International Journal of Soft Computing 3 (3): 182-188, 2008
ISSN: 1816-9503

Multicast Key Distribution Issues and Strategies

[1]V. Vasudevan and [2]R. Sukumar
[1]Department of Information Technology, Arulmigu Kalasalingam College of Engineering, India
[2]Department of Computer Science and Engineering, Sethu Institute of Technology, India

Abstract: In wireless networks high-valued transmissions would need to be encrypted and only paying users should have the decryption keys. As a result, it is very important to securely distribute the multicast data encryption key to a set of legitimate subscribers. Secure key distribution is one of the significant challenges in secure multicast over wireless networks. In this study, we analyze number of key distribution schemes that offer a novel solutions for securely distributing the group key and re-key messages for joining and leaving a mobile host in secure multicast group. We focus on 3 different approaches of group key management The Modified Huffman Technique, Key Management Using One-Way Function Trees and Kronos. We perform a comparative study on these three techniques. We also study the issues related to multicast key management and propose some solutions to overcome those difficulties.

Key words: Multicast, wireless networks, key management, scalability, group authentication, modified huffman technique, one-way function trees, kronos

INTRODUCTION

Many emerging applications, such as video conferencing, pay per view TV and stock market applications, are based upon a group communications model (Moyer *et al*., 1999). They have lead to the design and implementations of multicast model that provide efficient delivery of data from a source to a group of recipients. Applications such as pay-per-view distribution of digital media, pay-per-use multi-party games and restricted conferences fall in the category where the receiver set need to be restricted to legitimate subscribers. In addition, for applications such as multicast distribution of stock market information it is also important to authenticate the data source. A secure multicast system supported by key management generally offers the following security features.

Authentication: There are 2 types of authentication, depending on whether the sender or receiver is verified (one of the current techniques that supports user verification is digital signature (Ran and Benny, 1998)). Sender authentication is the process of reliably verifying the identity of the sender. Receiver authentication, on the other hand, aims at verifying the identity of the receiver.

Data confidentiality: Confidentiality means that the content of a message must be shared only by authorized users. Unauthorized intruders should not be able to make sense out of the message by simply sniffing (or eaves dropping) on it. In unicast communication, it can be achieved by encrypting the data using, for example, the well-known RSA (Debby *et al*., 1997). However, offering data confidentiality is more difficult in the multicast environment because a large number of users may be involved and membership may change quite rapidly.

Integrity: Integrity means that if the data is altered in transit over the network, such an alteration should be detected. One method to achieve this is Message Digest (e.g., MD5) (Ballardie and Crowcroft, 1995). Efficient techniques for unicast communication may be straight forwardly applied in the multicast environment. In addition to the above, other features need to be provided, such as traffic confidentiality (i.e., protection of traffic information such as its patterns from disclosure); non-repudiation (i.e., neither the sender nor the receiver of a message can deny the transmission); access control (i.e., access to information resources may be controlled by or for a system); and service assurance (i.e., resistance to certain attacks, such as denial of service). These features, however, are not related to key management issues and hence are out of the scope of this study. Data confidentiality is one of the most challenging problems in secure multicast. To achieve this, a secure multicast scheme must address key management issues, which include efficient organization and distribution of keys with low communication overheads, key storage cost and scheme complexity. This study reviews a number of such issues.

Corresponding Author: R. Sukumar, Department of Computer Science and Engineering, Sethu Institute of Technology, India

GROUP KEY MANAGEMENT: ISSUES

There are a number of issues related to multicast security in general and more specifically group key management.

Multicast application types: Two general multicast application types have been identified in the effort to provide a common ground for discussing the issues related to multicast security and group key management.

One-to-many multicast applications: The first multicast application type covers the cases where the multicast group has one sender and multiple receivers. Transmission is unidirectional from one sender to many receivers. The receivers are assumed to be passive consumers of the data, while the single sender is the producer of the data. The initiator of the group is assumed to be its owner and for simplicity it is also assumed to be the sender. Examples this multicast application includes Pay Per View (PPV) programs (e.g. Internet TV, Radio, Video) and other real-time data (e.g. news, stock prices, etc).

Two general cases exist with respect to the data being transmitted. In the first case, the group is concerned more about the authenticity and integrity of the data and not so much its confidentiality. An example would be subscriptions to publicly available data (e.g., Stock market data, government publications). These desired effects can be achieved using public key techniques and message integrity techniques, leaving the data itself readable to non-members. Of more concern here is the second case, where the aim is to prevent non-members from accessing the data. An example would be subscriptions to subscribers-only transmissions (e.g., pay per view Internet TV). Here, encryption techniques can be used for controlling access to the data.

Many-to-many multicast applications: The Many-to-Many multicast application type refers to the case where the relationship between the sender and receiver(s) is equal (democratic) and where the data is of immediate value only to the members of the group. Every member of the multicast group is both a sender and a receiver. An example of this multicast application would be conferencing. Membership of the group maybe Open or Closed. In the Open Many-to-Many multicast anyone can join the conference provided that the identity of the member is known. In the Closed Many-to-Many multicast only a predefined number of members can join and the identities of the members are known in advance.

The aim in the Many-to-Many multicast application type is to prevent non-members from accessing the data. Hence, encryption (in addition to message authenticity and integrity techniques) is used for controlling access to the data that is of immediate value only to the members of the group.

Size and distribution of group members: The effectiveness of a group key management protocol and its underlying multicast routing protocol is dependent to a certain extent on the size of the group and on the distribution of the group (dense or sparse). Related to this is also the issue of the frequency of changes to the membership, which may lead to the need of re-keying and other changes in the security parameters of the group.

Scalability of protocols and membership management: One of the primary issues in group key management is that of the scalability of the protocols it employs. Often these protocols rely on one (or few) security-managing entity (eg. key server) that is assumed to be trusted by all other entities in the entire Internet. Furthermore, the protocols often require host members to communicate securely with the entity (unicast). Not only does the entity (or entities) become a bottleneck in the scheme, it also becomes the "best" point of attack by intruders since it necessarily holds security parameters pertaining to the host members. Also, impacting scalability of protocols is the method used to perform a re-keying of the group key due a host member leaving the group or a new one joining. Re-keying of the group key involves a new group key being delivered to the (affected) members of the group.

Group authentication and sender authentication: In schemes in which a group key is used to encrypt the group traffic to afford membership control the decipherability of a multicast packet implies its origination from one of the members of the multicast group. That is, group authentication is achieved at the same time as data confidentiality. This level of authentication, although sufficient for some multicast applications, may not be enough for other applications in which the precise identity of the sender of the multicast packet needs to be known by the receivers of the packet. That is, sender authentication must be provided in addition to group authentication.

One simple approach to sender authentication within a multicast group would be for each member of the group to digitally sign the messages it sends, before the message is enciphered using the group key. This

approach requires the use of public key cryptography and depending on the multicast application, it may also require the existence of a public key infrastructure for its scalability.

Access control: The IP multicast model allows for any host to become a member of the group simply by requesting to join the group (Yang *et al*., 2001). Other group members may not necessarily be aware of the existence of other members in the group.

Although, the IP multicast model may be attractive in its native form to some applications, from the perspective of security such unlimited membership may be undesirable. The current framework views access control policies and their implementation to be an issue tightly related to the multicast application type. Similar to the independence (decoupling) of the group key management protocol from the underlying multicast routing protocol, the current framework proposes the independence of the group key management protocol from the access control model and implementation. This does not, however, preclude the possible developments (or extensions) of multicast routing protocols that exhibit some form of (limited) access control.

Membership verification: Membership verification refers to the ability of a member of a multicast group to request information and self-verify the constituents of the group. Although, this functionality may not be necessary (or even undesirable) in certain multicast applications (eg. pay per view transmissions), it may be highly desirable for other applications (eg. conferencing).

Solutions to the membership verification issue have been suggested in the context of cryptographic conference key distribution schemes, in which membership verification is in-built into the scheme itself or is a major feature of the scheme (Wallner *et al*., 1999; Yang *et al*., 2001).

BOOLEAN LOGIC MINIMIZATION TECHNIQUE FOR KEY MANAGEMENT

In this scheme, the creation of UID for the group members by the controller is not done through the usual binary notation. The UID is created by the technique suggested by Huffman for communication theory. The approach uses Petrick's method for Boolean logic minimization instead of the typical QuineMcLuskey's method. Petrick's method has a more systematic way of solving the Prime implicants and does not make mere guesses like the Quine Mcluskey scheme.

Modified huffman technique: Steps involved in generation of UID (User ID):

- The centralized authentication agent calculates the expiry of user subscription.
- The duration of stay has to be mapped in such a way it can be plugged as an input to the modified Huffman's technique instead of probability of transmission.
- Use Modified Huffman's technique to get the ID for each of the Users.
- The User IDs (UIDs) generated by this scheme is related to the keys possessed by them.

This method of UID generation makes sure that the user with expiry of subscription in the near future has the least number of bits in its ID. Therefore, the number of keys to be changed when that user leaves the group is going to be less. However, because of variable length coding there are going to be users whose UIDs have more bits than what they would have had if their UIDs were represented by normal binary notation. By reconstructing these UIDs again by this approach after some specific time period this can be resolved. This is because these users subscription expiration is now closer to the expiration deadline then previously.

Generation of UID: If the duration of stay of the users calculated as 5, 10, 20, 25 time units then the UIDs can be generated as follows. The time units are mapped to an input, equivalent to the probability of transmission. Each user's validity period is divided by the sum of duration of stay of all the users. This gives the values 5/60, 10/60, 20/60, 25/60. The inputs for generation of UID for each of the users are assigned in such a way that the user with least time units of validity has the highest value and henceforth. That is: User 1-25/60; User 2-20/60; User 3-10/60; User 4-5/60. Constructing Huffman codes is shown above. The codes will therefore be: User 1- 0; User 2-11; User 3 - 101; User 4-100. By having less number of bits for the user leaving most recently we can reduce the number of keys to be changed after the leave (Fig. 1).

User leaving the group: If a user leaves the group, the Group key and the auxiliary keys possessed by the leaving user have to be changed for preserving group security (Goshi and Ladner, 2003). In the Boolean logic minimization technique changing the group key is easy because the centralized agent simply has to encrypt with the compliments of the UID of the leaving user (Lorenz and Orda, 2002). But in the modified Huffman approach this cannot be applied. If the expected user

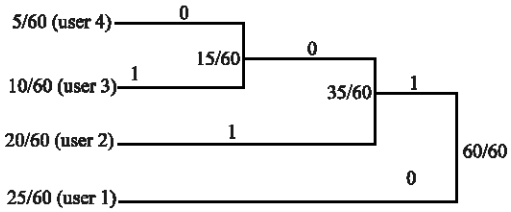

Fig. 1: Example of User ID generation

leaves the group, that is the user leaves according to the expiry of the subscription, then the centralized agent has to encrypt the new session key with k_{i+1}, $k_{i+1'}$ first, where 'i' is the most significant bit number of the user. In other words, if the leaving user has a 3 bit UID then first step of encryption is with k_3 and $k_{3'}$. This encryption makes sure that every user with one or more bits more than the leaving user will get the new group key. The immediately leaving user will have the smallest UID size. Then for the users with same number of bits, the compliments of each of the bits of the leaving user starting from the least significant bit is encrypted and it is repeated with next higher order bits until all the users of the same length UID gets the Group Key.

Algorithm:

1. Encrypt the new group key with k_{i+1}, $k_{i+1'}$ ('i' length of leaving user's UID)
2. If there are some users with same length UID in the group

   j = 0;

   do

   encrypt with j^{th} bit's compliment of leaving user's UID

   j = j+1;

   while(all the users of same length UID got the group key);

   else

   Stop the encryption.

KEY MANAGEMENT USING ONE-WAY FUNCTION TREES

Balenson *et al.* (2002) proposed a scheme based on the OFT approach. In this approach each group is associated with a one-way function tree, which is also a binary tree and is maintained by the group controller, or controller for short. Within the one-way function tree, each node x is associated with 2 keys, a node key k_x and its corresponding blinded node key $k'_x = g(k_x)$, where g is a one-way function. It is computationally infeasible that the adversary who knows k'_x can compute k_x Each internal node, say x, of the one-way function tree has exactly 2 children and its node key is defined by the following rule: $k_x = f(g(k_{left(x)})), g(k_{right(x)})$, where left(x) and right(x) denote the left and right child of x, respectively. Notation f denotes a mixing function, e.g., the bitwise XOR operation. The controller is responsible for managing the group key, which is the node key of the root and all the other keys associated with this one-way function tree. The group key is used for securing multicast communications while all the other keys associated with the one-way function tree are used only to facilitate efficient key updating when members are added to or evicted from the group and are also called auxiliary keys. Each member is associated with a leaf node of the one-way function tree and is securely given the node keys in the path from this leaf node to the root and the blinded node keys that are siblings to its path to the root.

The controller can securely communicate with each member by using a symmetric encryption function E with the member's node key.

If the node key associated with a leaf node changes, the node keys in the path from this leaf to the root will change, too. Thus, the blinded node keys associated with these changed node keys will also change. To maintain security, new values for these changed blinded node keys must be securely communicated to the appropriate subset of members. Let s be the sibling of x. The new value of k'_x must be securely communicated to the members who store it, i.e., the members who are associated with the descendants of s. Since the node key k_s is known to these members only, the controller can use E to encrypt the new value of k'_x with k_s and then broadcast the result.

Member addition: When a new member joins the group, an existing leaf node x that is closest to the root is split, creating new nodes left(x) and right(x). The member associated with x becomes associated with left(x) while the new member is associated with right(x) and is given a new node key. Then, the new values of the changed blinded node keys are securely broadcast to the appropriate subgroups according to the blinded node key updating operation described previously. An example of adding a member is shown in Fig. 2.

Removing a member: When the member associated with leaf node y is evicted from the group, node y's sibling, denoted by s, is reassigned to the parent of y. If s is a leaf node, its associated member is given a new node key. Otherwise, i.e., s is the root of a sub-tree, one leaf node of

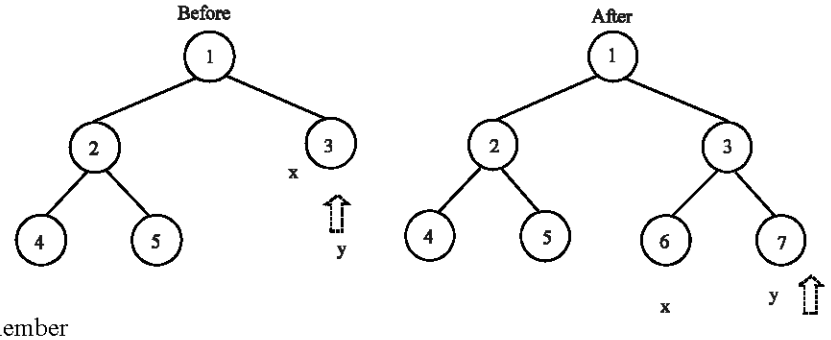

Fig. 2: Adding a member

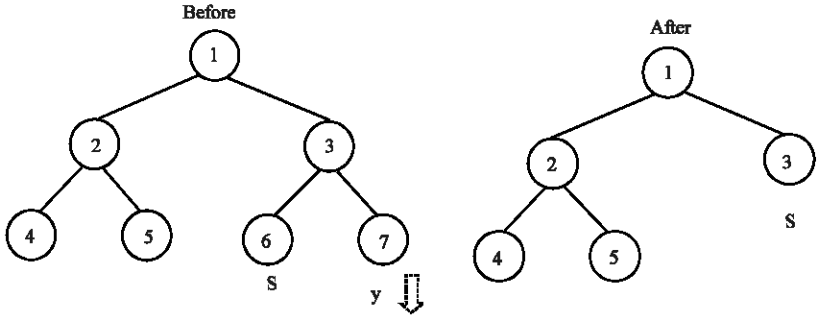

Fig. 3: Removing a member

this sub-tree is given a new key. Then, the new values of the changed blinded node keys are securely broadcast to the appropriate subgroups according to the blinded node key updating operation described previously. An example of evicting a member is shown in Fig. 3.

THE KRONOS APPROACH FOR SCALABLE GROUP RE-KEYING

Harney and Muckenhirn (1997), Harney and Harder (1999) and Mittra (1997) have proposed frameworks for distributing the task of key management for secure multicast groups. Under the Intra-domain Group Key Management Protocol (IGKMP), an administrative domain is divided into several "areas". Each host-member of a multicast group is assumed to reside in a particular area. IGKMP distinguishes between multicast groups for the purpose of key management and payload delivery. There is a Domain-wide Key Distributor (DKD) and an Area Key Distributor (AKD) corresponding to each area. Each host-member in a specific area is a member of a multicast group established for the purpose of key distribution that includes its AKD. All the AKDs and the DKD are members of another multicast group that is used by the DKD for transmitting the multicast data encryption key to the AKD, which in turn, transmits it to each host-member in its area. Note that there is a single group-wide multicast data encryption key under this scheme (Tsutomu, 1991; Sherman and McGrew, 2003).

Before a host can start participating in a group session, it has to join the group. Joining a group involves sending a request to the AKD (Setia *et al*., 2000). The AKD authenticates the request and checks the credentials of the member. If the member is allowed to join the group, the AKD will establish a private key that is shared with the member. This key will be used to encrypt the group wide data encryption key when it is sent to the member. Note that if an approach such as LKH is used at the sub-group or area level, a set of keys will need to be transmitted to the member. A member is considered to have joined the group only when it has received the group data encryption key.

We now describe a scalable approach, which we call Kronos, for re-keying a large and dynamic group that can be used within a distributed framework such as IGKMP. Under Kronos, group re-keys are not driven by member join or leave requests. Instead, at periodic intervals, all the member join and leave requests that have accumulated at an AKD are processed and the new multicast traffic encryption key is securely transmitted to the existing members of the group. An algorithm such as LKH can be used by each AKD to accomplish this task in a scalable manner. Note that most of the processing required for joins and leaves can be done during the time interval between re-keys. Further note that under this approach a

Table 1: Comparison of key management approaches

Metric	Modified Huffman technique	One-way function trees	Kronos approach
Re-keying method	Based on each membership change	Based on each membership change	Periodic
Frequency of re-keying	Dependent of group size	Dependent of group size	Independent of group size
Synchronization problem	Not Present	Not Present	Present
Key Management overhead	Low	High	High
Length of re-keying messages	Dependent on the number of users in the group	Dependent on the number of users in the group	Dictated by the security requirements of the application

new traffic encryption key will be transmitted by an AKD to the members in its area even if there has been no membership change during the previous time period (Table 1).

Two issues need to be addressed for this approach to work correctly. First, all the AKDs must use the same period for re-keying and must have their clocks synchronized so that they re-key at the same time. Second, the AKDs must share some state information that enables them to generate the same key without any communication. Further, no entity other than the AKDs should be able to generate the group key.

The first issue is addressed by having the AKDs agree in advance on the re-keying period and by using a clock synchronization algorithm such as the Network Time Protocol (NTP) (Setia *et al*., 2000). The second issue can be addressed as follows. First, all the AKDs need to agree on two shared secrets, say K and R_0. This can be accomplished by having the DKD (or the group coordinator) selecting K and R_0 and transmitting it to the AKDs using a secure channel. Alternatively, the AKDs can use a group key agreement algorithm such as Cliques to generate K and R_0 in a contributory fashion. Once the shared secrets are established, every AKD generates the multicast group key, R_1, by applying a secret-key encryption algorithm, E, to R_0 using K as the secret key. Thus, $R_1 = E_K(R_0)$. R_1 is then securely transmitted to the members of the group in the AKD's area. This process is repeated at each iteration. Thus

$$R_{i+1} = E_K(R_i),\ i \geq 0$$

The choice of the encryption function, E and the length of the key, K, is dictated by the security requirements of the application. Any function such as DES, triple DES, or IDEA (Setia *et al*., 2006) can be used. In addition, periodically, for enhanced security the AKDs should re-establish the shared secrets, K and R_0.

RESULTS AND DISCUSSION

The study of three group key management methods reveal that each scheme has its own advantages and disadvantages. The selection of a particular scheme for an application depends on factors such as group size, re-keying method and the nature of the application.

CONCLUSION

In this study, we outline the various security and protection issues in multicast content distribution. The issue of unauthorized hosts sending data to the multicast group leads to a number of vulnerabilities, as we have shown. However, there have been few solutions proposed for this problem. Thus, this remains an area for further research. We explain the issues and vulnerabilities that exist and discuss the research that has been done to provide solutions. We also briefly discuss some related areas. In this study, we have described Boolean logic minimization approach for key management. This uses Modified Huffman Technique for key generation. This method of UID generation makes sure that the user with expiry of subscription in the near future has the least number of bits in its ID. Therefore, the number of keys to be changed when that user leaves the group is going to be less. The Key management using one-way function trees is an improvement over BMS Scheme. This ensures that the evictee and the new member can not collude to get the group key that they should not know without incurring too much additional computational overhead. We also showed that Kronos can be used in conjunction with a distributed framework for key management such as IGKMP that uses a single group-wide session key for encrypting communications between members of the group.

REFERENCES

Balenson, D., D. McGrew and A. Sherman, 2002. Key management for large dynamic groups: One-way function trees and amortized initialization. Internet draft, IETF.

Ballardie, T. and J. Crowcroft, 1995. Multicast-specific security threats and counter-measures. In Proceedings of the Symposium on Network and Distributed Systems Security.

Canetti, R., J. Garay, G. Itkis, D. Micciancio, M. Naor and B. Pinkas, 1999. Multicast Security: A Taxonomy and Efficient Authentication. Proceeding of IEEE Infocom.

Debby M. Wallner, Eric J. Harder and Ryan C. Agee, 1997. Key Management for Multicast: Issues and Architectures. Informational RFC, draft-wallnerkey-arch-OO.trt.

Goshi, J. and R.E. Ladner, 2003. Algorithms for Dynamic Multicast Key Distribution Trees. Proceeding of ACM Symp. Principles of Distributed Computing (PODC).

Hamy, H. and C. Muckenhim, 1997. Group Key Management Protocol (GKMP) specification. RFC 2093.

Harney, H. and C. Muckenhirn, 1997. Group Key Management Protocol (GKMP) Architecture. RFC 2094.

Harney, H. and E. Harder, 1999. Logical Key Hierarchy Protocol Internet Draft. draft- harney-sparta-lkhp-sec-00.txt.

Lorenz, D.H. and A. Orda, 2002. Optimal partition of QoS requirements on unicast paths and multicast trees. IEEE/ACM Trans. Network, 10: 102-114.

Mittra, S., 1991. Iolus: A framework Scalable Secure Multicast. Proceedings of ACM SIGCOMM'97, Cannes, France, 1997. Tsutomu Sasao, "Bounds on the Average Number of Products in the Minimum Sum- of-Products Expressions for Multiple-valued Input Two-Valued Output Functions. IEEE. Trans. Comput., 40: 64-51.

Moyer, M.J., J.R. Rao and P. Rohatgi, 1999. A Survey of Security Issues in Multicast communications. IEEE. Network Mag., pp: 12-22.

Ran, C. and B. Pinkas, 1998. A Taxonomy of Multicast Security Issues. Intemet Drafi.

Setia, S., S. Koussih and S. Jajodia, 2000. Kronos: A Scalable Group Re-Keying Approach for Secure Multicast. In Proceeding of IEEE Symposium on Security and Privaq, Oakland, CA.

Sherman, A.T. and D.A. McGrew, 2003. Key Establishment in Large Dynamic Groups Using One-Way Function Trees. IEEE. Trans. Software Eng., 29: 444-458.

Wallner, D., E. Harder and R. Agee, 1999. Key management for multicast: Issues and architectures. IETF, RFC2627.

Yang, Y.R., X.S. Li, X.B. Zhang and S.S. Lam, Reliable group rekeying: A performance analysis. In Proceeding of the 2001 conference on applications, technologies, architectures and protocols for computer communications. ACM Press, pp: 27-38.

Yang, Y.R., X.S. Li, X.R. Zhang and S.S. Lam, 2001. Reliable Group Rekeying: A Performance Analysis. Proceeding of ACM SIGCOMM.