURE GROUP KEY DISTRIBUTION USING SUB GROUP CONTROLLERS AND HYBRID KEY TREES

This section explains the first technique named SGKP-1.It assumes a binary tree. Hierarchy of keys will be maintained like LKH [4]. SGCP-1 maintains two types of nodes namely dummy nodes—nodes other than live nodes and live nodes —member nodes and the siblings of the root node. Members occupy at leaf nodes. Each tree will have two sub group controllers (SGCs).The left SGC (LSGC) represents the left sub tree and right SGC (RSGC) represents the right sub tree. Each SGC is associated with two key pairs (two private keys and two corresponding public keys) one to represent its left subtree and the other for its right subtree and one group key. LSGC is associated with a left private key1 (LSK1) and left public key1 (LPK1) to represent the left sub tree of this SGC. Another key pair (left private key2 (LSK2) and left public key2 (LPK2)) represents the right subtree of this SGC. Similarly RSGC is associated with two key pairs - right private key1 (RSK1) and right public key1 (RPK1)) which represents the left sub tree of this SGC. The right private key2 (RSK2) and right public key2 (RPK2)) represents the right sub tree of this SGC. Each SGC knows one of the public key of the other SGC. Each member who occupies at leaf node is associated with the public key of the corresponding SGC and a group key along with its own

asymmetric key pair. Each live node (SGCs and members) knows complete tree structure. The intermediate nodes (other than live nodes) don't have any keys. Only the live nodes participate in key generation and key distribution. Dynamically selected members sponsor and co-sponsor which are from the same subgroup where the event is likely to happen will participate in the generation of keys using Diffie-Hellman key exchange algorithm. The selection algorithm for sponsor and co-sponsor is different for join and leave event. The right shallowest node (from the root) will be the insertion point. The joining point sponsor and co-sponsor will be selected independently by the existing group members.

When a SGC receives a message from other SGC (encrypted with one of its public key) it will be broadcasted to the entire subgroup to which the received SGC is responsible for. When a SGC receives (not from the other SGC) a message encrypted with one of its public key from its own subgroup it will broadcast that message to the other subtree of its own. For example, if a message comes from left subtree of the SGC then it will be broadcasted to the right subtree encrypted with the corresponding private key. Sponsor and co-sponsor are not fixed and depend on the event (join/leave) and the time of the event i.e tree structure at the time of the event. Sponsor and co-sponsor will be from the same sub group in which the event (leave/join) is taking place. All the nodes updates their keys as follows:

New Group key = new *GK*

New private key=old Private Key + new *GK* mod p

New public key= old Public key X g^{GK} mod p

4. 1 Join Protocol

When a member wants to join it sends the join request along with its public key. The joining point, sponsor and cosponsor will be identified by the existing members independently. The rightmost shallowest leaf node of the tree will be selected as the sponsor. If the sibling node of the sponsor is the leaf node then it becomes co-sponsor. Otherwise the sponsor for subtree rooted at the sibling node becomes the co-sponsor. Two new nodes will be created by the existing members and will be added as children to the joining point. The sponsor currently associated with the joining point will be given to the right child (from the root)of the joining point and the new member with its public key will be assigned to the left child of the joining point. The secret *GK* will be agreed between sponsor and cosponsor using Diffie Hellman key exchange protocol. Sponsor broadcasts *GK* encrypted with old group key .All the group members except the new member update their group key and also the public key of their corresponding SGC. Sponsor also sends to the newly joined member, the new *GK* encrypted with the joining members public key, updated public key of the corresponding SGC and tree structure. Now the entire group knows the ***GK*** (group key).

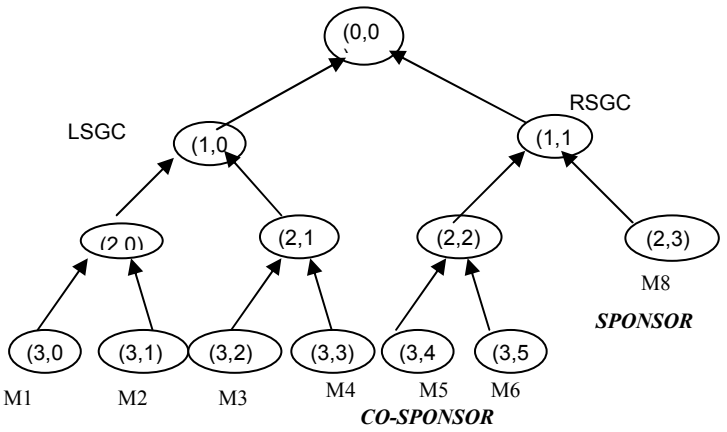


Figure 1: Tree before join

Suppose a new member M7 wants to join the group it broadcasts the join request along with its public key. The joining point in this case will be M8. In the above figure the joining point and the sponsor will be M8. The co-sponsor is M5. Two new nodes M8 and M7 will be created and the sponsor will be assigned to M8 and the new member will be M7. The secret *GK* will be agreed between M8 (sponsor) and M5 (co-sponsor*)* using Diffie Hellman key exchange protocol. M8 broadcasts the *GK* encrypted with old group key. All the group members update their group key and also the public key of their corresponding SGC(1,1).*M8* also sends to the joining member new *GK* encrypted with the joining members public key, updated Public key of the corresponding SGC(1,1) and tree structure.

SPONSOR→G: E (old GK, new GK)

SPONSOR (M8) →M7: E ($(PK_{(3,7)})$,$((1,1)^1$,GK)))

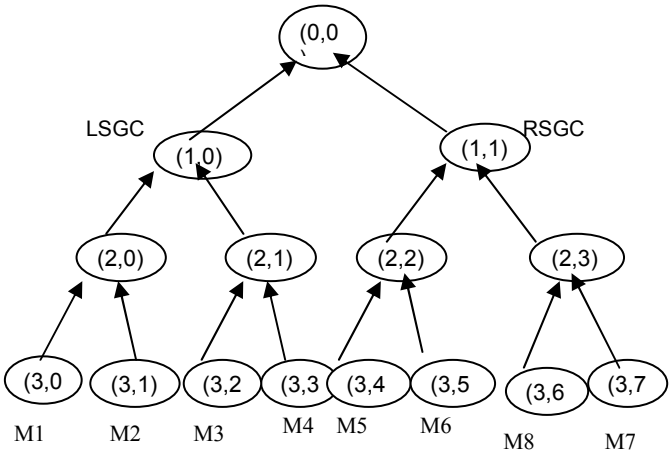


Figure 2: Tree after join

4.2 Leave Protocol

When an existing member wishes to leave the group, it broadcasts a leave request. In case of a leave, the algorithm for the selection of sponsor and co-sponsor will be different from that of join. The existing members will dynamically selects the sponsor and co-

sponsor. The selection criteria will depends on the tree structure at the time of the event. The three different possibilities are:1) If the sibling of the leaving member is a leaf node then it is the Sponsor at this moment and Co-Sponsor is the Sponsor of the subtree rooted at the sibling node of leaving node's parent. 2) If the sibling of the leaving member is a dummy node with two live members then these two will be selected as Sponsor and Co-Sponsor. 3) If the sibling node of the leaving member is not a leaf node and it has only one child, this child will be the Sponsor and The Co-Sponsor is the Sponsor of the subtree rooted at the sibling node of leaving node's parent. Sponsor and Co-Sponsor exchange new GK using DH algorithm. Sponsor broadcasts this new GK encrypted with the old public key of its SGC. For example, sponsor M8 broadcasts the GK encrypted with the old public key of RSGC (1, 1). RSGC updates its public and private key pairs and broadcasts the GK encrypted with the old public key of LSGC (1, 0).The LSGC receives this and encrypts it with the corresponding private key and distributes the GK encrypted with the old public keys (2-one for each sub group).All members i.e. M1, M2, M3, M4 get GK (group key) and update the public keys of their corresponding SGC. RSGC also broadcast the GK to its left subtree so that the members M5, M6 also updates the public keys they had (of RSGC (1, 0)) and also the new group key(GK).

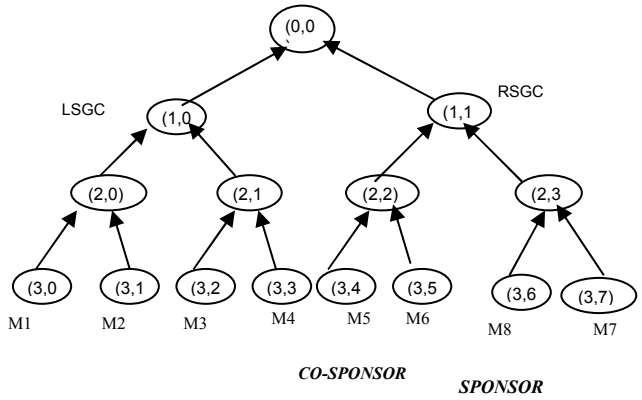


Figure 3: Tree before Leave

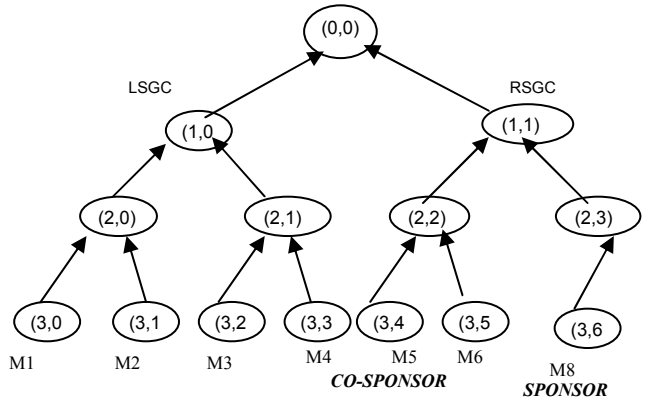


Figure 4: Tree after Leave

SPONSOR (M8) →RSGC: E (old PK of RSGC, new GK)

RSGC→T (2, 2): E (old SK of RSGC, new GK)

RSGC→T (1, 0): E (old SK of LSGC, new GK))

RSGC→LSGC: E (old PK of LSGC, new GK)

5.0 REKEYING LIMITED TO SUBGROUP USING HYBRID KEY TREES

This section explains the first technique named SGKP-2.It assumes a Binary tree. Hierarchy of keys will be maintained like LKH [4]. SGCP-2 maintains two types of nodes namely dummy nodes—nodes other than live nodes and Live nodes —member nodes and the siblings of the root node. Members occupy at leaf nodes. Each tree will have two sub group controllers (SGCs).The left SGC (LSGC) represents the left sub tree and right SGC (RSGC) represents the right sub tree. Each SGC is associated with two key pairs (two private keys and two corresponding public keys) one to represent its left subtree and the other for its right subtree and one group key. LSGC is associated with a left private key1 (LSK1) and left public key1 (LPK1) to represent the left sub tree of this SGC. Another key pair (left private key2 (LSK2) and left public key2 (LPK2)) represents the right subtree of this SGC. Similarly RSGC is associated with two key pairs - right private key1 (RSK1) and right public key1 (RPK1)) which represents the left sub tree of this SGC. The right private key2 (RSK2) and right public key2 (RPK2)) represents the right sub tree of this SGC.Two group keys will be there one for each subtree. Each SGC knows one of the public key of the other SGC and the group key of its own. Each member who occupies at leaf node knows the public key of the corresponding SGC and a group key of its own subtree along with its public and private key pair. Each live node (SGCs and members) knows complete tree structure. The intermediate nodes (other than live nodes) don't have any keys. Only the live nodes participate in key generation and key distribution.

Dynamically selected members Sponsor and Co-Sponsor**,** which are from the same subgroup where the event is likely to happen by the existing group members will participate in the generation of keys using Diffi-Hellman key exchange algorithm. The selection algorithm for sponsor and co-sponsor are different for join and leave event. The right shallowest node (from the root) will be the insertion point. Sponsor is the right most (from the root) shallowest leaf node

5.1 Join Protocol

When a member wants to join it sends the join request along with its public key. The joining point, sponsor and cosponsor will be selected by the existing members independently. The rightmost shallowest leaf node of the tree will be selected as the sponsor. If the sibling node of the sponsor is the leaf node then it becomes co-sponsor. Otherwise the sponsor for subtree rooted at the sibling node becomes the co-sponsor. Two new nodes will be created by the existing members and will be added as children of the joining point. The sponsor currently associated with the joining point will be given to the right child of the joining point and the new member with its public key will be assigned to the left child of the insertion point. The secret *GK* will be agreed between sponsor and cosponsor using Diffie-Hellman key exchange protocol. Sponsor broadcasts *GK* encrypted with old group key (eg. old *GK*) of that subgroup. All the group members of that subgroup update their group key and also the public key of their corresponding SGC. Since the newly joined member does not know the old group key of that subgroup where it joins it can't get the new *GK*. So, the Sponsor sends to the joining member, the new *GK* encrypted with the joining member's public key, updated

public key of the corresponding SGC and tree structure. Finally the SGC which is responsible for the subgroup where the event has happened sends the updated public key of its own to the other SGC. Since the old *GK* which was used earlier is not known to the newly joined member the backward secrecy can be maintained. Now when a member wants to send a message to the entire group it will be encrypted with the group key that the sender knows. So all the group members who belong to the sender subgroup will receive the message. Now the SGC whose is responsible for the senders subgroup will encrypt the message with the public key of the other SGC .Finally the other SGC will broadcasts the message to its own subgroup using its own group key.

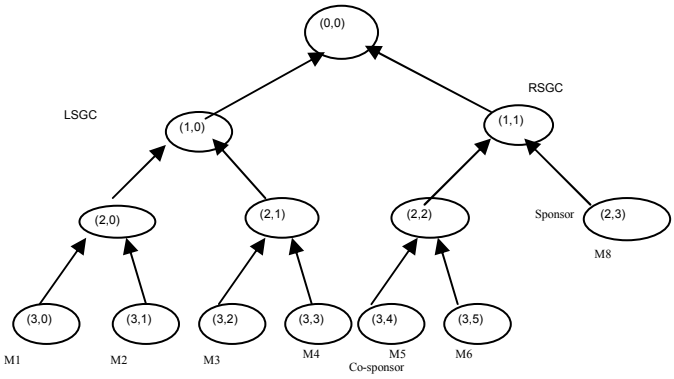


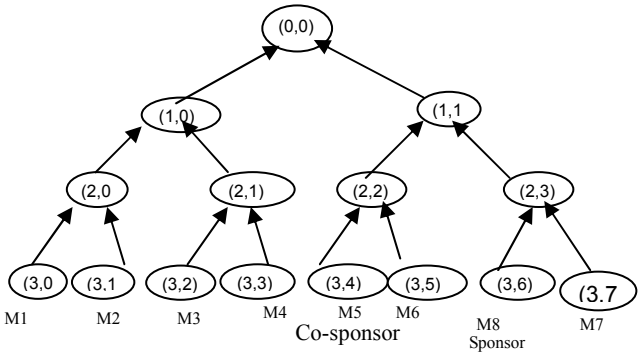
Figure 5: Tree before join

Suppose a new member M7 wants to join the group it broadcasts the join request with its public key. The insertion point and sponsor in this case will be M8. In the above figure two new nodes M8 and M7 will be created and the sponsor will be assigned to M8 and the new member will be M7.The co-sponsor is M5. The secret GK will be agreed between M8 (sponsor) and M5 (Co-Sponsor) using Diffie Hellman key exchange protocol. M8 broadcasts the GK encrypted with old group key that it had. All the group members of that subgroup update their group key and also the public key of their corresponding SGC(1,1).M8 also sends to the joining member new GK encrypted with the joining members public key, updated Public key of the corresponding SGC(1,1) and tree structure. Finally, RSGC sends it new public key to LSGC.

Sponsor→RSGC: E (old GK, new GK)

Sponsor(M8)→M7: E($((1,1)^1$,GK), $PK_{(3,7)}$)

RSGC →LSGC :$(1,1)^I$



5.2 Leave Protocol

It is initiated when the group members receive a leave event. The sponsor and co-sponsor are not fixed. The selection criteria will depends on the tree structure at that moment. The three different possibilities are:1) If the sibling of the leaving member is a leaf node then it is the sponsor at this moment and co-sponsor is the sponsor of the subtree rooted at the sibling node of leaving node's parent. 2) Otherwise if the sibling of the leaving member is a dummy node with two live members then these two will be selected as sponsor and co-sponsor. 3)Otherwise if the sibling node of the leaving member is not a leaf node and it has only one child, this child will be the sponsor and the co-sponsor is the sponsor of the subtree rooted at the sibling node of leaving node's parent. Sponsor and Co-Sponsor exchange new *GK* using Diffie-Hellman algorithm. Sponsor broadcasts this new *GK* encrypted with the old public key of the SGC to which the leaving member belongs to. For example, sponsor M8 broadcasts the *GK* encrypted with the old public key of RSGC (1,1). RSGC updates its public and private key pairs and broadcasts the *GK* encrypted with the old private key which the other sub tree(T(2,2))knows. So the members M5, M6 also updates the public keys they had (of RSGC (1,1)) and also the new group key (*GK*).Finally RSGC sends its new public key to the LSGC.

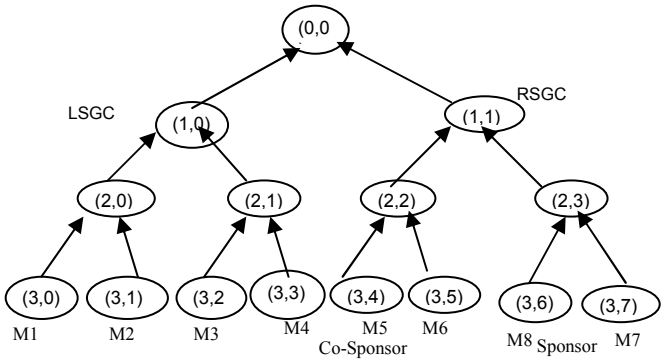


Figure 7: Tree before Leave

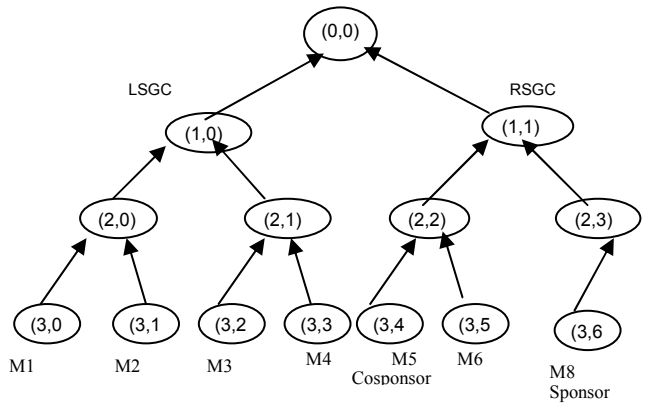


Figure 8: Tree after Leave

SPONSOR (M8) →RSGC: E (old PK of RSGC, new GK)

RSGC→T (2, 2): E (old SK of RSGC, new GK)

RSGC →LSGC :(1, 1)^{I}

6. ANALYSIS

6.1 Number of keys maintained at each member

S.No	Protocol	Keys
1	SGKP-1	4
2	SGKP-2	4
3	DLPKH	3l+2

6.2 Total number of unique keys

S.No	Protocol	Keys
1	SGKP-1	2n+9
2	SGKP-2	2n+10
3	DLPKH	4n-2

6.3 Total no. of keys at a particular time

S.No	Protocol	Keys
1	SGKP-1	4n+12
2	SGKP-2	4n+12
3	DLPKH	n(3l-1)

6.4 Total no. of rounds

S.No	Protocol	Join	Leave
1	SGKP-1	3	3
2	SGKP-2	3	3
3	DLPKH	3	2

6.5 Total no. of messages sent

S.No	Protocol	Join	Leave
1	SGKP-1	4	7
2	SGKP-2	5	6
3	DLPKH	2l+3	2l-2

6.6 No. of encryptions and decryptions by sponsor & member

A) Join

SGKP-1

Entity	DH	No. of Encryptions	No. of Decryption
Sponsor	1	1(symmetric)	----
Co-sponsor	1	----	--
RSGC	---	---	--
LSGC	---	---	---
Newmember	--	---	1(Asymmetric)
Nonmember	--	---	1(symmetric)

SGKP-2

Entity	DH	No. of Encryptions	No. of Decryptions
Sponsor	1	1(Asymmetric)	----
Co-sponsor	1	----	--
**RSGC	--	2(Asymmetric)	1(Asymmetric)
LSGC	--	--	1
New Member	--	--	1(Asymmetric)
Nonmember	---	----	1(symmetric)

**We assume that the event is happened in the right sub tree.

DLPKH

Entity	DH	No. of Encryptions	No. of Decryption
Sponsor	1	1(Asymmetric)	----
Co-sponsor	1	----	--
New member	--	---	1(Asymmetric)
Nonmember	--	---	1(Asymmetric)

B) Leave

SGKP-1

Entity	DH	No. of Encryptions	No. of Decryptions
GL-1	1	1(Asymmetric)	----
GL-2	1	----	--

**RSGC	--	2(Asymmetric)	1(Asymmetric)
LSGC	--	1	1
Leaving member	--	--	--

**We assume that the event is happened in the right sub tree.

SGKP-2

Entity	DH	No. of Encryptions	No. of Decryptions
GL-1	1	1(Asymmetric)	----
GL-2	1	----	--
**RSGC	--	1(Asymmetric)	1(Asymmetric)
LSGC	--	--	1
Leaving member	--	--	--
Nonmember	---	----	1(symmetric)

**We assume that the event is happened in the right sub tree.

DLPKH

Entity	DH	No. of Encryptions	No. of Decryptions
Sponsor	1	l-1(Asymmetric)	----
Co-sponsor	1	----	--
non member	--	---	1(Asymmetric)
Leaving member	--	---	--

6.7 Key updates required by each member

SGKP-1	SGKP-2	DLPKH
3	3	3l-2

6.8 Key updates required by Sponsor

SGKP-1	SGKP-2	DLPKH
1	1	3l-3

7. CONCLUSION

With frequent joins and evictions, the rekeying mechanism is the critical issue in any group key management protocols. Several techniques have been proposed earlier for the efficient rekeying. Some of the earlier techniques like DLKH [10] suffer from lack of backward secrecy in join operation, techniques like LKH [4] and FDLKH [7] requires the establishment of the secure channels for the distribution of the group key. DLPKH [9] eliminates some of the problems of the above schemes but it also suffers from the requirement of huge keys for each member as well as high computational resources because of the use of public key cryptography for all the operations and it reveals the private keys to others. When compared to the above discussed schemes our scheme has certain advantages like the overhead of secure channel establishment for the secure group key distribution and is eliminated with the use of hybrid key trees and also the storage requirements for each member is less and moreover less computational resources will be required when compared to DLPKH scheme. The major advantage of the scheme proposed in this paper is that only a subset of the group members will be disturbed in the re-keying process which was not the case with any of the above discussed schemes. The number of keys maintained at each member in this scheme, number of messages sent, size of the messages and number of encryption and decryptions are always constant unlike the other schemes which typically depends on the height of the tree.

REFERENCES

[1] H. Harney and C. Muckenhim, Group Key Management Protocol (GKMP) Architecture," RCF 2094, July 1997.

[2] H. Harney and C. Muckenhim, Group Key Management Protocol (GKMP) Specification, RFC 2093, July 1997.

[3] G. H. Chiou and W. T. Chen. Secure Broadcast using Secure Lock. IEEE Transactions on Software Engineering, 15(8):929–934, August 1989.

[4] C. K. Wong, M. Gouda, and S. S. Lam. Secure Group Communications Using Key Graphs. IEEE/ACM Transactions on Networking, 8(1):16–30, February 2000.

[5] D.A. McGrew and A.T. Sherman. Key Establishment in Large Dynamic Groups using One-way Function Trees.

[6] B. DeCleene, L. Dondeti, S. Griffin, T. Hardjono, D. Kiwior,J. Kurose, D. Towsley, S. Vasudevan, and C. Zhang. Secure group communications for wireless networks. MILCOM, June 2001.

[7] FDLKH: fully decentralized key management scheme on a logical key hierarchy. Springer Berlin / Heidelberg, Volume 3089/2004

[8] S. Rafaeli and D. Hutchison. Hydra: a decentralized group key management. 11th IEEE International WETICE: Enterprise Security Workshop, June 2002.

[9]DLPKH: Distributed Logical Public Key Hierarchy Rakesh Bobba, Himamshu Khurana.

[10] O. Rodeh, K. Birman, and D. Dolev. Optimized group rekey for group communication systems. Network and Distributed System Security, February 2000.