Journal of Computer Science 7 (8): 1212-1217, 2011
ISSN 1549-3636
© 2011 Science Publications

An Effective Contributory Re-Keying Approach to Compute Conference Key

[1]D. Thilagavathy and [2]M. Rajaram
[1]Department of CSE, Adhiyamaan College of Engg Hosur, India
[2]Vice Chancellor of Anna University of Technology, Tirunelveli, India

Abstract: Problem statement: With the explosive growth of and internet and web applications many emerging group-oriented distributed applications such as tele/video-conferencing, multiplayer games are based on group communication model that need security services such as privacy and data integrity Hence a secure distributed group key agreement is required to establish and authenticate a common group key for secure and private communication. There is a need for security services to provide group-oriented communication privacy and data integrity. It is important that members of the group can establish a common secret key for encrypting group communication. A key tree approach has been proposed by many authors to distribute group key in such a way that the rekeying cost scales with the logarithm of the group size for a join or leave request. The efficiency of this key tree approach critically depends on whether the key tree remains balanced over time as members join or leaves **Approach:** Instead of performing individual re-keying operations, an interval-based approach of re-keying is adopted in the proposed scheme. **Results:** In the proposed scheme Queue-merge algorithm is used for rekeying which substantially reduces the computation cost and communication cost. The comparison shows that queue merge algorithm performs better than Batch algorithm in terms of minimizing the key tree and presumes better node density thereby reducing the computation cost. **Conclusion:** Performance comparison also shows reduced number of renewed nodes for various rekeying interval which reduces the communication cost.

Key words: Authentication, dynamic peer groups, group key agreement, secure group communication, re-keying approach, exchange protocol, ad-hoc network, re-keying efficiency, queue-batch algorithm

INTRODUCTION

Distributed group key agreement protocol is different from traditional centralized group key management protocols. Centralized protocols rely on a centralized key server to efficiently distribute the group key. An excellent body of work on centralized key distribution protocols exists. In those approaches, group members are arranged in a logical key hierarchy known as a key tree. Using the tree topology, it is easy to distribute the group key to members whenever there is any change in the group membership (e.g., a new member joins or an existing member leaves the group).

In the distributed key agreement protocols we consider, however, there is no centralized key server available. This arrangement is justified in many situations-e.g., in peer-to-peer or ad hoc networks where centralized resources are not readily available.

Moreover, an advantage of distributed protocols over the centralized protocols is the increase in system reliability, because the group key is generated in a shared and contributory fashion and there is no single-point-of-failure. In the special case of a communication group having only two members, these members can create a group key using Diffie–Hellman key exchange protocol (Diffie and Hellman, 1976). In the protocol, members and use a cyclic group of prime order with the generator. They can generate their secret exponents. Member can compute its public key and send it to receiver. Since both members know their own exponent, they can each raise the other party's public key to the exponent and produce a common group key. Using the common group key and can encrypt their data to prevent eavesdropping by intruders.

MATERIALS AND METHODS

To prevent a new user from reading past communications (backward confidentiality) and a departed user from reading future communications (forward confidentiality) (Diffie and Hellman 1976), the re-keying, which means renewing the keys associated with the nodes of the key tree, is performed. In this study, we propose, based on the tree-based group

Corresponding Author: D.Thilagavathy, Department of CSE, Adhiyamaan College of Engineering, India

Diffie-Hellman protocol (Kim *et al*., 2001), several group key agreement protocols for a dynamic communication group in which members are located in a distributed fashion and can join and leave the group at any time.

Related work: Diffie and Hellman (1976) proposed the first two-party single-round key agreement protocol. Joux proposed a single-round three party key agreement protocol that uses bilinear pairings. Burmester and Desmedt (1995) had proposed a multiparty two-round key agreement (BD) protocol using a ring structure of participants. The BD protocol makes the active adversary control over the channel of all these protocols. These protocols assume only a passive adversary and justify their security on purely heuristic models.

Burmester and Desmedt proved that their ring structure based group key agreement protocol is secure against a passive adversary in standard model under decision Diffie-Hellman (DDH) assumption. Several variations of Diffie-Hellman protocol and Kim *et al*., (2004) protocol have been suggested to incorporate authentication and a trial and error approach has been adopted to provide informal security.

To achieve secure group communication and re-keys at each join or leave event. Li *et al*. (2001.) and Yang *et al*. (2001), then apply the periodic re-keying concept in Kronos (Setia *et al*., 2005) to the key tree setting. All the key-tree-based approaches (Sherman *et al*., 2003) require a centralized key server for key generation. Burmester and Desmedt (1995) propose a computation-efficient protocol at the expense of high communication overhead. Steiner *et al*. (Ateniese *et al*., 1998) propose Cliques, in which every member introduces its key component into the result generated by its preceding member and passes the new result to its following member.

Cliques are efficient in re-keying for leave or partition events, but imposes a high workload on the last member in the chain. Kim *et al*. (2001) propose TGDH, which arranges keys in a tree structure. The setting of TGDH is similar to that of the One-Way Function Tree (OFT) scheme (Sherman *et al*., 2003) except that TGDH uses Diffie–Hellman instead of one-way functions for the group key generation. Kim *et al*. (2001) also suggest a variant of TGDH called STR which minimizes the communication overhead by trading off the computation complexity. All the above schemes are decentralized and hence avoid the single-point-of-failure problem in the centralized case, though they introduce high message traffic due to distributed communication. A reference (Kim *et al*., 2004) considers re-keying at single join, single leave, merge, or partition events. Our work considers a more general case that consists of a batch of join and leave events.

Comparison between the centralized and decentralized re-keying is studied by Amir *et al*. (2004). In particular, Amir *et al*. (2004) suggest a centralized key distribution scheme based on Cliques (Ateniese *et al*., 2003) and compare the performance of both schemes. In contrast, our work compares the centralized and decentralized key management schemes adapted from a key tree setting. Rather than emphasize the re-keying efficiency, (Ateniese *et al*., 2003) focus on the security issues and develop authenticated group key agreement schemes based on the Burmester-Desmedt model, Cliques and TGDH, respectively

Group key establishment: The tree-based group Diffie-Hellman (TGDH) protocol (Diffie and Hellman, 1976) 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:

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

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:

$$\begin{aligned} K_V &= (BK_{2v+1})^{K2V+2} \bmod p \\ &= (BK_{2v+2})^{K2V+1} \bmod p \\ &= \alpha^{K2V+1\,K2V+2} \bmod p \qquad (2) \end{aligned}$$

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.

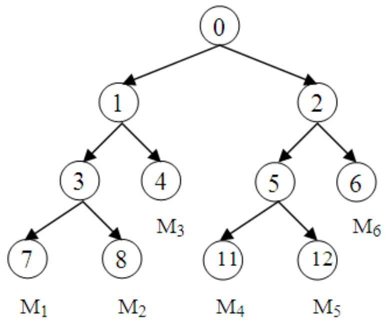

Fig. 1: Key tree used in the tree-based group diffie-hellman protocol

To illustrate, consider the key tree in Fig. 1. Every member M_i generates their 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 M4, 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.

Secured group key communication: The secured group key authentication communication model comprises of the following phases

Tree based group key in dynamic peers: Tree based group Diffie-Hellman is used to efficiently maintain the group key in a dynamic peer group with more than two members. Each member maintains a set of keys, which are arranged in a hierarchical binary tree. We assign a node ID to every tree node. For a given node, we associate a secret (or private) key and a blinded (or public) key. All arithmetic operations are performed in a cyclic group of prime order with the generator. 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 secret key of a non leaf node can be generated by the secret key of one child node of and the blinded key of another child node. 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.

To provide both backward confidentiality (i.e., joined members cannot access previous communication data) and forward confidentiality (i.e., left members cannot access future communication data), re-keying, is performed whenever there is any group membership change (join of new member or leaving of existing member).

Individual rekeying: Individual re-keying is performed after every single join or leave event. 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 or departed member. Tree group key use the convention that the rightmost member under the sub tree 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.

Individual rekeying, that is, rekeying after each join or depart request, has two drawbacks (Lee, 2003; Perrig, 1999). First, it is inefficient since each rekey message has to be signed for authentication purposes and a high rate of join/depart requests may result in performance degradation because the signing operation is computationally expensive. Second, if the delay in a rekey message delivery is high or the rate of join/ depart requests is high, a member may need a large amount of memory to temporarily store the rekey and data messages before they are decrypted.

Interval based distributed re-keying: Interval based distributed re-keying algorithms significantly reduce the computation and communication costs of maintaining the group key. The interval-based approach provides re-keying efficiency for dynamic peer groups while preserving both distributed (i.e., no centralized key server is involved) and contributory (i.e., each member contributes to the resulting group key) properties. Interval-based re-keying maintains the re-keying 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 key.

The interval-based algorithms are developed based on the following assumptions. The group communication satisfies view synchrony that defines reliable and ordered message delivery under the same membership view. Intuitively, when a member broadcasts a message under a membership view, the message is delivered to same set of members viewed by the sender. Note that this view-synchrony property is essential not only for group key agreement, but also for reliable multipoint-to-multipoint group communication in which every member can be a sender. Since the interval-based re-keying operations involve nodes lying on more than one key path, more than one sponsor may be elected. Also, a renewed node may be re-keyed 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.

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

Interval-based re-keying maintains the re-keying 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 re-keying is done through the approaches Rebuild algorithm, the Batch algorithm and the Queue-batch algorithm.

Rebuild and batch algorithm: The Rebuild algorithm minimizes the resulting tree height so that the re-keying operations for each group member can be reduced. At the beginning of every re-keying interval, reconstruct the whole key tree with all existing members that remain in the communication group, together with the newly joining members. The resulting tree is a left-complete tree, in which the depths of the leaf nodes differ by at most one and those deeper leaf nodes are located at the leftmost positions. Rebuild is suitable for some cases, such as when the membership events are so frequent that we can directly reconstruct the whole key tree for simplicity, or when some members lose the re-keying information and the simplest way of recovery is to re-build the key tree.

Batch rekeying techniques have been recently presented as a solution to overcome this problem. In such methods, a departed user will remain in the group longer and a new user has to wait longer to be accepted. All join and leave requests received within a batch period are processed together at the same time. A short rekey interval does not provide much batch rekeying benefit, whereas a long rekey interval causes a delay to joining members and increases vulnerability from departing members who can still receive the data.

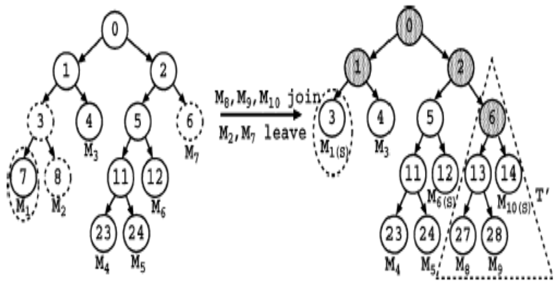

Fig. 2: Example of the queue-merge phase

Given the numbers of joins and leaves within a re-keying interval, we 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: Rebuild and batch re-keying approaches perform all re-keying steps at the beginning of every re-keying interval. This results in high processing load during the update instance and thereby de-lays the start of the secure group communication.

Thus a more effective algorithm Queue-batch algorithm is proposed to develop. It reduces the re-keying load by pre-processing the joining members during the idle re-keying interval. The Queue-batch algorithm is divided into two phases, namely the Queue-sub tree phase and the Queue-merge phase. The first phase occurs whenever a new member joins the communication group during the re-keying interval. In this case, append this new member in a temporary key tree. The second phase occurs at the beginning of every re-keying interval and we merge the temporary tree (which contains all newly joining members) to the existing key tree. To illustrate consider Fig. 2 where $M_{8,}$ M_3 ,M_{10} join and $M_{2,}$ M_7 leave and a temporary tree is merged with the existing tree.

Pseudo-code of the queue sub tree phase:
Queue-sub tree (T'):

```
if (a new member joins){
if(T'==NULL)/*no new member in T'*/
create a new tree T' with the only new member;
else{/*there are new members in T'*/
find the insertion node;
add the new member to T';
elect the rightmost member under the sub tree
rooted at the sibling of the joining node to be
the sponsor;
if(sponsor)/* sponsor's responsibility*/
re-key renewed nodes and broadcast new
Blinded keys;
}
}
```

Pseudo-code of the Queue merge phase:
Queue-merge (T, T’, M^l, L):

```
if (L==0){/* There is no leave*/
add T’ to either the shallowest node
(Which need to be the leaf node) of T
such that the merge will not increase the
resulting tree height, or the root node of T
if the merge to any location will increase
the resulting tree height;
} else {/* there are leaves*/
add T’ to the highest leave position of the key tree T;
remove remaining L-1 leaving leaf nodes and promote
their siblings;
}
elect members to be sponsors if they are the rightmost
members of the sub tree model rooted at the sibling
nodes of the departed leaf nodes in T, or they are the
rightmost member of T’;
if(sponsor)/*sponsor’s responsibility*/
re-key renewed nodes and broadcast new blinded keys;
```

Analysis of the queue-batch algorithm*:* The main idea of the Queue-batch algorithm exploits the idle re-keying interval to pre-process some re-keying operations. When we compare its performance with the Rebuild or Batch algorithms, we only need to consider the re-keying operations occurring at the beginning of every re-keying interval. When J = 0, Queue-batch is equivalent to Batch in the pure leave scenario. For J>0, the number of renewed nodes in Queue-batch during the Queue-merge phase is equivalent to that of Batch when J = 1.

Performance evaluation on secured group key management: To reflect the latency of generating the latest group key for data confidentiality, we evaluate the performance of the interval-based algorithms using simulation-based experiments. Our simulation results show the Queue batch algorithm performs best among the others.

The analysis of the two proposed algorithm are based on two performance measures i.e., number of exponentiation operations and the number of renewed nodes. The number of exponentiation operation gives a measure of the computation load in terms of node density to communication group’s packets drop (Fig. 3). The number of renewed nodes is said to be renewed if it is a non leaf node and its associated keys are renewed. These metric measures the communication cost since the new blinded keys of the renewed nodes have to be broadcast to the whole group (Fig. 4).

RESULTS AND DISCUSSION

The existing key tree is completely balanced prior to the interval-based re-keying event. Every existing member has the same leave probability. The computation of the blinded group key of the root node is counted in the blinded key computations. With this assumption, the number of blinded key computations simply equals the number of renewed nodes, provided that the blinded key of each renewed node is broadcast only once.

The figure above provides an inference of the batch re-keying to queue batch re-keying algorithm performance comparison in terms of re-keying interval to the no. of renewed nodes. It shows reduced no of renewed nodes for queue batch re-keying algorithm compared to that of batch re-keying model in various re-keying intervals.

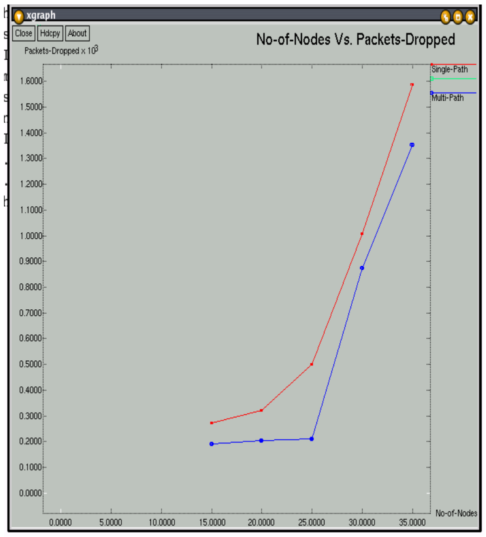

Fig. 3: Node density and packets dropped

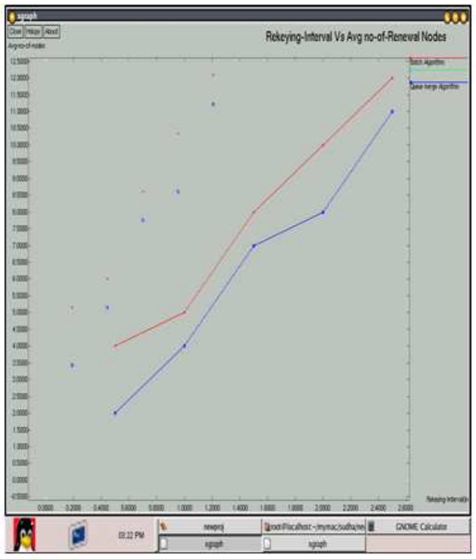

Fig. 4: Rekeying nodes and no. of renewed nodes

CONCLUSION

The proposed model of this study provides a distributed collaborative key agreement protocols for dynamic peer groups. The key agreement setting is performed in which there is no centralized key server to maintain or distribute the group key. We show that one can use the TGDH protocol to achieve such distributive and collaborative key agreement. To reduce the re-keying complexity, we propose to use an interval-based approach to carry out re-keying for multiple join and leave requests at the same time, with a tradeoff between security and performance.

Our simulation results shows that the Queue-batch algorithm can significantly reduce both computation and communication costs when there is highly frequent membership events. The proposal also addresses both authentication and implementation for the interval-based key agreement algorithms.

REFERENCES

Amir, Y., Y. Kim, C. Nita-Rotaru, J.L. Schultz and J. Stanton *et al*., 2004. Secure group communication using robust contributory key agreement. IEEE Trans. Parallel Distributed. Syst., 15: 468-480. DOI: DOI: 10.1109/TPDS.2004.1278104

Ateniese, G., M. Steiner and G. Tsudik, 1998. Authenticated group key agreement and friends. Proceedings of the 5th ACM Conference on Computer and Communication Security, (CCS '98), ACM New York, USA., pp: 17-26. 10.1145/288090.288097

Burmester, M. and Y. Desmedt, 1995. A Secure and Efficient Conference Key Distribution System. Adv. Cryptol., 950: 275-286. DOI: 10.1007/BFb0053443

Diffie, W. and M. Hellman, 1976. New directions in cryptography. IEEE Trans. Inf. Theory, 22: 644-654. DOI: 10.1109/TIT.1976.1055638

Kim, Y., A. Perrig and G. Tsudik, 2001. Communication Efficient Group Key Agreement. In: Trusted information, Springer, New York, ISBN: 0792373898, pp: 229-244

Kim, Y., Y. Kim and G. Tsudik, 2004. Tree-based group key agreement. ACM Trans. Inf. Syst. Sec., 7: 60-96. DOI: 10.1145/984334.984337

Lee, P.P.C., 2003. Distributed and Collaborative Key Agreement Protocols with Authentication and Implementation for Dynamic Peer Groups. M. Phil. Thesis. The Chinese University of Hong Kong.

Li, X.S., Y. R. Yang, M.G. Gouda and S.S. Lam, 2001. Batch rekeying for secure group communications. Proceeding 10th International Conference on World Wide Web, (WWW'01), ACM New York, NY, USA, pp: 525-534. DOI: 10.1145/371920.372153

Perrig, A., 1999. Efficient Collaborative Key Management Protocols for Secure Autonomous Group Communication. International Workshop on Cryptographic Techniques and E-Commerce, (CrypTEC '99), CiteSeerx, USA., pp: 192-202. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.42.4440

Setia, S., S. Koussih and S. Jajodia, 2000. Kronos: A Scalable Group Re-Keying Approach for Secure Multicast. Proceeding IEEE Symposium Security and Privacy, May, 14-17, IEEE Xplore Press, pp: 215-228.DOI: 10.1109/SECPRI.2000.848459

Sherman, A.T. and D.A. McGrew, 2003. Key establishment in large dynamic groups using one-way function trees. IEEE Trans. Software Eng., 5: 444-458. DOI: 10.1109/TSE.2003.1199073