Key Management Scheme for Multi-Layer Secure Group Communication

R. Aparna
Dept. of Computer Science and Engg.
Siddaganga Institute of Technology,
Tumkur, Karnataka, India.
raparna@sit.ac.in

B.B. Amberker
Dept. of Computer Science and Engg.
National Institute of Technology,
Warangal, Andhra Pradesh, India.
bba@nitw.ac.in

***Abstract*—Many emerging applications are based on group communication model and many group communications like multimedia distribution and military applications require a security infrastructure that provides multiple levels of access control for group members in which group members are divided into a number of subgroups and placed at different privilege levels based on certain criteria and a member at higher level must be capable of accessing communication in its own level as well as its descendant lower levels and converse is not true. Two key management schemes have been developed to provide hierarchical access control: first method is key-based and the second one is share-based. Constructing a hierarchical group communication model based on secret shares is more flexible rather than using keys. In this paper we use share-based key management scheme and propose to reduce the storage at higher level users as compared to the scheme proposed by Dexter et al. by reusing a part of the pre-positioned information as common for all the layers in the system. We compare the storage and encryption cost of our scheme with the scheme proposed by Dexter et al. We also address periodic group rekeying.**

Keywords: Group Communication, Group Key, Multi-layer, Secret Sharing, Pre-positioned information, Activating share

I. Introduction

Many emerging applications like secure audio and visual broadcasts, pay-per-view, scientific discussion, teleconferencing are based on group communication model. Several users participate in these applications and multicast communication is an efficient means of distributing data to a large group of participants [7], [12], [18], since it reduces the demands on network and bandwidth resources. But, the communication among these participants must be carried out confidentially. Thus, a common key known as *group key* or *secret key* must be established with all the users in the group, so that any group member can encrypt the message using this key and all others can decrypt the message using the same key. The group being dynamic in nature, allows member join and leave events. Efficiently managing group key for large, dynamically changing groups is a difficult problem. Every time when a new member joins the group, the group key must be changed in order to provide *backward access control* (i.e., new members should not be able to access past communication). In the same manner, when a user leaves the group, the group key must be changed so that leaving member cannot have access to future communication that takes place between remaining group members, known as *forward access control*. This group key updating process is referred to as *rekeying*.

Rekeying process involves changing the group key whenever there is a membership change and distributing it among members of the group in a secure manner. To communicate changed key among group members securely, rekey messages are constructed, encrypted and multicast to the group. The overhead involved in rekey operation i.e., key updation, number of encryptions performed and communication cost must be minimum and should be independent of the group size, which improves scalability.

Several secure group key management techniques have been proposed to support scalable secure multicasting [15], [29], [16], [21]. In a typical multicast key management scheme, there is a trusted third party, known as Key Distribution Center (KDC). This single trusted centralized entity is responsible for generating and distributing keys securely to the group members. Among the schemes which involve KDC [3], [29], [17], [11], the scheme proposed by Wong et al. in [29] is most efficient and is widely used since it improves scalability. The scheme uses a hierarchical tree structure in which users are maintained at the leaf level and every user is assigned with keys along the path of its location till the root. Besides group key, the KDC shares auxiliary keys known as Key Encryption Keys (KEKs) and these are used solely for the purpose of updating the group key and other KEKs. In addition, every user shares a private key that is known by itself and the KDC.

Hierarchical tree structure is also used in Centralized Key Management with Secret Sharing (CKMSS) [9], [10]. In this scheme, KDC considers a t degree (t is a non-negative integer) polynomial with the constant term of the polynomial being the secret key. It computes t distinct shares of the polynomial and stores them at the users known as *pre-positioned information*. To compute the group key, $(t+1)$ shares of the polynomial are required and this $(t+1)^{th}$ share is sent as an *activating share* by the KDC. Once the group key is computed, it is used until a member joins or leaves the group. For every membership change (join/leave), to perform rekey operation, KDC multicasts an activating share to enable the members to compute new group key.

Both the key-based and share-based schemes discussed above are designed for managing keys for a single data stream and for the same category of users. These schemes are not efficient for many multimedia applications that distribute data in multi-layer or multi-object format [20]. Multi-layered or multi-object services comprise of users that subscribe to different objects or layers, or possibly multiple of them. For example, in an HDTV broadcasts, users with normal TV receivers can receive the normal format, while other users with HDTV receiver can receive both the normal format and the extra information needed to achieve HDTV resolution.

Another example can be a multicast scalable video service where the video is encoded into 3 layers: Base Layer (BL), Enhancement Layer 1 (EL1) and Enhancement Layer 2 (EL2). Here, the users can be classified into 3 different groups based on the quality of the movie they purchase. The users purchasing the movie at the basic quality level belong to BL group, users purchasing the movie at the medium-quality level belong to both BL and EL1 groups, whereas the users purchasing the movie at the best quality level belong to all the three i.e., BL, EL1 and EL2 groups. Thus, users with access to higher quality movie must also have access to lower quality ones.

One more application where we need layered approach is in military : Military troop contains different category of militants like Captains, Lieutenants, Sergeants, Corporals, Soldiers etc. and this requires a hierarchical group communication model. Captains are at the higher level, Lieutenants at the second higher level, Sergeants at the level below Lieutenants, Corporals at the next lower level and Soldiers must be at the lowest level as considered in [22]. Soldiers should be able to communicate only with other Soldiers (peer members), whereas Sergeants can communicate with other Sergeants as well as with Corporals and Soldiers, similarly Captains should have access to all the communications that take place between different classes.

To manage this type of scenario, a naive solution is to extend key-based and share-based tree structure, such as using independent trees for each layer which is inefficient and does not scale well when there are many objects or layers. Hence, we need to have a multi-group key management scheme that exploits the overlap in the memberships of different objects or layers. Two key management schemes have been developed to provide hierarchical access control: The scheme proposed in [25], [24] is key-based, where each layer has its own session key and whenever there is a membership change in any level, corresponding session key is changed and securely transmitted to appropriate group members. The scheme proposed in [5] is share-based in which the secret key is considered as the constant term of a t degree polynomial. This polynomial is evaluated at different points to compute distinct shares. Since the degree of the polynomial is t, at least $(t+1)$ distinct shares are required to reconstruct the polynomial using Lagrange's polynomial interpolation. Based on this principle, for each layer, a polynomial is considered and t distinct shares of this polynomial are stored at the members of that layer (pre-positioned information) and KDC sends $(t+1)^{th}$ share as an activating share so that members can compute group key for that layer by constructing the polynomial and evaluating it at 0. Whenever there is a membership change in any level, KDC just sends a different activating share to the members of that level so that they can compute a new group key for that level.

In this paper we consider the share-based scheme, but try to reduce the storage at the users of the higher level as compared to the storage required in [5]. Remaining part of the paper is organized as follows: Section 2 gives an overview of related work in secure group communication and multi-layer secure group communication. Section 3 describes the new scheme and handles different events, in Section 4 we compare the scheme with other schemes and Section 5 concludes the paper.

II. Related Work

Secure Group Communication: Group key management protocols can be mainly classified into two categories: centralized and contributory. The centralized key management schemes [3], [29], [11], [28], [27], [16] assume a trusted KDC which is responsible for generating and distributing keys to group members. Contributory key agreement schemes [6], [4], [13], [1], [2], [19], [14] require every group member to contribute an equal share to the common secret key. Secret key for secure group communication is computed as a function of all members contribution. Besides these two categories, a decentralized key management scheme proposed in [17] divides the group into several subgroups with a manager for each subgroup. The responsibility of key management is divided among subgroup managers.

The centralized key management scheme proposed by Wong et al. [29], [28] is one of the efficient method which employs a logical key structure to maintain and co-ordinate key generation. It addresses the problem of minimizing number of rekey messages during membership change (join / leave). In this approach KDC maintains a key tree with degree d, where each node on the tree corresponds to either member's private key, or group key or an auxiliary key. This scheme achieves scalable rekeying, which requires $2log_d n$ rekeying overhead for user join event and $(d-1)log_d n$ for user leave event, where n is the group size. The scheme proposed in [15] reduces the overhead during user join event by allowing the users to calculate new key on their own using one-way function. The scheme proposed by McGrew and Sherman in [16] uses a one-way function tree in which keys are generated using one-way functions. In both the schemes rekeying overhead is reduced to $(log_2 n)$ instead of $(2log_2 n)$.

In contributory key agreement scheme no trusted group server exists and this might occur in applications where there is no trust on a single entity, or there are no servers. In this approach all the users contribute their shares for group key computation. The scheme proposed in [14] uses Diffie-Hellman key exchange protocol [6] to compute auxiliary keys and group key. The schemes presented in [13], [8], [26] use logical tree structure and compute group key in $(log_2 n)$ number of rounds.

These key management schemes are designed for a group communication scenario in which all the group members have the same access privilege. Directly applying these schemes to applications which contain members with different access privileges leads to inefficient solutions.

Multi-layer Secure Group Communication: In [22], a military application is considered to illustrate multi-layer secure group communication. Military officers belonging to different categories (Captains, Lieutenants, Sergeants, Corporals, Soldiers etc.) are divided into subgroups and are hierarchically placed one above the other. Higher level officials can have access to the communication between its descendant lower level subgroups. To provide this feature, it uses a one-way hash function to compute a chain of keys. The main idea of using one-way hash function is to relate classes' keys in such a way that, knowing a key of its own class, a member can compute keys of lower classes. Thus, Captains are given with random key K, for the Lieutenants K is hashed once i.e., computed H(K) and sent, for Sergeants K is sent after hashing it twice i.e., H(H(K)), Corporals are assigned with a key K which is hashed thrice and Soldiers with K which is hashed four times. If Captains want to access the communication between Sergeants, they hash the key K twice and use it as the key for that level.

In [24], [25], Sun and Liu used tree based hierarchical approach to handle broadcasting multimedia applications in different sets of objects or layers to different groups of users. They consider a multicast scalable video service which is encoded into 3 layers: BL, EL1 and EL2 and defined following two groups: Data Group (DG)- a set of users who receive the same single data stream and Service Group (SG)- a set of users who receive the same set of layers. Video encoding allows three multicast streams to be assigned to three DGs. The users subscribing to highest quality level join all three DGs, the users subscribing to the moderate quality level join BL and EL1 DGs and the users subscribing to the basic quality level join only BL DG. The users subscribing to the same quality level belong to the same SG.

$D_1, D_2, \ldots, D_M$ denote DGs, where M is the total number of DGs and SGs are represented by S_t, where $t = [i_1^t, i_2^t, \ldots, i_M^t]^T$ is a binary vector having M elements with $i_j^t = 0$ or 1, $\prod_{j=1}^{M} i_j^t \neq 0$, and $S_t = \{D_1, i_1^t\} \cap \{D_2, i_2^t\} \cap \ldots \cap \{D_M, i_M^t\}$, where $\{D_j, 0\} = \overline{D_j}$ and $\{D_j, 1\} = D_j$. These notations are illustrated in Fig.1.

Key management in traditional centralized tree based key management scheme is illustrated in Fig. 2, in which a separate key tree is constructed for each DG. Although it is easy to implement using independent trees, a substantial overhead is introduced due to the overlapping membership in different DGs. When a user leaves the service, all the DGs to which it belongs to must change the relevant keys and if a user switches from one SG, say, S_{t1} to another SG, say, S_{t2}, all DGs which include users in S_{t1}, but not in S_{t2} need to perform rekeying.

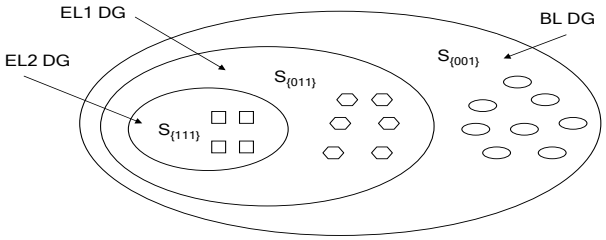
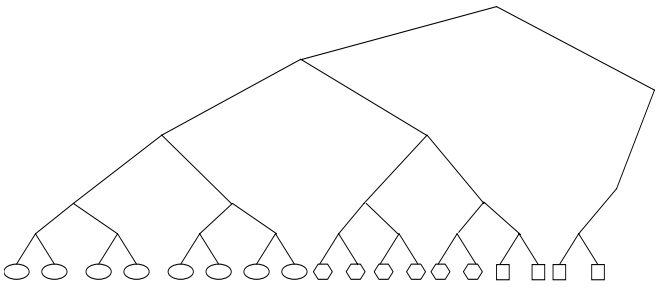


Fig.1 Data Group and service groups in a multi-layer multicast service


Tree for BL data group

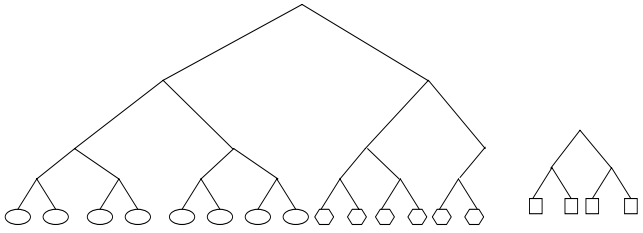


Fig. 2 Traditional independent tree key management for three layers

In order to manage the keys of all the subgroups, independent trees are integrated into one key graph in [24], [25] and is shown in Fig. 3. In [5], same key graph structure as illustrated in Fig. 3 is used and for key management, a secret sharing scheme using threshold cryptography is employed, i.e., in this scheme, instead of each node being assigned with a random key as in [24], [25], it is assigned with a set of shares (pre-positioned information). The assignment of unique shares to the nodes allows the generation of new node keys every time a different activating share is multicast.

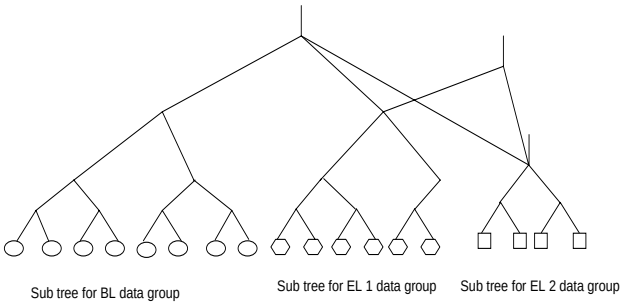


Fig. 3 Integrated graph key management scheme for three layers

III. New Scheme

We propose to use the key graph structure as in [24], [25] and we use Shamir's secret sharing scheme for key management at the group key level and random keys at the subgroup level i.e., auxiliary keys are randomly generated and assigned. In Shamir's (t, n) secret sharing scheme [23], the secret s is defined to be the coefficient a_0 of a t degree polynomial $P(x) = a_{t-1}x^{t-1} + a_{t-2}x^{t-2} + \ldots + a_1x + a_0$, where the coefficients $a_0,\ a_1,\ \ldots,\ a_{t-1}$ are the elements of a finite Galois Field $GF(p)$, and p is a large prime. The trusted dealer computes the shares s_i by evaluating the polynomial $P(x)$ at n distinct points and securely distributes s_i to user U_i, $i = 1, 2, \ldots, n$. The secret s can be recovered by reconstructing the polynomial $P(x) = \sum_{i=0}^{t-1} y_i \prod_{0 \le j \le t-1, j \ne i} (x - x_j)/(x_i - x_j)$ from any t of the n shares, and computing $P(0)$.

In our scheme, we try to reduce the storage at the higher level users as compared to the storage required in [5]. We number the levels from lower level to higher level as 1 to M. During initial group set up phase, KDC considers a t degree polynomial with the constant term being the secret key, computes t distinct shares of the polynomial and securely stores them as pre-positioned information at the members belonging to level 1 group. Since all the members in the system belong to this group, they all get these t shares. For the members to compute the secret key, polynomial must be reconstructed, which requires $(t+1)$ shares and this $(t+1)^{th}$ share is sent by the KDC as an activating share (AS). Thus, all the members can compute the group key for level 1.

The secret key for level 2 group is considered to be a constant term of $(t + 1)$ degree polynomial, whose t distinct evaluation points are same as the evaluation points of the polynomial considered for level 1. Hence, the same t shares that are supplied as pre-positioned information for level 1, act as pre-positioned information for level 2 group also, but, along with these t shares, one more share is provided. Thus, users belonging to level 2 group store $(t + 1)$ shares and are capable of computing group keys for level 1 and level 2 after receiving an activating share from KDC. To compute group key for level 1, they construct a t degree polynomial using first t pre-positioned information and activating share and to compute group key for level 2, they construct a $(t+1)$ degree polynomial using $(t + 1)$ pre-positioned information and activating share, evaluate both the polynomials at 0 to get the group keys.

In the same manner, for level 3 group, a $(t + 2)$ degree polynomial is considered whose $(t + 1)$ shares are same as the ones used for level 2 group and one more extra share is provided by KDC. Thus, users of level 2 group store $(t + 2)$ elements from $GF(p)$. Likewise, as we move up the hierarchy, degree of the polynomial considered is one more than the degree of the polynomial for previous level and users must store just one extra element from $GF(p)$. Thus, users belonging to level h, $h = 1$ to M, must store $(t + h)$ elements from $GF(p)$ and can compute secret keys of levels from 1 to h by using the same activating share. In the scheme proposed by Dexter et al. in [5], for different levels different sets of pre-positioned information are stored with the users. Thus, users belonging to higher level must store different sets of pre-positioned information of all lower levels, which increases storage at the users. In our scheme, we are reducing this storage cost. As we move up the hierarchy, degree of the polynomial is increased. Though the amount of computation required is more, it increases the resistance of the system to attacks, hence, the system is more secure.

A pre-positioned information is stored in the following format: $(i, P(i), l)$, where $i = 1\ to\ (t + j)$, $j = 0\ to\ (l - 1)$, $l = 1\ to\ M$ (number of levels) and the share $P(i)$ is the evaluation of polynomial $P(x)$ at i. Fig. 4 shows an example integrated key graph for three levels in which three independent groups are integrated to form a three level hierarchy. In figure, G_1, G_2, and G_3 represent the roots of independent groups and S_1, S_2 and S_3 are the roots of integrated groups, i.e., S_1 is the root for BL, S_2 for EL1 and S_3 for EL2.

We are addressing the following events after considering three levels:

1. When a new member joins the service.

2. When a member leaves the service or when a member moves from one service level to another service level.

A. Member Join Event

When a new member, U_{new}, joins any group i, $i = 1, \ldots, M$, keys along the path from the joining point till the root must be changed and conveyed to relevant users. Instead of KDC changing the keys or sending a new activating share, we allow the members of the group i themselves to compute the new group key and auxiliary keys on their own by applying one-way hash function to the corresponding previous keys. For

the new user, U_{new}, KDC unicasts pre-positioned information, auxiliary keys and group key after encrypting it with the private key of new user.

B. Member Leave Event

If a member leaves any group, keys along the path from leaving position till the root are to be changed. To change the group key, KDC generates and sends a new activating share securely and other keys along the path are changed and conveyed to relevant existing members. We illustrate this with the following example: from Fig. 4, if member M_8 leaves the group, the following messages are constructed and sent to users:
KDC $\rightarrow \{M_1$ to $M_4\} : [AS]_{K'_{1-8}}$, $[K'_{1-8}]_{K_{1-4}}$
KDC $\rightarrow \{M_5, M_6\} : [AS]_{K'_{1-8}}$, $[K'_{1-8}]_{K'_{5-8}}$, $[K'_{5-8}]_{K_{5-6}}$
KDC $\rightarrow M_7 : [AS]_{K'_{1-8}}$, $[K'_{1-8}]_{K'_{5-8}}$, $[K'_{5-8}]_{K'_{7-8}}$, $[K'_{7-8}]_{K_7}$
KDC $\rightarrow \{M_9$ to $M_{14}\} : [AS]_{K_{9-16}}$
KDC $\rightarrow \{M_{17}$ to $M_{19}\} : [AS]_{S_3}$

C. Member Moving from One Service Level to Another

If a member moves from one level to its higher level, say from level i to j $(i < j)$, then it must be provided with extra pre-positioned information meant for levels from $(i + 1)$ to j and the auxiliary keys in the leaving subgroup must be changed. If a member moves onto a descendant level, say from level j to i, $(i < j)$, then extra pre-positioned information meant for the levels from $(i + 1)$ to j must be changed and conveyed to the members of levels $(i + 1)$ to j. In level j subgroup, auxiliary keys possessed by moving member are to be changed.

For example, in Fig. 4, if a member M_{12} moves from level 2 to level 3, it is inserted as sibling of member M_{17}. In the pre-positioned information, the share $(t + 2, P_3(t + 2), 3)$ must be changed to provide backward access control. Following messages are constructed and sent to users:
KDC $\rightarrow \{M_{17}, M_{18}\} : [P'_3(t+2)]_{S_3}$, AS
KDC $\rightarrow M_{19} : [P'_3(t+2)]_{S_3}$, $[K'_{19-20}]_{K_{19}}$, AS
KDC $\rightarrow M_{12} : [P'_3(t+2)]_{K_{12}}$, $[K'_{19-20}]_{K_{12}}$, AS
KDC $\rightarrow \{M_9, M_{10}\} : [K'_{9-16}]_{K'_{9-12}}$, $[K'_{9-12}]_{K_{9-10}}$
KDC $\rightarrow M_{11} : [K'_{9-16}]_{K'_{9-12}}$, $[K'_{9-12}]_{K'_{11-12}}$, $[K'_{11-12}]_{k_{11}}$
KDC $\rightarrow \{M_{13}, M_{14}\}$: $[K'_{9-16}]_{K_{13-16}}$

D. Periodic Rekeying

If the content has very high value, even though there is no membership change, group key must be changed for all the levels which leaves the attacker with very less time to attack on the current key values. To achieve this periodic rekeying, we propose to send an activating share after every rekey interval, so that all the users can compute new group keys for all the levels to which they belong.

Instead of multicasting the activating share, other efficient method which we propose is to allow the users to compute new group key after every rekey interval just by applying a one-way hash function on the current group keys. This reduces the computation at the users, since it avoids reconstruction of the polynomial.

TABLE I
STORAGE AND ENCRYPTION COST

	Dexter et al. Scheme	**New scheme**
Storage at each user	th	$t + h + M$
Number of elements Encrypted during Leave	$t(h + M)$	$h + Mt$

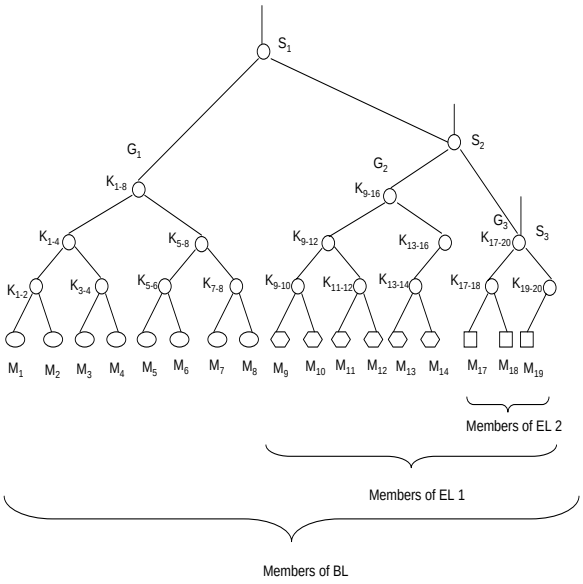


Fig. 4 An example Integrated key graph with auxiliary keys and group keys

IV. COMPARISON

Table 1 gives the comparison of our scheme with the scheme proposed by Dexter et al. in terms of storage and encryption cost. In Dexter et al. scheme [5], each user must store h sets of pre-positioned information (h is the height), where each pre-positioned information contains t shares. In our scheme, each user stores $h - 1$ number of keys for subgroups, and the size of the pre-positioned information varies from t (for level 1 group members) to $(t + M)$ (for level M, the highest level group members). In Dexter et al. scheme, if a member leaves the group, in order to change the keys along the path till the root, corresponding pre-positioned information must be changed, which leads to $t*h$ encryptions. Also, pre-positioned information meant for different levels must be changed, i.e., $t * M$ encryptions. Hence, number of elements encrypted is $(ht + lt)$. Whereas in our scheme, keys along the path and activating share are encrypted, thus, it is just $(h + Mt)$.

V. CONCLUSION

Managing multiple groups with overlapped membership is one of the important issue in group communication scenario. We proposed a scheme for such hierarchical group key management using share-based approach. It possible for the members at higher levels to compute the keys for its own

level along with all its descendant levels just by storing extra pre-positioned information. Our scheme is secure, even if a member compromises, it is not possible to get the group key unless activating share is obtained. The storage and computation costs are reduced al compared to Dexter et al. scheme. Periodic rekeying proposed using hashing technique is quite efficient as compared to multicasting an activating share.

REFERENCES

[1] Y. Amir, Y. Kim, C. Nita-Rotaru, J. Schultz, J. Stanton, and G. Tsudik. Secure Group Communication using Robust Contributory Key Agreement. *IEEE Transactions on Parallel and Distributed Systems*, 15,5, pp. 468-480, 2004.

[2] Y. Amir, C. Danilov, M. Miskin-Amir, J. Schultz, and J. Stanton. The Spread Toolkit: Architecture and Performance. *Technical Report*, CNDS-2004-1, Johns Hopkins University, 2004.

[3] C. Blundo, A. De Santis, A. Herzberg, S. Kutten, U. Vaccaro, M. Yung. Perfectly Secure Key Distribution for Dynamic Conferences, in *Advances in Cryptology-CRYPTO'92*, LNCS, 740, pp. 471-486, 1993.

[4] M. Burmester. and Y. Desmedt, A Secure and Efficient Conference Key Distribution System, *Advances in Cryptology - EUROCRYPT'94*.

[5] S.Dexter, R.Belostotskiy, A.M.Eskicioglu, Multi-layer Multicast Key Management with Threshold Cryptography, *Proceedings of Security, Steganography and Watermarking of Multimedia Content VI*: Sanjose, January 19-22, 2004.

[6] W. Diffie and M. Hellman. New Directions in Cryptography. *IEEE Transactions on Information Theory*, IT-22(6): 644-654, Nov 1976.

[7] L.R.Dondeti, S.Mukherjee and A.Samal, Survey and Comparison of Secure Group Communciation Protocols, *Technical Report*, University of Nebraska-Lincoln, June 1999.

[8] L.R.Dondeti, S.Mukherjee and A.Samal, DISEC: A Distributed Framework for Scalable Secure Many-to-many Communication, *In Proceedings of Fifth IEEE Symposium on Computers and Communications*, pp. 693-698, 2000.

[9] A.M.Eskicioglu, M.R.Eskicioglu, Multicast Security using Key Graphs and Secret Sharing, *Proceedings of the Joint International Conference on Wireless LANS and Home Networks and Networking*, Atlanta, GA, pp.228-241, August 26-29, 2002.

[10] A.M.Eskicioglu, S.Dexter, E.J.Delp, Protection of Multicast Scalable Video by Secret Sharing: Simulation results, *Proceedings of SPIE Security and Watermarking of Multimedia Content V*, Santa Clara, C.A. January 21-24, 2003.

[11] A.Fiat and M.Naor, Broadcast Encryption. In D.R.Stinson, editor, *Proceedings of CRYPTO'93*, pp 480-491, 1993.

[12] T.Hardjono and G.Tsudik, IP Multicast Security: Issues and Directions, *Annales De Taleum*, pp. 324-334, July-August 2000.

[13] Y. Kim, A. Perrig, and G. Tsudik. Simple and Fault-Tolerant Key Agreement for Dynamic Collaborative Groups. *In the* 7^{th} *ACM COnference on Computer and Communications Security*. ACM Press, pp. 235-244, 2000.

[14] P.P.C.Lee, J.C.S. Lui, and D.K.Y.Yau, Distributed Collaborative Key Agreement Protocols for Dynamic Peer Groups, *Proc. IEEE International Conference on Network Protocols (ICNP)*, Nov.2002.

[15] Marcel Waldvogel, Germano Caronni, Dan Sun, Nethalie Weiler, and Bernhard Platter. The Versakey framework: Versatile group key management, *IEEE Journal on Selected areas in Communications*, 17(9): 1614-1631, September 1999.

[16] D.A.McGrew and A.T.Sherman. Key establishment in Large Dynamic Groups using One-Way Function Trees. *IEEE Transactions on Software Engineering*. Volume 29, Issue 5, pp. 444-458, May 2003.

[17] S.Mittra, Iolus: A framework for Scalable Secure Multicasting. *In proceedings of the ACM SIGCOMM*. Volume 27,4 (NewYork, Sept.), ACM, NewYork, pp. 277-288, 1997.

[18] M.J.Moyer, J.R.Rao and P.Rohatgi, A Survey of Security Issues in Multicast Communication, *IEEE Network*, pp.12-23, Nov-Dec.1999.

[19] A.Perrig, Efficient Collaborative Key Management Protocol for Secure Autonomous Group Communication, *Proc. of International Workshop CrypTEC*, 1999.

[20] A.Puri and T.Chen, Multimedia Systems, Standards and Networks, Marcel Dekker Inc, 2000.

[21] S.Rafaeli and D.Hutchison, A Survey of Key Management for Secure Group Communication, *ACM Computing Surveys*, 35(3), pp. 309-329, Sept. 2003.

[22] H.Ragab Hassan, A.Bouabdallah, H.Bettahar, Y.Challal, An Efficient Key Management Algorithm for Hierarchical Group Communication, *Proceedings of the First International Conference on Security and Privacy for Emerging Areas in Communications Networks (SECURECOMM)*, pp. 270-276, Sept.5-9, 2005.

[23] A.Shamir, How to share a secret, CACM, 22(11), pp. 612-613, November 1979.

[24] Y.Sun, K.J.Ray Liu, Multi-layer Key Management for Secure Multimedia Multicast Communications, Proceedings of International Conference on Multimedia and Expo (ICME'03), July 6-9, 2003.

[25] Y.Sun, K.J.Ray Liu, Scalable Hierarchical Access Control in Secure Group Communications, *Proceedings of INFOCOM 2004, Twenty-third Annual Joint Conference of the IEEE Computer and COmmunication Societies*, March 7-11, 2004.

[26] W.Trappe, Y.Wang, K.J.R.Liu, Establishment of Conference keys in Heterogeneous Networks, *In Proceedings of IEEE International Conference on Communications*, Volume 4, pp. 2201-2205, 2002.

[27] D. Wallner, E. Harder and R.Agee. Key Management for Multicast: Issues and Architectures. *Request For Comments (Informational) 2627, Internet Engineering Task Force*, June 1999.

[28] C.K.Wong., Simon S. Lam., Keystone:A Group key management service, *In proceedings of International conference on Telecommunications*, Acapulco, Mexico, May 2000.

[29] C.K.Wong, M. Gouda, and S.S. Lam. Secure Group Communication Using key Graphs. *IEEE/ACM Transactions on Networking*, Volume 8, No.1, pp.16-30, Feb.2000.