IOSR Journal of Computer Engineering (IOSRJCE)
ISSN: 2278-0661, ISBN: 2278-8727 Volume 5, Issue 2 (Sep-Oct. 2012), PP 36-40
www.iosrjournals.org

Securing Group Communication in Partially Distributed Systems

[1]Pankesh Bamotra, [2]Prashant Dwivedi, [3]Nishant Gupta, [4]Rajat Pandey
([1,2,3,4] Computer Science & Engineering, Vellore Institute of Technology, India)

Abstract :*This paper deals with symmetric key exchange in partially distributed systems. Unlike the traditional distributed approaches for key exchange utilized when we have a centralized KDC (Key Distribution Center), which is a trusted server responsible for key exchange between the users involved in group communication or a KDC for each node, we divide the group of nodes in regions with each having its own KDC. Each user can communicate securely with members of its own region as well as with those belonging to other regions. We use a hierarchical approach to represent the partial distributed structure of the distributed system using key graphs. The outcome is a secure group communication providing authenticity, confidentiality and integrity of messages delivered between groups. Each secure group is represented in form of a triplet (U, N, R) where U is the set of users, N is the set of keys held by the users and R represents a relation on U and R. This approach has been developed keeping in view the scalability issues of the distributed systems where number of group members may increase or decrease with time. For the same reason we use rekeying strategies to redistribute the keys every time a user joins or leaves the group. This approach is irrespective of frequency of joins and leaves.*

Keywords – *Group communication, multicast, partially distributed systems, rekeying, security*

I. Introduction

In many distributed applications like time server, *group communication* plays an important role in multicasting the information from one authentic user to many target users. Consider one such case in which the communication network of a business firm is divided into a distributed network with one branch in New Delhi, second in Calcutta and third at Hyderabad. There is a project on which employees from various branches are working together on. If we use group communication approach then the contribution of one team member can be made available to other colleagues remotely without incurring much cost. Now suppose that a new employee or anyone who wants to join the project but on the condition that he shall not have access to previous transactions of the already existing members. We may have the protocol as follows. Whenever a person joins the group the access mechanism (we shall use keys) is dynamically changed so that the new member can't access the previous message exchanged between the other members. Similarly whenever a member leaves the group the access mechanism to the data is again dynamically changed so the leaving member can't access any future transactions which take place between other members of the group in the near future.

Now talking about the partially distributed systems we shall first describe distributed systems formally. Distributed systems are an abstraction of multiple network systems connected together locally or located geographically far apart but behave as if a single system exist. There is abstraction of resources viz. the user is unaware of the location, name, number of copies of that resource. Now with respect to our protocol we discuss about the partially distributed systems. In this case various nodes (collection of systems) are segregated as *regions* as described in the figure (Fig. 1) below. Here the communication inside the region follows a usual protocol in which the sender needs to know about the *public key* of the receiver who decrypts the message using his *private key*. However cross-region communication involves both public and private-key cryptosystems and *symmetric key* as well.

In the initial setup all the users of all the nodes have to authenticate themselves with a trusted server, the KDC here. In this process each of the users is allotted an ID and a public-private key pair which is used for further communications. This is what is referred to as *user-key*. For the group communication the KDC also sends to each user/ member a group key to be shared by all members of the group. This gives an extension to the above example where suppose that different departments of different branches contribute to the project and only members of that project and now of that department can access their respective contributions. For example say that developer group of the project would like only developers to see the project design reports while coding team may want only coders to have access to the java modules. This seems to give a communication tradeoff in linear proportion with number of users 'n'. However the one-to-one initial setup cost is incurred only once i.e. at the beginning. Only changing the keys for each departure or entry comprises the real communication cost.

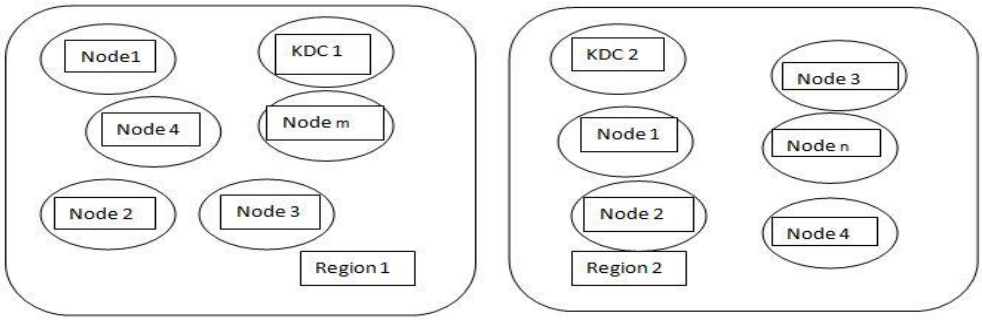

Fig. 1 A simple partially distributed architecture

Here, the KDC is responsible for rekeying after every departure and entry happens anywhere in that region. As a result a new *region-key*, *node-key*, *individual key* and an *ID* is given to the new member (refer [1]) while the already existing members are broadcasted a message about the new *region-key* encrypted using the previous group key along-with new node key to existing members of that node.

II. APPROACH

The approach that this protocol uses has been discussed in the paper published by Wong et. al. [2]. The notion here slightly differs in that instead of subgroups we consider nodes and regions to be analogous to groups and thus we have a notion of node-key and region-key. To represent a secure group/region we use a triplet (U, N, R), where U represents number of users, N denotes the key set and R represents a relation between U and N which in mathematical terms is given as $R \subset U \times N$. This representation can be made clear from the following illustration. Let us consider the former example of a company project. Let there be an ongoing project in company in which all the teams including the analysis team, design team and implementation team are currently working on. Thus they form the three nodes in our case. The trusted serves distributes them the key in following way. In the initial setup phase the individual keys are given to each user along-with their node key and the region key. Now suppose that in the design team one of the members is transferred from this project then the procedure is as follows. The node key for thedesigning team has to recomputed by the KDC/ trusted server and multi-casted to the remaining members.

III. Tree Key Graphs

Key graphs are a diagrammatic representation Fig. 2 of KDC along with the nodes and the users at the lower levels. It is basically a directed acyclic graph i.e. it has no cycles. There are two kinds of nodes in the graph – U-nodes represented by a square and K-nodes representing the keys. Following are some properties about key graphs:-

1) There is a bijection between set U and the set of u-nodes in graph.
2) There is a bijection between set N and the set of k-nodes in graph.
3) $(u,k) \in R$ iff there exists a directed path from that u-node to k-node.

We use two functions associated with each secure group.

$$USERSET(k) = \{u \mid (u, k) \subset R\}$$
$$KEYSET(u) = \{k \mid (u, k) \subset R\}$$

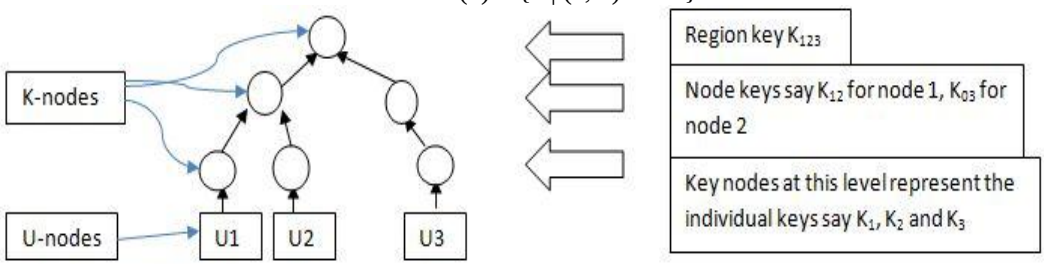

Fig.2 Various types of nodes in Tree key graph

The tree key graphs are a special case of this key graphs in which there is only a single root. Every such tree is has following two parameters:-

1) Height of the tree - It refers to the longest path in terms of edges in a key graph.
2) Degree of the tree – It refers to maximum number of edges incident on a node.

It is quite obvious from the figure itself that each path ends with a user node and hence each user has number of keys equal to the height of the tree at most.

3.1 Rekeying strategy using group-oriented protocol

Now we discuss about the rekeying concept which is the core point of this paper. Before going to the rekeying strategy let's talk about the role of rekeying and some useful notations. After the initial setup each of the existing users in a region has his set of keys. Now suppose a new user wants to join a node, he sends a request message to the KDC requesting for the new region and node key (refer [1]). To achieve this, the KDC sends rekey messages having both the newly generated region key/node key to the respective users. Following notation has been used to represent this context:-

$$A \rightarrow B\ (Z)$$

Where

1. If B is a single user then representation means sending the message from Z from A to B.
2. If B is a set of users in a node then it stands for a multicast /unicast message from A to B.

3.2 Rekeying during joining in a tree key representation

After the KDC receives the joining request from the user 'r' in the node 'n' (say) the new keys have to be distributed to the joining user as well as the existing users so the new user doesn't have access to previous transactions. To do so the KDC shall create a new 'k' node, k_u for the new user and finds an existing 'k' node representing a node (to which 'r' belongs) in the region and chooses it to be its parent node which Wong. et. al[2]. refers to as the *joining point.* After this the rekeying messages are constructed, encrypted and sent to the respective nodes. As represented in the Fig.3 below, the parent node of the new user's k_u node will get a new *node key* and also the *region key* will change.

As shown in the Fig. 3, user U_4 joins the group by sending joining request to the KDC server it is assigned an individual key K_4. The *joining node* here is K_{03}. We see that the region key changes from $K_{1\text{-}3}$ to $K_{1\text{-}4}$ and the *node key* at K_{03} changes to K_{34}. We also see that users U_1, U_2 need to have only the new region key $K_{1\text{-}4}$ because there is no change in its *node key* while users U_3, U_4 needs new region key $K_{1\text{-}4}$ as well as new node key K_{34}.

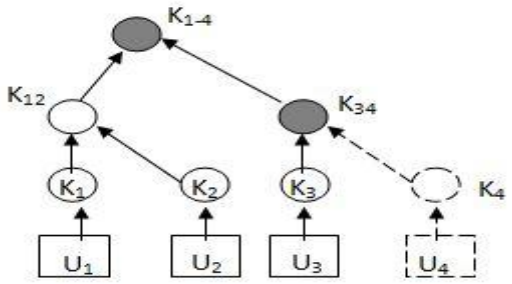

Fig.3 Joining of a user in tree key graph

To distribute the new keys to the respective users the server constructs the rekey messages which is a collection of encrypted keys which is decrypted by the user using appropriate keys. We use group oriented protocol to achieve this rekeying. We now discuss on this approach in the event of the user joining the group.

One of the key points while discussing distributed systems is the communication traffic. Thus we choose to use such a protocol which tries to minimize the total number of intra- region messages. In group-oriented protocol the KDC frames a single rekeying message whose length is $O(\log_d(n))$ for a region with 'n' nodes and key tree degree 'd'. Thus there is an obvious advantage in terms of network traffic as only one multicast message is needed to be sent to all the existing users in the region and a single unicast message to the joining user. In our example also there would be one multicast message and one unicast message depicted as following:-

1. KDC $\rightarrow$ $\{U_1,U_2,U_3\}$: $(K_{1\text{-}4})_{K1\text{-}3}$,$(K_{34})_{K3}$
2. KDC $\rightarrow$ $\{U_4\}$: $(K_{1\text{-}4},K_{34})_{K4}$, here the subscripts in the message refers to encryption key

Though we have reduced the number of messages but the encryption cost remains to $2(h\text{-}1)$, where 'h' is height of the tree.

3.3 Rekeying during leaving in a tree key graph representation

The user node $K_u$in Fig. 4 sends the request for leaving request to the KDC which finds the parent of the K_u and removes the corresponding K node and U node for the user 'u'. The parent of K_u is referred to as the *leaving point.* Now to prevent the leaving user from accessing the future transactions the rekeying is again performed and new keys are assigned. There is a new node key for the members of the node of which user 'u' was the part and also the region key is generated and distributed. In our example the rekeying follows group oriented protocol so following messages are transmitted to respective users.

1. $M_1 = (K_{1\text{-}3})_{K12},(K_{1\text{-}3})_{K3}$
2. $M_2 = (K_{03})_{K3}$

3. KDC → { U_1,U_2,U_3}: M_1, M_2

We see that this message is'd' times bigger than join message and the encryption cost is *d(h-1).*

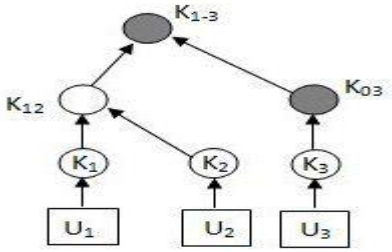

Fig.4 Leaving by a user in tree key graph

IV. Inter And Intra Region Communication

In this section we discuss how inter-region and intra-region communication takes place. To begin with we first discuss basic concepts of cryptography. First is symmetric cryptography which is the cryptography protocol that we use in our model of partial distributed systems for inter-region communication while for intra-region communication we use the well-known concept of public-key cryptography. In symmetric cryptography there is a shared secret key known only to the communicating parties and no one else in between. While in public-key cryptography(E.g. RSA) each communicating party has a pair of keys, the public key which is known to all and the private key which is kept secret. Till now we have discussed only the intra-node scenario of rekeying which helps established a security amongst the users inside a particular region that new users can't have access to past transactions of the existing users and leaving users can't have access to future transactions. Now we discuss how the communication takes place after the things are established.

The intra-node communication is not much of an importance so we discuss it in brief. When the users belonging to the same region want to communicate with each other they first contact the KDC to know the node key and the public key of the other user with whom it wants to communicate with. Then it encrypts the message using the two keys which is decrypted by the other user using its prior knowledge of its node key and its secret key. Thus the communication remains secure. The inter-node communication is a more likely and interesting scenarios which involves parties or the users belonging to different regions. The phenomenon of symmetric key distribution is depicted below in the Fig. 5.

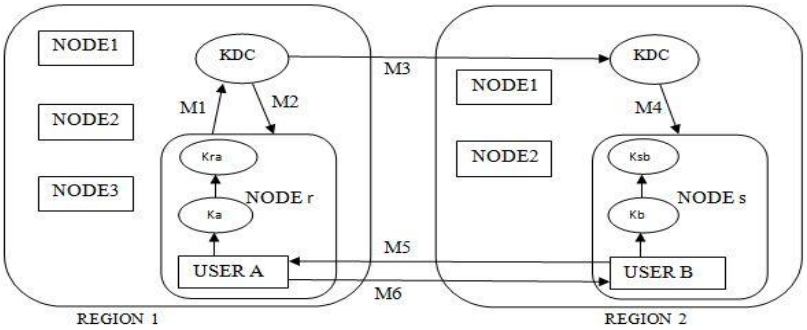

Fig.5 Key exchange in partially distributed systems for group communication

M1= (Req_a, ID_a, ID_b) **Req_q**: request code by user A , **ID_a, ID_b**: Identifiers of user A and user B

M2= E((Req_a,ID_a,K_{ab},),(K_{ra},K_a)) **K_{ra}**: node key of user A , **K_a**: individual key of user A, **K_{ab}**: shared key

M3= E((K_{ab},K_{ra}, ID_a,ID_b),K_2) **K_2**: secret key of KDC at region

M4= E((K_{ab}, K_{ra}, ID_a),(K_{sb},K_b)) **K_{sb}**: node key of user B , **K_b**: individual key of user B

M5= E((N_{rand}, K_{sb}), K_{ab}) **N_{rand}**: random number generated by user B, **K_{sb}**: node key of user A

M6= E (N_t,K_{ab}) **N_t=f(N_{rand})**: f is a previously decided function

Now we discuss the phenomena happening in the figure above. The distribution of the secret depends on at what locations the user A and user B exists. If they lie in the same region then we can use a fairly simple centralized approach but when both the users lie in different regions the phenomena is as above in the figure. It is assumed that the KDC is a trusted party that distributes the keys to all the users and holds a table containing information about the individual/ secret keys of the users in that region as well as the node keys and the secret keys of all other KDCs too.The process of key exchange happens as below :-

1. User A sends the request to KDC of its own region consisting of code request, its identifier and the identifier of the user (B) it wants to communicate with in the other region.

2. The KDC of A's region then acknowledges this message by encrypting the secret shared key (generated by the KDC) for secure communication between A and B with the node key and individual key of the user A.
3. The KDC_A then sends a message containing the shared key K_{AB}, node key of A and identifiers of A and B encrypted with the secret of KDC_B which is also a trusted party.
4. KDC_B then extracts the information from the message sent by KDC_A and sends the shared key and A's identifier to user B encrypted with its node key and individual key.
5. To authenticate/ validate the genuineness of user a user B sends a dummy message containing a random number and its node key encrypted with the shared key K_{AB}. If A is the genuine person who wants to communicate with B then he is also the owner of this shared key at the first place. A then finds a functional value ($f(N_{rand})$) of the random number and sends back a message to B encrypted by K_{AB}. Now the user B is sure of the authenticity of user A and both have the shared keys too. Thus A and b can communicate securely using this shared key.

V. Conclusion

This paper is an extension to work done on hierarchical group communication. We have basically provided the application of hierarchical group communication with respect to the partially distributed systems which divide the users in terms of regions and nodes. The aim is to provide secure communication to the users in the system so that each time a user joins or leaves the system he doesn't have access the previous transactions or the future transactions. We used the concept of trees to achieve this hierarchical representation of users in the system. In this paper we have kept in mind the limitations of the distributed systems which may under perform during high network traffic so we used the group-oriented protocol for rekeying. Inter-region secure communication was achieved by using the concepts of public-key cryptography.This paper can serve as a platform for developing partially distributed system for practical scenarios as discussed in the introduction of this paper. Systems which involve group communication and need its communication to be secured on each leave and join in the system can use this paper as its baseline.

References

Books:

[1] Pradeep K. Sinha, "Distributed operating systems concepts and design", PHI learning private limited, 2007

Journal Papers:

[2] Chung Kei Wong, *Member, IEEE*, Mohamed Gouda, and Simon S. Lam, *Fellow, IEEE, "Secure Group Communications Using Key Graphs*" IEEE/ACM TRANSACTIONS ON NETWORKING, VOL. 8, NO. 1, FEBRUARY 2000