A Secure and Scalable Framework for Group Communication

Annappa, G. Prameela Rani
Department of Computer Engineering
National Institute of Technology, Karnataka, Surathkal
annappa@gmail.com, prameela.j04cs@gmail.com

Abstract - C*ritical networking issues in group communication involve security, scalability and dynamic membership changes. Security provides confidentiality, authenticity, and integrity of messages exchanged between group members. Tree Group Diffie Hellman is an efficient key agreement protocol for peer groups. But, it is not scalable beyond 100 members. On the other hand, large multicast groups don't support dynamic membership changes. In this Paper, we propose a new framework, which addresses the problems of scalability and dynamic membership change.*

I INTRODUCTION

Collaborative applications like voice and video conferencing, white-boards and distributed simulations require services such as reliability, ordered message delivery, fault-tolerance. Group communication systems provide these services. Security is crucial for these applications because they operate in dynamic network environment and communicate over insecure open networks such as the Internet.

Group oriented applications require multicast [12] to minimize the volume of the network traffic they generate. Security threats [8] on the Internet are holding back the widespread adoption of multicast. Hence, IP multicast by itself cannot provide any mechanism to limit the access to the data being transmitted. Also, compared to unicast, multicast is more susceptible to attacks [8] and it is difficult to extend other features to it in a scalable manner.

Some group applications require a large number of members to participate in the group events. The framework should be capable of supporting large number of members and dynamic membership changes (frequent joins and leaves).

II RELATED WORK

A Frameworks

In large multicast groups [5] which are centralized and hierarchical type the communication is mostly one to many. These are scalable but are not suitable for groups with dynamic membership changes. Two notable protocols in this category are Iolus [6] and Nortel [5].

Dynamic Peer Groups [1] (DPG) typically assume many-to-many communication patterns. These are relatively small in size. The best protocols for these groups are the contributory key management protocols. Some of the contributory key management protocols are BD (Bermester-Desmedit), GDH (Group Diffie Hellman), TGDH (Tree Group Diffie Hellman) and STR (Skinny Tree).

B Key Management techniques

Group key management techniques [1, 15] proposed in the past generally fall into three categories: 1) Centralized, 2) Distributed, and 3) Contributory. Centralized technique is conceptually simple as it involves a single entity that generates and distributes keys to group members via a pair-wise secure channel established with each group member. This technique is scalable.

Distributed group key management is more suitable over unreliable networks. It involves dynamically selecting a group member that acts as a key distribution server. But it requires a key server to maintain long-term pair-wise secure channels with all current group members for distributing the group keys.

In contrast, contributory technique [2] requires every group member to contribute an equal share to the common group secret, computed as a function of all members' contributions. It avoids the problems with the single point(s) of trust and failure and do not require establishing pair-wise secret channels among group members. Also, it offers strong key management security properties. These protocols are not scalable.

III SECURITY ISSUES

One of the most important security requirements of a group communication is the key freshness. A key is fresh if it can be guaranteed to be new i.e. the old key should never be reused through some sequence of operations by either an adversary or authorized party. Only the involved parties should know the session key. Assume that a group key is changed m times and the sequence of successive group keys is $K^l = \{ K_0, K_1,.. K_m\}$. Four important security properties are defined as follows

1. Group Key Secrecy guarantees that it is computationally infeasible for a passive adversary to discover any group key $K_i \in K^1$ for all i.
2. Forward Secrecy guarantees that a passive adversary who knows a contiguous subset of old group keys (say $\{ K_i, K_{i+1},\ldots, K_j \}$) cannot discover any subsequent group key K_l for all i, j, l where $0 \le i < j < l$.
3. *Backward Secrecy* guarantees that a passive adversary who knows a contiguous subset group keys (say $\{K_i, K_{i+1}, \ldots , K_j \}$) cannot discover preceding group key K_l for all l, j, k where $l < i < j$.
4. *Key Independence* guarantees that a passive adversary who knows a proper subset of group keys $K_m \subset K^l$ cannot discover any other group key $K \in (K^l - K_m)$.

Evaluation Criteria

The parameters that are used in the evaluation of the frameworks are number of keys with the Mediator (if exists) and with each member, number of messages to update keys on a join and leave, processing time for key management, space needed for the key tree and the time required for tree updation for join and leave etc.

IV FRAMEWORK DESIGN

A Outer Framework

The new framework is a two level hierarchical framework that make use of secure distribution trees [6]. The secure distribution tree is composed of many smaller secure subgroups arranged in a two level hierarchy to create a single virtual secure group. Each subgroup is relatively independent and thus localizes the effect of group membership changes to one subgroup. Hence the features scalability and dynamic membership operations are supported.

Each subgroup in the secure distribution tree has its own group address (class-D address) and its own subgroup keying material. There is no global group key. Fig.1. shows the outer framework architecture. The highest level in the framework is the Group Security Mediator (GSM) and the lower levels are the subgroups. The Mediator maintains the group information of each subgroup. Each subgroup has sponsors for all membership operations that act as the inter mediators between GSM and its subgroup members.

B Inner Framework

Inner framework is the framework at the subgroup level. The functionality of each subgroup is similar to the TGDH [10] protocol. Every member calculates the group key from the group tree. The procedure for calculating the group key is based on Diffie Hellman key exchange [4] mechanism. Because sponsors are not fixed, it avoids single point of failure and provides decentralized property at the lower layers.

The sponsor for the join of $(n+1)^{th}$ member is the member n. For the k^{th} member leave, $((k-1)\%n)$ becomes sponsor, where n is the total number of members in the subgroup.

V OPERATIONAL SEMANTICS

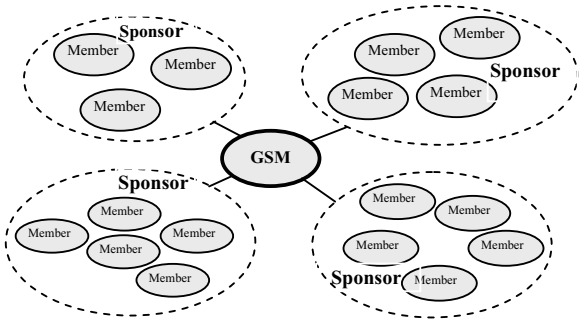


Fig. 1: Architecture of the Outer framework

The basic operations in group communication are membership operations (join and leave) and group operations (group initialization, shutdown and data transfer).

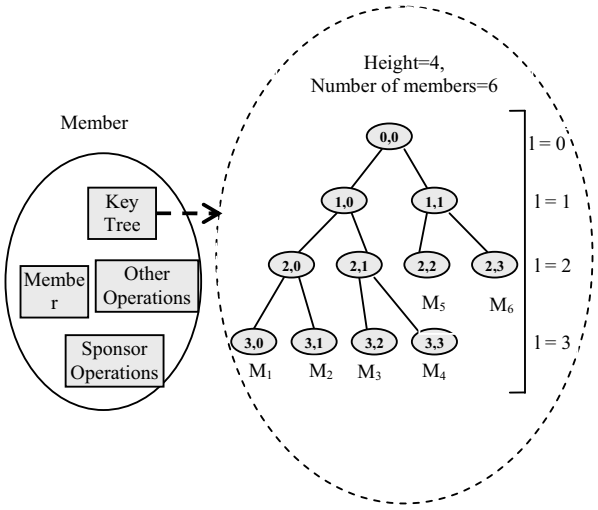


Fig 2: Architecture of the inner framework

A Group Initialization

Group Initialization of the secure multicast group requires only that the GSM for the group be started.

B Member Join

The member who is joining the group will generate a random number and calculates the blind random. The blind random along with his information is sent to the Mediator as join request. Join protocol is given in table 1.

C Member Leave

A member leaves the group either by self leave or by forcible leave from the mediator. Incase of self-leave, the member will send the leave request with his number (in the subgroup) to all the members in its subgroup. If the member leaving is the only person in the subgroup, then he sends his leave request to the mediator. In forcible leave case, the mediator will send the leave message containing the member number to the corresponding subgroup. Leave Protocol is given in table 2.

D Data Transfer

The member sending the data will become the sponsor itself. He multicasts the data encrypted with K_{SGRP} directly to its local subgroup and unicast to mediator. Every member in that local subgroup decrypts the message and gets the data. The Mediator then, decrypts it and re-multicasts the data with the other subgroup keys. The main disadvantage with this method is, Mediator decrypt and re-encrypt data would require an enormous amount of encryption bandwidth. Therefore, it is suitable when the size of data is small. For large data, the sender generates a random key on a per-transmission basis and uses this key to encrypt the data. The random key is encrypted with the local subgroup key. Now the random

key and the message are sent to Mediator. Mediator then, decrypts the random key and re-encrypts with other subgroup keys and multicast the message to the corresponding groups.

E Shutdown

When all the members leave the group, the Mediator shuts down by itself.

VI KEY MANAGEMENT

Every member in the subgroup maintains a key tree [7] with the blinded keys of all members. The key tree is a complete binary tree. The root is located at level 0 and the member's keys are at the leaf level. The nodes are denoted as (l, v), where *l* is he level and *v* is the node number in that level. The structure of the key tree is shown in Fig 2. Each node is associated with the key $K_{(l,v)}$ and the blinded key (bkey) $BK_{(l,v)} = f(K_{(l,v)})$ where $f(k) = \alpha^k \bmod p$, where α and p are large primes. The key $K_{<l,v>}$ is computed recursively as follows:

$$K_{<l,v>} = \alpha^{K_{<l+1,2v>}K_{<l+1,2v+1>}} \bmod p$$
$$= f(K_{<l+1,2v>}K_{<l+1,2v+1>})$$

Key tree updations for various membership operations are as follows.

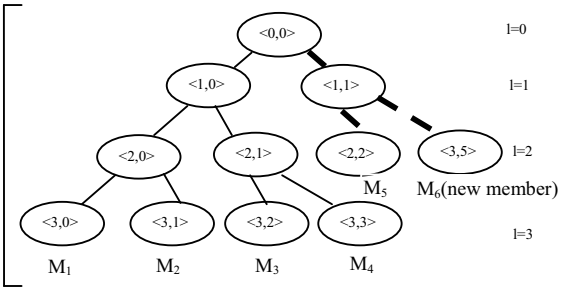


Fig 3(a): Tree when a member joins

Table 1: Join protocol

Step 1: The new member sends the join request to Mediator sending his blind key.

$$BK = \alpha^{r_{n+1}}$$

M_{n+1} ---------- > Mediator

Step 2: Mediator decides whether to accept or deny the request. If the request is accepted, he searches for the free group (group which has members less than the maximum number of members per group). Once found, he forwards (multicast) the request to the selected group. Now, the join sponsor of that subgroup (last member) takes the control. If no subgroup is free, a new subgroup will be formed, and the new member himself become join sponsor and generate the group key on his own.

C ---------- > $\{M_1, M_2, M_3, \ldots, M_n\}$

Step 3: Every member(including sponsor)

- Updates key tree by adding new member node, and new intermediate node.
- Sponsor M_n additionally, updates his share, encrypt this with the old group key and multicast it to the members.

Step 4: Every member now, updates the sponsor share in the tree and modifies the tree.

Step 5: Sponsor sends the latest key tree to the new member in a secure way and also sends the key information to the Mediator.

Step 4: The new member becomes the new join-sponsor. Mediator updates this information

Table 2: Leave Protocol

Step 1: The member who wants to leave the group, multicasts the leave request to all the members incase of his self leave, or Mediator sends the leave message incase of forcible leave of the member.

Step2: Every member(including sponsor)

- Updates key tree by removing the leaving member node and relevant parent node,
- Updates all keys and bkeys from the leaf to the root node,

Leave-sponsor additionally(the member immediate left to the leaving member)

- Updates his share and send it to all the members

Step 3: Every member updates and computes the group key.

Step 4: Sponsor sends the key information to the Mediator.

A Join Rekey

Once the member receives the join request, he first chooses the insertion point, which is the last leaf node position in the complete binary tree. Now, a new member node is created and is joined to the tree using a new temporary node and, all the keys in that key path are updated. Fig 3(a) shows the key tree when sixth member joins.

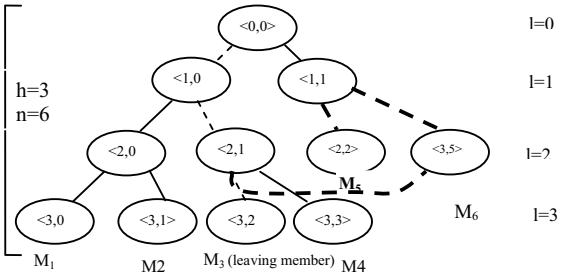


Fig 3(b): Tree when m3 leaves

B Leave Rekey

When a member receives the leave request, he identifies the corresponding member node in the tree. Once identified, that node will be replaced with the last member node, and all the keys in those key paths are updated accordingly. Fig 3(b) shows the key tree when a member leave.

C Update Key

This occurs when a sponsor sends his update share message to the group members. The sponsor node is identified, updated with the new value and all the keys in that key path are updated accordingly.

D Periodic Refresh

In addition to the membership rekey operations; group key is updated periodically for more security purposes. In this case one of the sponsors updates his share and sends the update share message to the group members.

VII RESULTS AND ANALYSIS

A Experimental Setup

Framework is implemented as a simple text conference. It is tested in LAN with 30 systems. For analyzing the tree updation time and encryption time the key management data structure is used separately.

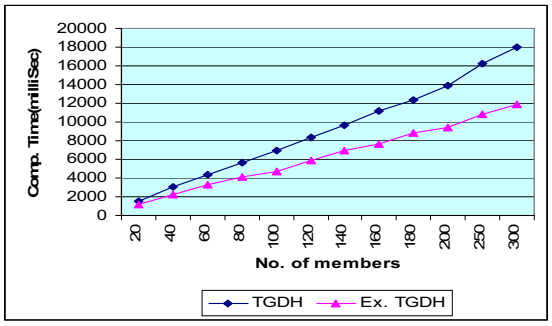
B Analysis

1. Key Computation (Tree updation) Times

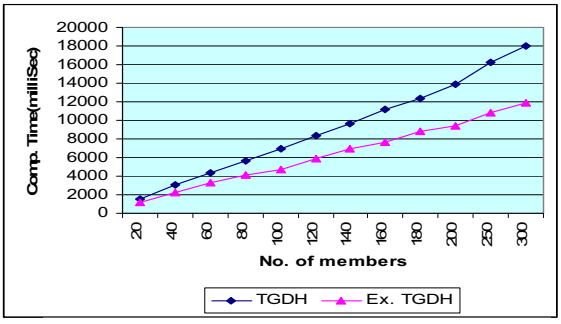
Computation times for join and leave operations are the times to modify the key tree when the corresponding membership operation occurs. Graph 1 shows the computation time (in milli seconds) for join re-key (for n sequential member joins) and Graph 2 for n sequential leaves. The maximum number of members per subgroup is taken as 100 members, as it is proven that TGDH performs better up to 100 members. From the graph, it is observed that the key computation times are significantly less for new framework (extended TGDH) compared to TGDH. This is because after 100 members join the subgroup, next member joins in the new subgroup.

2. Message encryption and decryption times

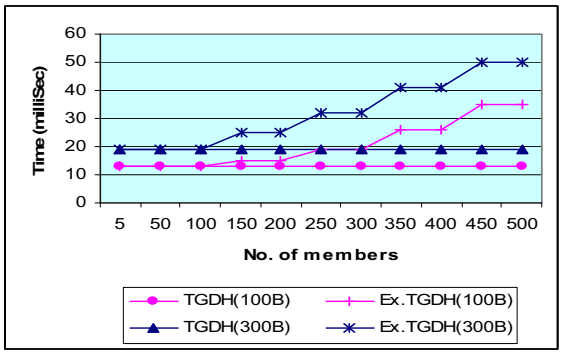
The limitation of the new framework is number of encryptions and decryptions. But these are not considerable compared to the key computation times. Graph 3 shows the encryption and decryption times in milliseconds for small data. Here the new framework takes more time than TGDH, because, TGDH requires only one encryption and decryption, where as the new framework needs k encryptions and 1 decryption, where k is the number of subgroups.



Graph 1: Computation time for join



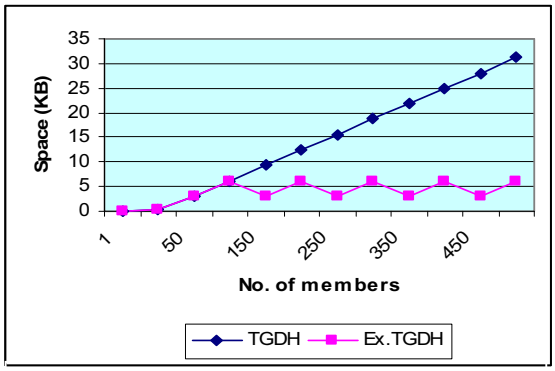
Graph 2: Computation time for leave



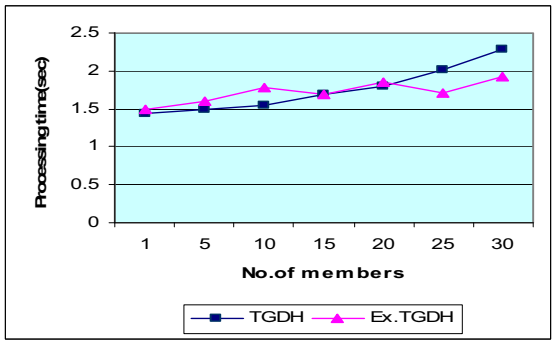
Graph 3: Encryption and Decryption times

3. Number of keys and Space Required

TGDH as well as the new framework requires each member to maintain the group tree. IDEA algorithm is used for message secrecy; the key size is 128-bit. So, total space required at each member is (2n-1)*2*128 bits. The new framework needs Mediator to maintain one key for each subgroup. Graph 4 shows the space requirements in KB.



Graph 4: Space Requirements



Graph 5: Processing Time for join

4. Processing Time for membership Operations

Total processing time includes key computation time, communication time, sponsor time and the Mediator time (for the new framework). It is the response time of the new member. For analyzing this, number of members per subgroup is taken as 10 for the new framework. As the new framework also involves Mediator time, initially, it

takes some extra time compared to TGDH. Graph 5 shows the processing times for join operation.

VIII CONCLUSION AND FUTURE WORK

A new framework which is secure and scalable is presented. This framework is an extension to the TGDH protocol. Scalability is achieved by two-level secure distribution trees. This framework is completely decentralized except at the top level. As it uses secure one way trees for key management, it provides more security. This framework has got good practical results in computation times and space requirements.

Even-though the new framework is scalable and secure, the only limitation is centralized Mediator. This framework can further be extended to make it complete decentralized by each subgroup somehow maintain the group keys of other subgroups. But, in doing so, the parameters should be preserved. Furthermore, this research area is still relatively young, and new techniques will surely evolve over time to replace or supplement those that currently exist.

IX REFERENCES

[1] Y. Amir, Y. Kim, C. Nita-Rotaru, and G. Tsudik, "On the performance of group key agreement protocols," Tech. Rep. CNDS 2001-5, Johns Hopkins University, Center of Networking and Distributed Systems, 2001.

[2] Amir Y, Kim Y, Nita-Rotaru C, Schultz J, Stanton J And Tsudik G. " Secure group communication using robust contributory key agreement". *IEEE Trans. Parallel and Distributed Systems. 15*, 5, 468–480, 2004.

[3] D. Balenson, D. McGrew, and A. Sherman, "Key Management for Large Dynamic Groups: One-Way Function Trees and Amortized Initialization," Internet Draft Report.

[4] Bruce Schnier, "Applied cryptography", second addition

[5] M. J. Moyer, J. R. Rao, and P. Rohatgi, "A survey of security issues in multicast communications," *IEEE Network*, vol. 13, no. 6, pp. 12–23, Nov.–Dec. 1999.

[6] S.Mitra, "Lolus: A Framework for Scalable Secure Multicasting "Proc. ACM SIGCOMM 1997

[7] M. J. Moyer, J. R. Roo, and P. Rohatgi, "Maintaining Balanced Key Trees for Secure Multicast." IETF internet draft June 1999.

[8] Tony Ballardie, Jon Crowcroft: "Multicast-Specific Security Threats and Counter-Measures". IEEE, ISBN: 0-8186-7027-4, 1995

[9] C.Wong, M. Gouda, and S. Lam, "Secure group communications using key graphs," *IEEE/ACM Trans. Networking*, vol. 8, no. 1, pp. 16–30, Feb. 2000.

[10] http://www.csm.ornl.gov/~dunigan/gkm.html

[11] http://www.cc.gatech.edu/~jlfan/mgkm.html

[12]http://www.cse.wustl.edu/~jain/cis788-97/ip_multicast