A New MDS Code Based Re-Keying for Efficient Multi Cast Key Distribution

Nagabhushanam Vennela [#1]

[#1]M.Tech Scholar,
Department of Computer Science & Engineering,
Maharaj Vijayaram Gajapathi Raj College of Engineering,
Vijayram Nagar, Chintalavalasa, Vizinagaram, AP, India.

Abstract

Multicasting is one of the best means of distributing required data in terms of resources usage. For any multicast group communication, we found that group key agreement was challenging issue because of its varying nature in dynamic mode. Along with group key management, we also found that efficient key distribution is an added problem for secure group communications. In this paper, we mainly studied on a new multicast key distribution scheme by gradually reducing the computation complexity. We also know there was many encryption algorithms proposed till to date, but instead of using such an existing conventional encryption algorithms, the proposed scheme employs a new technique called as MDS codes, a new class of error control codes, to dynamically distribute the multicast keys. This new MDS scheme drastically reduced the computation overhead load of each group member compared to various existing schemes which employ traditional encryption algorithms. As this new scheme can be easily combined with any real time key-tree-based schemes, this proposed scheme provides much more balanced communication complexity and storage complexity together for a very secure multicast key distribution in dynamic nature.

Keywords

Multicast, MDS Codes, Rekeying, Erasure Decoding, Key Distribution.

1. Introduction

Now a day's, multicasting is a new efficient means in various real time applications for distributing data in terms of resource usage like CPU, Memory, IO storage and so on. For this multicast communication, to maintain privacy for data we use symmetric encryption technique. All the designated participants include both receivers and members in a multicast group share a common session key which is used for encryption. When we go with many applications, however, the multicast group membership keep on changes dynamically, i.e., when some new members are given authorization to join in a new multicast session while some already existed old members should be excluded from that group. Thus in both the reasons, session keys shall change dynamically to ensure both ***forward secrecy method*** *and* ***backward secrecy method*** *of* total multicast sessions. The major differences between the two secrecies are as follows: the forward secrecy method is maintained if an old member who has been removed from the current session in that group can't able to *access* the communication of the current running session, and the backward secrecy method is guaranteed in such a way that, if a new member of the current session can't recover the communication which were done in past sessions. This method requires for each session we need a new key that is only known to the current session members, i.e., session keys need to be dynamically distributed to authorized session members who are currently available in the session.

In this paper, we mainly study how a multicast group key can efficiently be distributed in computation. We used a very new model in this proposed paper, where for each and every group member the session keys are generated and later they were distributed based on their issued order to all the members which was done by central group controller (GC), as it has very less communication complexity as compared with several distributed key exchange protocols, which is a very desired property in most of the wireless communication applications [1], [2], [3], [4], [5]. The GC will distribute following resources to group members including communication, storage, and computation resources. In the proposed model the communication overhead complexity is usually measured by the number of data bits that are required to transmit from group controller to various group members that are present in that group, whereas the storage complexity is mainly measured by the number of data bits that the group controller and group members need to store to obtain session keys. Hereafter, in this paper we discuss the problem of how resources can effectively be used to distribute session keys within the group for various group members is referred to as the group key distribution problem.

For a multicast group key distribution with a very large number of members, KTREE-based schemes (also called as Key Tree) were introduced to decompose a very large group into number of individual multiple layers of subgroups with smaller size [3], [4], [5]. Using these various schemes, a group membership key change can be securely and effectively handled for change of their session keys in the corresponding subgroups present in that group without affecting other users present in that group. Thus, with this scheme the communication complexity cost is gradually reduced. For a communication group containing of n members, KTREE based schemes have a communication complexity maximum of O(logn) and a storage complexity maximum of O(n) for the group controller and also O(logn) for each group member. It has been clearly shown that if a group member can store at most O(logn) keys, then the lower bound of the communication complexity is assumed to be of O(logn) if and only if a structure preserving protocol for the group controller is used for group key distribution [6]. Thus, the KTREE based schemes are of practical interest for a variety of applications because of its balance between communication complexity and storage complexity.

In this paper, we finally propose a new dynamically based group key distribution scheme that severely reduces the overall computation complexity cost and yet maintains at least the same degree of security by using symmetric encryption algorithms without increasing more communication cost or storage complexity cost. In our proposed scheme, the information that is having close relation with session keys is initially encoded using error control codes rather than conventional encryption mechanisms. In general, the two methods like encoding and decoding of a proper error control code have much lower computation complexity than compared with various conventional encryption and decryption algorithms, which has been verified by our experiments conducted in this paper. Thus, the computation complexity of key distribution can be significantly reduced. The same similar idea of using error control codes to achieve privacy was also employed in other papers like [7], [8], and [9]. The major difference between already available schemes and our proposed scheme in this paper is that our scheme allows dynamic group membership changes with a very low storage complexity, whereas the existing schemes only work for a predefined static group membership. Several experiments are conducted to show great reduction of our scheme in computation complexity than using other commonly used traditional encryption algorithms on 3-ary balanced key trees.

2. Background Work

In this section, we mainly discuss about various key distribution techniques that were used in existing and also represent the key distribution architecture diagram in this section.

2.1 Key Distribution

In this we briefly describes key distribution method in group key distribution mechanism, where all users in the group share a common group key denoted as (k_g). The group key is mainly used to encrypt the data which is

transmitted to different users present in that group. Whenever any user is revoked from the current group, in order to protect the privacy of the remaining users present in current group, the group controller (GC) needs to change the current key and generate new group key for the set of current active users and distribute the new group key to the remaining users who are in active in that group. To simplify the distribution of the new group key, each user maintains additional keys (e.g. user oriented, key oriented, group oriented), which are shared with a subset of users present in the current group.

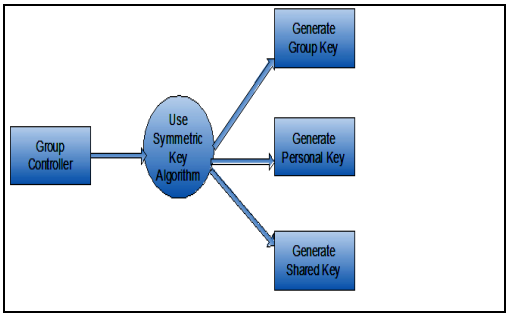

Fig.1. Key Distribution

To send the generated new group key (k_g) to the set of remaining users, the group controller (GC) initially encrypts the group key using the shared keys which is not known to the previously revoked users. To reflect current group membership, the group controller also needs to change the old shared keys of all other users and it generates and distributes the shared keys that are not known to the revoked users. There are mainly two approaches available with the group controller for distributing the new shared keys. In the very first approach, the group controller explicitly transmits the new shared keys to the current users automatically without intervention of any user. In our current work, we adopt the second approach where the group controller and the users update the shared keys using the following technique:

$$k_g = f(k_g, k_x)$$

Where k_x is the old shared key, k_g is the new shared key, f is a one-way function. Using this technique, only those current users who knew the old shared key k_x will be able to get the new shared key k_g.

3. Proposed Key Distribution

In this section we will discuss about the proposed K-TREE based new key distribution technique along with MDS based rekeying on a key tree.

3.1 K-TREE Based Rekeying Scheme

Inorder to reduce the communication complexity overhead of rekeying operations, a K-TREE based scheme and many of its variations have been proposed [3], [4], [5], [10]. This proposed scheme mainly used for reducing the communication complexity of rekeying operations to O(logn), whereas each member in that current group needs to store O(logn) keys, and the Group Controller needs to store at least O(nlogn) keys, where n is the multicast group size. This is the most practical proposed key distribution scheme, which balances both the communication overhead and storage complexity overhead for dynamic multicast key distribution. Here, we briefly describe a basic K-TREE based scheme for the rekeying operation.

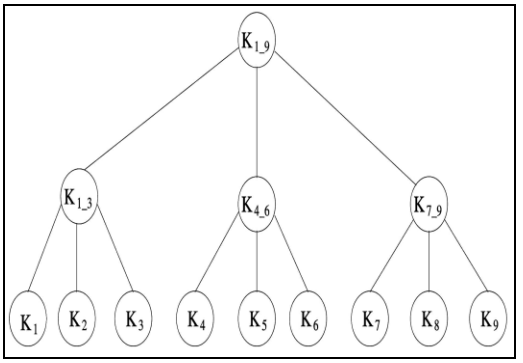

Fig. 2. A Key Tree for a Nine-Member Group

During the rekeying process, the Group Controller can thus able to multicast securely to a subgroup of members using their shared subgroup key instead of individual member keys.

Fig. 2 clearly shows a key tree for a nine-member group. K (1<= i <= 9) is the individual key of member i. K_{1-9} is known to be the group session key that is shared by all the members present in that tree. Finally, K_{1-3}, K_{4-6}, and K_{7-9} are three subgroup keys for the three corresponding subgroups respectively. For example, K_{1-3} is shared by members 1 through 3, who form the first subgroup and likewise all the other groups follow the same principle.

3.2 MDS Code-Based Rekeying on a KTREE

As we discussed in other KTREE-based rekeying schemes, MDS codes are also used to rekey from bottom (i.e Leaves) to up (Root). In Fig. 2, when member 9 leaves from that tree at any instance, the new subgroup key K_{7-8} is rekeyed before the GC server changes the new group session key to K_{1-8}. When MDS codes are used for the rekeying process instead of existing key generation servers, each node (leaf or intermediate) key becomes a pair of (j_i, s_i), as discussed in the previous section. The Group Controller stores all the key pairs on the KTREE. Whenever cryptography encryption techniques are needed for rekeying a subgroup key, a new MDS code word is constructed from all the key pairs (j_i, s_i), of the corresponding immediate child nodes and then multicast by the GC. Note that in the process of rekeying, each level of the KTREE may use the same or different MDS codes. However, for the simplicity of our paper implementation, the same MDS code can be used for all levels in a tree, since the security of the basic scheme does not depend on the MDS code.

3.3 Comparison with Traditional Cryptographic Schemes

To evaluate the proposed KTREE scheme, a multicast key distribution scheme is implemented to disseminate 128-bit session keys among a 3-ary balanced key tree. The proposed new scheme is compared with several traditional cryptographic schemes. As the communication and storage complexity are the same among all the schemes, it suffices to simply compare the computation complexity.

The comparison considers the following scenario, where each three-member group within the 9 arc tree has one member that departs. These departures at each and every sub group are not constrained to happen at the same time, but in our practical observation, they might tend to be close, for example, at the end of one movie broadcast, etc. This makes a batch process possible, which means that all remaining members could be rekeyed at once.

TABLE 1
Computation Time Comparing to the RC4 Approach (Multicast Group Size of 59,049)

time (us)	RS	RS(MD5)	AES	IDEA	CAST-128	RC4
GC	19	28	81	74	99	227
member	2	5	23	82	23	61

The computation time of the proposed key distribution is also compared to conventional stream ciphers, as shown in Table 1, for a selected multicast group size of 53,049. Notice that the computation times of both the Group Controller and the member using the RC4 cipher are significantly larger than using several other schemes. Even though RC4 itself is a fast stream cipher, its key scheduling process has dominant effect in this particular scenario, where only 128-bit data is encrypted/ decrypted using any given key. Results under other multicast group sizes are similar, which are thus not duplicated here.

4. Implementation Modules

This proposed scheme is mainly divided into four modules based on our proposed life cycle. In this section we will discuss about the following modules in a detail.

1. Key Generation
2. Message Transmission
3. Cryptography
4. Authentication

1. Key Generation

This module is mainly used for generating the session keys as well as the secured keys used by the group members to communicate with the GC (group controller). The private keys for privacy are generated using MDS method. The GC (Group Controller) sends number of group members to the KGC (Key Generation Center). The keys are generated by the KGC and submitted to the Group Controller. In session key generation, initially sixteen decimal digits are generated by using random number generation method .Then each decimal digit is split and compared with pre determined binary format.

2. Message Transmission

This module is mainly used for performing the multicasting using the proposed technique and algorithm which was used in this paper. This message transmission module will try to transmit the data from one node to other node within the same group or other group within the tree.

3. Cryptography

This module is mainly used for securely transmitting all user defined messages and also the new session keys whenever group membership changes within the tree. This module mainly helps in achieving high privacy for the user data.

4. Authentication

This module is mainly used for performing authentication of all the group members. Each member must contact Group Controller initially to get registered with the group and must obtain a private key to communicate with Group Controller. Without getting any private key from GC the members can't able to communicate with each other with in the tree.

5. Conclusion

In this paper, we have presented a very new dynamic multicast key distribution scheme using MDS codes. The computation complexity of key distribution method is greatly reduced by employing only erasure decoding of MDS codes instead of more expensive conventional encryption and decryption computations. Easily combined with KTREE's or other rekeying protocols that need encryption and decryption operations, this scheme provides much lower computation complexity while maintaining low and balanced communication complexity and storage complexity for dynamic group key distribution. By conducting several experiments, we know that this scheme is thus practical for many applications in various broadcast capable networks such as Internet and wireless networks.

6. References

[1] D.R. Stinson, "On Some Methods for Unconditionally Secure Key Distribution and Broadcast Encryption," Designs, Codes and Cryptography, vol. 12, pp. 215-243, 1997.

[2] D.R. Stinson and T. van Trung, "Some New Results on Key Distribution Patterns and Broadcast Encryption," Designs, Codes and Cryptography, vol. 14, pp. 261-279, 1998.

[3] M. Waldvogel, G. Caronni, D. Sun, N. Weiler, and B. Plattner, "The VersaKey Framework: Versatile Group Key Management," IEEE J. Selected Areas in Comm., vol. 7, no. 8, pp. 1614-1631, Aug. 1999.

[4] D.M. Wallner, E.J. Harder, and R.C. Agee, "Key Management for Multicast: Issues and Architectures," IETF Internet draft, Sept.1998.

[5] C.K. Wong, M. Gouda, and S.S. Lam, "Secure Group Communications Using Key Graphs," Proc. ACM SIGCOMM '98, Sept. 1998.

[6] R. Canetti, T. Malkin, and K. Nissim, "Efficient Communication-Storage Tradeoffs for Multicast Encryption," Advances in Cryptology—Proc. Int'l Conf. Theory and Application of Cryptographic Techniques (EUROCRYPT '99), May 1999.

[7] A. Shamir, "How to Share a Secret," Comm. ACM, vol. 24, no. 11, pp. 612-613, Nov. 1979.

[8] R.J. McEliece and D.V. Sarwate, "On Sharing Secrets and Reed- Solomon Codes," Comm. ACM, vol. 26, no. 9, pp. 583-584, Sept.1981.

[9] R. Blom, "An Optimal Class of Symmetric Key Generation Systems," Advances in Cryptology—Proc. Workshop Theory and Application of Cryptographic Techniques (EUROCRYPT '84), pp. 335-338, 1984.

[10] J. Snoeyink, S. Suri, and G. Varghese, "A Lower Bound for Multicast Key Distribution," Proc. IEEE INFOCOM '01, Apr. 2001.

7. About the Authors

Nagabhushanam Vennela is a student of the Department of Computer Science & Engineering of Maharaj Vijayaram Gajapathi Raj College of Engineering, Chintalavalasa, Vizinagaram. Presently he is pursuing his M.Tech from this college. He completed his MCA Degree from Andhra University. His area of interest includes Network Security.