Proceedings of the International Conference on Man-Machine Systems (ICoMMS)
11 – 13 October 2009, Batu Ferringhi, Penang, MALAYSIA

Authenticated Collaborative Key Agreement for Dynamic Peer Groups

[1]Dr.M.Rajaram, [2]D.Thilagavathy
[1]Govt Colleg of Engineering, Tirunelveli.
[2]Adhiyamaan College of Engg, Hosur
rajaramgct@rediffmail.com

***Abstract* - Many emerging network applications are based upon a group communication model. In a peer-to-peer or ad-hoc network which do not have a previously agreed upon common secret key, communication is susceptible to eavesdropping, Hence a secure distributed group key agreement is required to establish and authenticate a common group key for secure and private communication. This paper presents an authenticated distributed collaborative key agreement for dynamic peer groups. The protocol is distributed in nature in which there is no centralized key server, collaborative in nature in which the group key is contributory, dynamic in nature in which existing members may leave the group while new members may join. Instead of performing individual rekeying an interval-based approach is used. The Queue-batch algorithm used for rekeying substantially reduces the computation and communication cost. Key authentication provided focuses on security improvement.**

***Keywords*—Authentication, dynamic peer groups, group key agreement, rekeying, secure group communication, security.**

I. INTRODUCTION

Many collaborative applications such as audio and video conferencing, white-boards, interactive chats and multiuser games, that rely on group communication requires the need for group-oriented security mechanisms. The security requirements of these applications are fairly typical, e.g., confidentiality, data integrity, authentication and access control. These are achieved through some form of group key management. A group key can be established either by *key Distribution* whereby a trusted third party chooses a conference key and securely distributes to all conferees, or by *key Agreement* in which a shared conference key is derived by all participants.

The *centralized group key distribution* has the disadvantages such as centralized key server must be a trusted one, conference conversations are disclosed to the centralized key server, single point failure may occur and cannot be used in situations like peer-to-peer or ad hoc networks where centralized resources are not readily available[5].

In this paper, a group key agreement protocol for a dynamic communication group in which members are located in a distributed fashion and can join and leave the group at any time is proposed.

To prevent a new user from reading past communications (backward confidentiality) and a departed user from reading future communications (forward confidentiality)[6], the *rekeying*, which means renewing the keys associated with the nodes of the key tree, is performed whenever there is any group membership change including any new member joining or any existing member leaving the group.

Rekeying performed immediately after every single join or leave event is known as individual rekeying. In batch rekeying join and leave requests during a rekeying interval is collected and the new key is computed[7].

Before the group membership is changed, a special member called the *sponsor* is elected to be responsible for updating the keys held by the new member (in the join case) or departed member (in the leave case). The rightmost member under the subtree rooted at the sibling of the join and leave nodes will take the sponsor role. The existence of a sponsor does not violate the decentralized requirement of the group key generation since the sponsor does not add extra contribution to the group key.

II. GROUP KEY ESTABLISHMENT

The tree-based group Diffie–Hellman (TGDH) protocol [5] is used to establish the group key in a dynamic peer group. Each member maintains a set of keys, which are arranged in a hierarchical *binary tree*. A node ID *v* is assigned to every tree node. For a given node *v* a *secret* (or private) key K_V and a *blinded* (or public) key BK_V are associated. All arithmetic operations are performed in a cyclic group of prime order *p* with the generator α. Therefore, the blinded key of node *v* can be generated by

$$BK_V = \alpha^{K_V} \bmod p \qquad (1)$$

Each leaf node in the tree corresponds to the individual secret and blinded keys of a group member. Every member holds all the secret keys along its *key path* starting from its associated leaf node up to the root node. Therefore, the secret key held by the root node is shared by all the members and is regarded as the *group key*.

The node ID of the root node is set to 0. Each *nonleaf* node *v* consists of two child nodes whose node ID's are given by 2v+1 and 2v+2. Based on the Diffie–Hellman protocol, the secret key of a nonleaf node can be generated by the secret key of one child node of *v* and the blinded key of another child node of *v*.

$$\begin{aligned} K_V &= (BK_{2v+1})^{K_{2v+2}} \bmod p \\ &= (BK_{2v+2})^{K_{2v+1}} \bmod p \\ &= \alpha^{K_{2v+1} K_{2v+2}} \bmod p \end{aligned} \qquad (2)$$

Unlike the keys at non leaf nodes, the secret key at a leaf node is selected by its corresponding group member through a secure pseudo random number generator. Since the blinded keys are publicly known, every member can compute the keys along its key path to the root node based on its individual secret key.

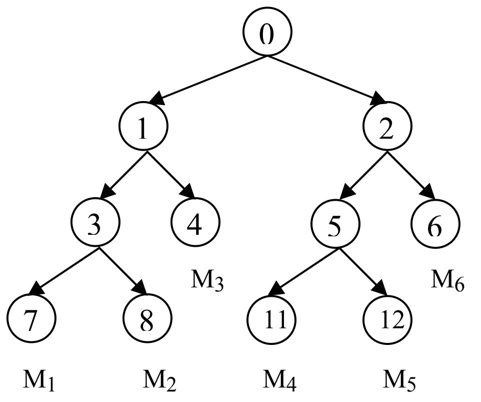

Fig.1. Key tree used in the tree-based group Diffie–Hellman protocol.

To illustrate, consider the key tree in Fig. 1. Every member M_i generates its own secret key and all the secret keys along the path to the root node. For example, member M_1 generates the secret key K_7 and it can request the blinded key BK_8 from M_2, BK_4 from M_3, and BK_2 from either M_4, M_5, or M_6. Given M_1's secret key K_7 and the blinded key BK_8, M_1 can generate the secret key K_3. Given the blinded key BK_4 and the newly generated secret key K_3, M_1 can generate the secret key K_1. Given the secret key K_1 and the blinded key BK_2, M_1 can generate the secret key K_0 at the root. Any communication in the group can be encrypted based on the secret key K_0 which is the group key

III. REKEYING

An interval-based rekeying is used in order to eliminate the difficulties of individual rekeying such as inefficiency and out-of-sync problem.

Interval-based rekeying maintains the rekeying frequency regardless of the dynamics of join and leave events, with a tradeoff of weakening both backward and forward confidentialities as a result of delaying the update of the group. Interval-based rekeying is done through the approaches *Rebuild algorithm*, the *Batch algorithm*, and the *Queue-batch algorithm*.

The interval-based algorithm is based on the following assumptions:

- All members are trusted in the key establishment process.
- The group communication satisfies *view synchrony* property [1].
- Rekeying operations of all members are synchronized to be carried out at the beginning of every rekeying interval.
- When a new member sends a join request, it also includes its individual blinded key.
- All members know the existing key tree structure and all the blinded keys within the tree.
- To obtain the blinded keys of the renewed nodes, the key paths of the sponsors should contain those renewed nodes. Since the interval-based rekeying operations involve nodes lying on more than one key paths, more than one sponsors may be elected. Also, a renewed node may be rekeyed by more than one sponsor. Therefore, it is assumed that the sponsors can coordinate with one another such that the blinded keys of all the renewed nodes are broadcast only once.

The motivation of the *Rebuild algorithm* is to minimize the resulting tree height so that the rekeying operations for each group member can be reduced. At the beginning of every rekeying interval, the whole key tree is reconstructed with all existing members that remain in the communication group, together with the newly joining members.

Given the numbers of joins and leaves within a rekeying interval, *Batch algorithm* attach new group members to different leaf positions of the key tree in order to keep the key tree as balanced as possible.

Queue-Batch Algorithm

Queue-batch is an effective rekeying algorithm which reduces the rekeying load by pre-processing the joining members during the idle rekeying interval, unlike other approaches which perform rekeying at the beginning of every rekeying interval, which results in high processing load during any update instance.

The Queue-batch algorithm is divided into two phases, namely the *Queue-subtree* phase and the *Queue-merge* phase. The first phase occurs whenever a new member joins the communication group during the rekeying interval. This new member is appended in a temporary key tree . The second phase occurs at the beginning of every rekeying interval and the temporary key tree (which contains all newly joining members) is merged to the existing key tree.

The Queue-batch algorithm is illustrated in Fig. 2, where members M_8, M_9, and M_{10} wish to join the communication group, while M_2 and M_7 wish to leave.

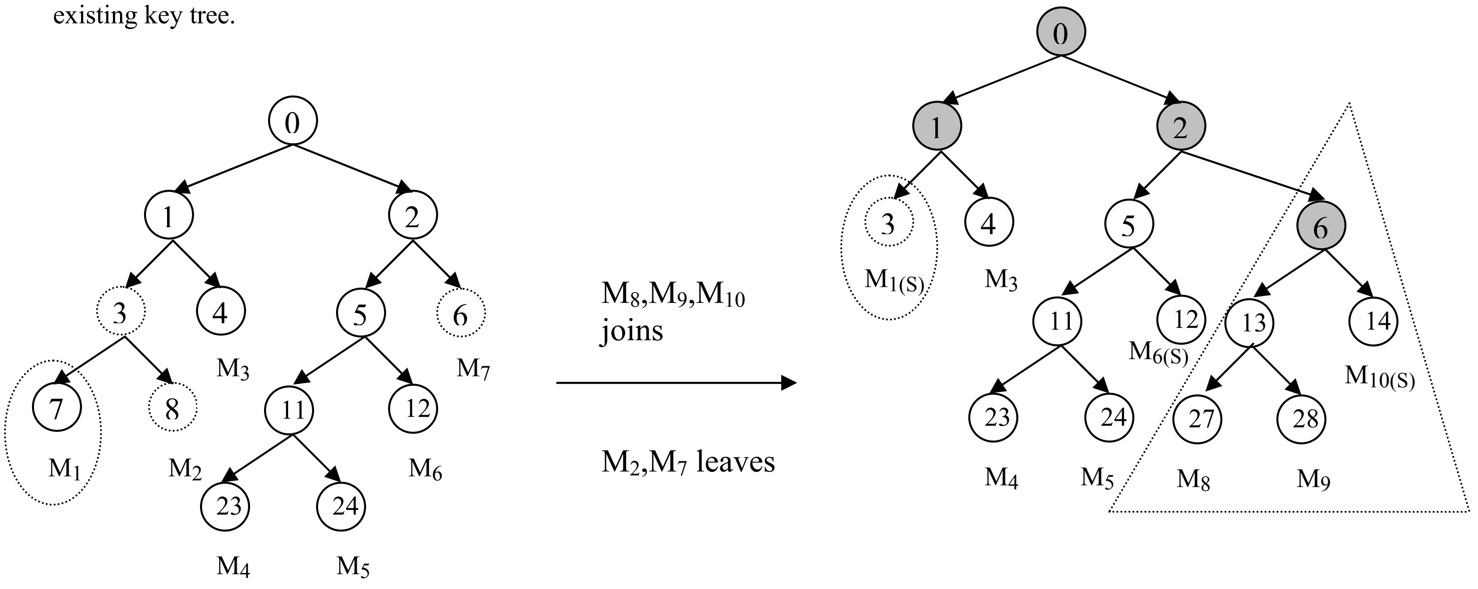

Fig.2. Example of Queue-Merge phase

Queue-subtree phase:

1) When a new member joins, a new temporary key tree is created if one does not exists previously.
2) Otherwise the insertion node in the existing temporary key tree is found and the new member is appended to the tree.
3) Elect the rightmost member under the subtree rooted at the sibling of the joining node to be the sponsor.
4) Every sponsor node rekeys renewed nodes and broadcasts new blinded keys.

Queue-merge phase:

1) If no members leave then add the temporary key tree to the shallowest node of the existing key tree such that the merge will not increase the resulting tree height, otherwise merge it to the root node.
2) When there are leaves add the temporary key tree to the highest leave position of the key tree and remove the remaining leaving leaf nodes and promote their siblings.
3) Elect the rightmost member under the subtree rooted at the sibling of the departed node to be the sponsor.
4) Every sponsor node rekeys renewed nodes and broadcasts new blinded keys.

Then the rekeying process is as follows:

1) In the *Queue-subtree* phase, the three new members M_8, M_9, and M_{10} first form the temporary key tree. M_{10}, in this case, is elected to be the sponsor.
2) In the *Queue-merge* phase, the temporary key tree is added at the highest departed position, which is at node 6. Also, the blinded key of the root node of temporary key tree, which is BK_6, is broadcast by M_{10}.
3) The sponsors M_1, M_6, and M_{10} are elected. M_1 renews the secret key K_1 and broadcasts the blinded key BK_1. M_6 renews the secret key K_2 and broadcasts the blinded key BK_2.
4) Every members compute the group key.

IV. KEY AUTHENTICATION

Authenticated Tree-Based Group Diffie–Hellman (A-TGDH) protocol is proposed that provides key authentication . The session-based group key is coupled with the certified permanent private components of the group members.

Each member holds two types of keys: *short-term secret* and *blinded* keys as well as *long-term private* and *public* keys. Short-term keys are randomly generated when a member joins the group and become expired when the member leaves, while long-term keys remain permanent across many sessions and are certified by a trusted CA.

The protocol satisfies several requirements that are crucial for key establishment [2]:

1) Perfect forward secrecy: The compromise of long-term keys does not degrade the secrecy of past short-term keys.
2) Known-key security : The compromise of past short-term keys does not degrade the secrecy of future short-term keys.
3) Key authentication : All group members are assured that no outsiders can identify the group key.

A -TGDH Protocol

Every node v in the key tree is associated with a secret key K_v and a blinded key BK_v. The *blinded key set* $BK`_v$, is constructed, which refers to a number of copies of BK_v's encrypted by the long-term private component of every descendant member of the sibling of node v. The set of the descendant members of node v is given by $\mathbf{M}_v$. The ith member, M_i, holds a short-term secret key γM_i and the corresponding blinded key $\alpha^{\gamma M_i}$ mod p as well as a long-term private key xM_i and the corresponding public key α^{xM_i} mod p, where all arithmetic operations are to be performed on the cyclic group of prime order p with generator α. Every member acquires the certificates of all other members and hence their long-term public keys from a trusted CA *prior to* the key agreement process[3].

Case 1(Node v is a nonleaf node)

Node v is a nonleaf node with child nodes $2v+1$ and $2v+2$ and $v \neq 0$ (since the blinded group key need not be broadcast)

$$\text{Then } K_v = \alpha^k \bmod p \quad (3)$$

$$\text{where } k = K_{2v+1} K_{2v+2} + K_{2v+1} \sum_{M_i \in \mathbf{M}_{2v+2}} xM_i + K_{2v+2} \sum_{M_i \in \mathbf{M}_{2v+1}} xM_i$$

$$BK`_v = \begin{cases} \{ \alpha^{kvxM_i} \bmod p : M_i \in \mathbf{M}_{v+1} \} \\ \quad \text{if node } v \text{ is the left child of its parent} \\ \{ \alpha^{kvxM_i} \bmod p : M_i \in \mathbf{M}_{v-1} \} \\ \quad \text{if node } v \text{ is the right child of its parent} \end{cases} \quad (4)$$

Case 2(Node v is a leaf node associated with member M_i)

$$K_v = \gamma M_i \quad (5)$$
$$BK`_v = \alpha^{\gamma M_i} \bmod p \quad (6)$$

If a node v needs to be renewed, a sponsor can broadcast $BK`_v$ and any member can include its short term blinded key in its join request.

V. KEY CONFIRMATION

Key confirmation is enforced to guarantee all group members that they are actually holding the same group key. Providing complete key confirmation requires every member to demonstrate its knowledge of the group key to all other members.

In the approach used for key confirmation, a sponsor is elected to broadcast the blinded group key so that every other member can verify if its computed blinded group key is identical to the one it receives. If a member finds that the keys are different, then it will report the error.

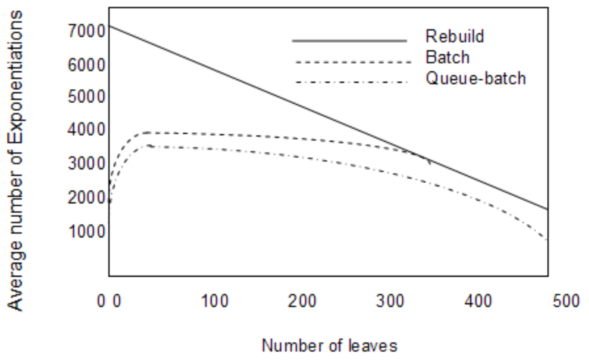

Fig.3. Average numbers of exponentiations at different numbers of leaves

VI RESULTS AND OBSERVATION

The three algorithms are examined based on two performance measures, namely:

1. *Number of exponentiation operations:* This metric gives a measure of the computation load of all members in the communication group.
2. *Number of renewed nodes:* A node is said to be *renewed* if it is a nonleaf node and its associated keys are renewed. This metric measures the communication cost since the new blinded keys of the renewed nodes have to be broadcast to the whole group.

The Figures 3 and 4 illustrate the average number of exponentiations and the average number of renewed nodes under different numbers leaving members. It is observed that Queue-batch outperforms the other two interval-based algorithms in all cases. There is a significant computation/communication reduction when the peer group is very dynamic (i.e., high number of members that wish to join or leave the communication group).

The results show that the interval-based algorithms outperform the individual rekeying approach in terms of both computation and communication costs and that Queue-batch performs the best among the three interval-based algorithms. Queue-batch performs much better when the join and leave events occur very frequently. It also demonstrates its robustness in keeping the key tree balanced and its capability in minimizing the number of rounds required to update the group key.

Queue-batch performs much better than the other two algorithms when the group is highly dynamic ie) in case of frequent joins and frequent leaves. When the number of join events is high, Queue-batch benefits from the pre-processing of join events in the Queue-subtree phase. In addition, when the number of leave events is high, Queue-batch reduces the heights of the existing tree nodes through node pruning. Batch, however, replaces the leaving leaf nodes with the joining ones and hence preserves the heights of tree nodes. This distinction implies that Queue-batch requires fewer rekeying operations for the members whose associated leaf nodes are promoted to shallow positions. As a result, the performance gain of Queue-batch is more significant in the presence of frequent membership events.

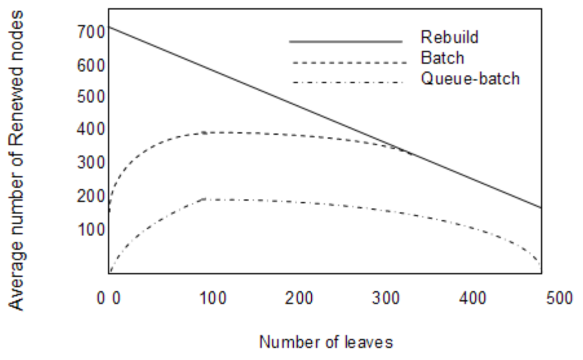

Fig.4 Average numbers of renewed nodes at different numbers of leaves

VII. CONCLUSION

The protocol combines the features of distributive, collaborative and authentication for the key agreement for a dynamic peer group. There is no centralized key server to maintain or distribute the group key. TGDH protocol is used to achieve such distributive and collaborative key agreement. To reduce the rekeying complexity, an interval-based approach is used to carry out rekeying for multiple join and leave requests at the same time, with a tradeoff between security and performance. In particular, the Queue-batch algorithm can significantly reduce both computation and communication costs when there exist highly frequent membership events. It also provides key authentication.

REFERENCES

[1] Y. Amir, Y. Kim, C. Nita-Rotaru, J. L. Schultz, J. Stanton, and G.Tsudik, "Secure group communication using robust contributory key agreement," *IEEE Trans. Parallel Distrib. Syst.*, vol. 15, no. 5, pp. 468–480, May 2004.

[2] G. Ateniese, M. Steiner, and G. Tsudik, "Authenticated group key agreement and friends," in *Proc. 5th ACM Conf. Computer and Communication Security*, Nov. 1998, pp. 17–26.

[3] S. Blake-Wilson and A. Menezes, "Authenticated Diffie-Hellman key agreement protocols," in *Proc. 5th Annu. Workshop on Selected Areas in Cryptography (SAC'98)*, 1998, vol. LNCS 1556, pp. 339–361.

[4] Y. Kim, A. Perrig, and G. Tsudik, "Communication-efficient group key agreement," in *Proc. 17th IFIP Int. Information Security Conf. (SEC'01)*, Nov. 2001, pp. 229–244.

[5] Y. Kim, A. Perrig, and G. Tsudik, "Tree-based group key agreement," *ACM Trans. Inf. Syst. Security*, vol. 7, no. 1, pp. 60–96, Feb. 2004.

[6] P. P. C. Lee, J. C. S. Lui, and D. K. Y. Yau, Distributed collaborative key agreement and authentication protocols for dynamic peer groups The Chinese University of Hong Kong, CS&E Tech. Rep., Jul. 2005.

[7] X. S. Li, Y. R. Yang, M. G. Gouda, and S. S. Lam, "Batch rekeying for secure group communications," in *Proc. 10th Int. World Wide Web Conf. (WWW10)*, Orlando,
FL, May 2001, pp. 525–534.

[8] A. Perrig, "Efficient collaborative key management protocols for secure autonomous group communication," in *Int. Workshop on Cryptographic Techniques and E- Commerce (CrypTEC '99)*, Jul. 1999, pp. 192–202.

[9] M. Steiner, G. Tsudik, and M. Waidner, "Key Agreement in dynamic peer groups," *IEEE Trans. Parallel Distrib. Syst.*, vol. 11, no. 8, pp. 769–780, Aug. 2000.

[10] M. Waldvogel, G. Caronni, D. Sun, N. Weiler, and B. Plattner, "The versakey framework: versatile group key management," *IEEE J. Sel. Areas Commun.*, vol. 17, no. 9, pp. 1614–1631, Sep. 1999.