Group Key Management: A Taxonomy

Saïd Gharout
Heudiasyc Lab. UMR CNRS 6599
Compiegne University of Technology
BP 20529, 60205, Compiegne, France
sgharout@hds.utc.fr

Yacine Challal
Heudiasyc Lab. UMR CNRS 6599
Compiegne University of Technology
BP 20529, 60205, Compiegne, France
ychallal@hds.utc.fr

***Abstract*—Group-oriented services are among the emerging technologies of the last few years. The advantages of using IP multicast in group communications, such as saving bandwidth, simplicity and efficiency are typically interesting for new services combining voice, video and text over Internet. Group key management, which is an important building block in securing group communications, has received a particular attention in both academic and industry research communities. This is due to the economical relevance of group-based applications, such as video on demand, video-conferencing, collaborative work. The key management concerns the distribution and updates of the key material each time a member joins or leaves the group. The dynamic aspect of group applications due to free membership joins and leaves in addition to members' mobility makes difficult the design of efficient and scalable key management protocols. In this paper we review existing key management protocols to secure group communication in the literature and analyze their advantages and shortcomings. We also provide our own original proposals, depicting their advantages over the existing solutions.**

Keywords-security; confidentiality; group communication; key management

I. INTRODUCTION *(HEADING 1)*

The phenomenal growth of the Internet in the last few years and the increase of bandwidth in today networks have provided both inspiration and motivation for the development of new group-oriented applications and services, combining voice, video and text "over IP". Nowadays, group-oriented applications are increasingly deployed over the Internet such as video conferencing, interactive group games, video on demand (VoD), IP-TV, e-learning, software updates, database replication and broadcasting stock quotes.

Unfortunately, the strengths of group-oriented applications, implemented either by IP Multicast [34], overlay Multicast [44] or other means is their lack of security. Indeed, the open and anonymous membership [14] and the distributed nature of multicasting are serious threats to the security of this communication model. For this purpose, many efforts have been conducted to address the many issues relating to securing group communication, such as: access control, confidentiality, authentication and watermarking.

Group communication confidentiality requires that only group members could read data even if the data is broadcasted into the entire network [20], [3]. Typically, the distribution of data with commercial value or State top-secret content requires the use of appropriate mechanisms to prevent non-legitimate recipients from having access to the content. To ensure confidentiality in group communication, only the customers authorized for the service would have access to the content for only the duration corresponding to their authorization. A straightforward solution is to encrypt the group-intended data by the sender with a group key, called *Traffic Encryption Key* (TEK), common to all authorized recipients. Therefore, this symmetric encryption should prevent other users from having access to the content. However, when the authorized duration for a recipient expires, it is necessary to change the common group key into a new key in order to prevent the leaving customer from having access to the content beyond the limit of his authorized duration. Therefore, the sender has to share the new TEK with all legitimate recipients except the leaving one. This phase is called re-keying, and should be performed each time a customer joins the secure session to prevent him from having access to old content (what is called Backward Secrecy) or leaves the session to prevent him from having access to future content (what is called Forward Secrecy). The role of a group key management protocol is to generate, update and distribute TEKs to legitimate group members. To ensure perfect backward and forward secrecy, a re-keying must be done each time there is membership changes (join or leave) in the group. The impact of this rekeying process on group members, commonly called *1-affects-n* phenomenon, measures the number of affected members by a re-keying process. This *1-affects-n* phenomenon is a challenging issue in designing group key management protocols. If the group size keeps increasing, such a phenomenon will significantly degrade the system performance. In the last few years, a lot of group key management

protocols have been conducted in the literature to address the confidentiality issue in group communication [31]. Even though a multitude of data confidentiality mechanisms currently exist for the fixed Internet, this security service remains a challenging problem in terms of scalability, efficiency, and performance. A critical problem with any re-key technique is scalability: as the re-key process should be triggered after each membership change, the number of TEK update messages may be important in case of frequent join and leave operations. Some solutions propose to organize the secure group into subgroups with different local TEKs. This reduces the impact of the key updating process, but needs decryption and re-encryption operations at the borders of subgroups. These operations may decrease the communication quality and causes computational overheads.

The rest of this paper is organized as follows: In section II, we overview group key management protocols. In section III,

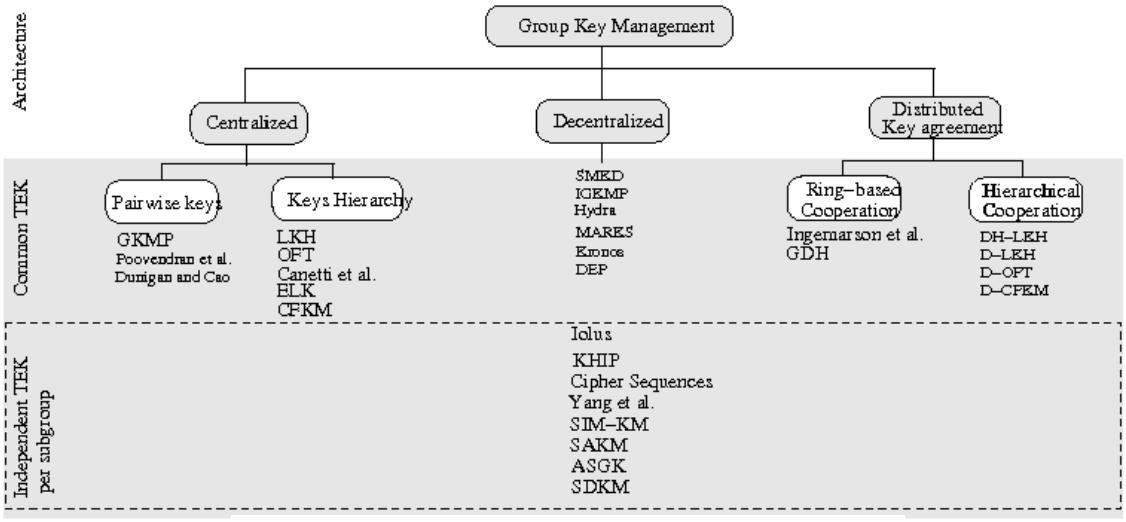

Figure 1. A taxonomy of group key management protocols

we discuss the advantages and inconveniences of each category of protocols. Finally, we conclude the paper in section IV.

II. GROUP KEY MANAGEMENT PROTOCOLS

Group key management has been extensively studied in the literature. Judge and Ammar [20], Rafaeli and Hutchison [31], Zhu and Jajodia [45] surveyed some group key management solutions. In this section we present relevant group key management protocols. In figure 1 we summarize some existing solutions in group key management. Existing key management solutions could be classified into three categories: centralized, decentralized and distributed architectures.

A. Centralized Architectures

In this approach, the key distribution function is assured by a single entity which is responsible for generating and distributing the *traffic encryption key* (TEK) whenever required. The most proposed centralized protocols in the literature use a common *Traffic Encryption Key* (TEK) for group members. Two techniques are used to ensure Key Management: Pairwise Keys and hierarchy of keys. In the pairwise keys the key server shares a secret key with each group member. These pairwise secret keys are generally called Key Encryption Keys (KEK) and are used to establish secure channels between the key server and each group member in order to re-distribute the TEK securely whenever required. In dynamic groups, a new TEK is sent to valid group members encrypted with their respective KEKs, including the new member in case of a join (to ensure backward secrecy), and excluding the leaving member in case of a leave (to ensure forward secrecy). A typical solution that fits into this category is the Group Key Management Protocol (GKMP) proposed by Harney and Muckenhirn in [16]. Similar solutions are those proposed by Dunigan and Cao in [12] and Poovendram et al. in [29]. This scheme has the drawback to require a high number of update messages (in the order of O(n) with n being the number of valid group members) to transmit the new TEK after membership changes. The aim of the hierarchy of keys approach is to reduce the required number of TEK update messages induced by re-keying after membership changes. Therefore, in contrast to the pairewise keys approach, the key server shares secret keys with subgroups of the entire secure group in addition to the individual secured channels. The Logical Key Hierarchy (LKH) protocol proposed at same time by Wong et al. in [41], [42] and Wallner et al. in [40], is a typical solution fitting into this category. The intermediate keys shared with different combinations of subgroups form a hierarchy (generally a binary tree) of keys.

The number of update messages induced by this protocol is in the order of log (n) with n being the number of valid group members. One-way Function Trees (OFT) [22], the One-way function chain tree [5], and the Efficient Large group Key distribution (ELK) [28] are variants of LKH protocol that allow to save some update message transmissions of intermediate keys of the hierarchy by replacing them with one-way function computations. In Centralized Flat table Key Management (CFKM) [39], the key hierarchy is replaced by a flat table and allows hence reducing the number of keys maintained by the Key Server.

B. Decentralized Architectures

In this category, a hierarchy of key managers share the labor of distributing the TEK to group members in order to avoid bottlenecks and single point of failure.We can distinguish protocols that use common TEK for the whole group and protocols that use a common TEK per subgroup.

1) Common TEK protocols

Ballardie proposed in RFC1949 [1] the Scalable Multicast Key Distribution (SMKD); a protocol where the main core of the multicast tree (constructed using CBT routing protocol [2]) mandates the secondary cores and other trusted routers to propagate the distribution of the TEK. This protocol has the drawback to be routing dependent. DeCleene et al. [15], [9] proposed the Intra-domain Group Key Management Protocol (IGKMP). In this architecture, the network is organized into administratively scoped areas in which a Domain Key Distributor (DKD) and many Area Key Distributors (AKD) are defined. Each AKD is responsible for one area. The DKD generates the TEK and multicasts it to the set of AKDs. When an AKD receives the TEK it propagates it to the group members of its area. In Hydra protocol (Rafaeli et al. [30]), the group is organized into subgroups, and each subgroup i is controlled by a server called the Hydra server (HS_i). If a membership change occurs at subgroup i, the corresponding HS_i generates the group TEK and sends it to the other HS_j*s* involved in that session. Setia et al. proposed the Kronos protocol [36], where the whole domain is divided into smaller areas managed by different Area Key Distributors (AKDs). After each specific period of time, each AKD generates a new TEK and distributes it to the members of its area. The AKDs share some secret parameters that allow them to generate the same TEK after each time period. In MARKS [4], Briscoe suggests slicing the time length to be protected into small portions of time and using a different key for encrypting each slice. The encryption keys are the leaves in a binary hash tree that is generated from a single seed. A blinding function, such as MD5 [32] is used to create the tree nodes. Dondeti et al. proposed the Dual Encryption Protocol (DEP) in [11]. DEP considers the case where intermediaries may be not trusted, and thereby proposes to use a double encryption scheme in TEK distribution in order to prevent those intermediaries from having access to propagated TEKs.

2) TEK per subgroup protocols

The common TEK protocols has the drawback to require that all group members commit to a new TEK, whenever a membership change occurs in the group, in order to ensure perfect backward and forward secrecy. This is commonly called *1-affects-n* phenomenon.

In order to mitigate the *1-affects-n* phenomenon, another decentralized approach consists in organizing group members into subgroups. Each subgroup uses its own independent TEK. Indeed, in this scheme when a membership change occurs in a subgroup, it affects only the members of the same subgroup. Mittra proposed in [23] the Iolus architecture which is a framework of a hierarchy of multicast subgroups. Each subgroup is managed by a Group Security Agent (GSA) which is responsible for key management inside the subgroup. A main controller called the Group Security Controller (GSC) manages the GSAs. Figure 2 illustrates a hierarchy with six subgroups. Each of them uses its own TEK. When a membership change occurs in a subgroup, only that subgroup is involved in a re-key process. This way, Iolus scales to large groups and mitigates *1-affects-n* phenomenon. However, Iolus has the drawback of affecting the data path. Indeed, there is a need for translating the data that goes from one subgroup, and thereby one key, to another. Shields et al. proposed another protocol that uses the same concept, called the Keyed HIerarchical multicast Protocol (KHIP) [37]. KHIP operates at the routing level where core routers ensure the translation of the packets. Instead of translating data itself, the protocol translates only the headers of the packets that contain a random key with which data is encrypted. In the case of the framework proposed by Molva and Pannetrat in [24], each time multicast messages pass through special nodes on the multicast tree, they are transformed using special functions called Cipher Sequences. Another decentralized solution with TEK per subgroup is the architecture proposed by Yang et al. in [43] where multicast group is organized into a set of subgroups, and each subgroup is managed by a Key Server which redistributes periodically a new TEK to its subgroup members. Challal et al. proposed in [6] SAKM protocol. The idea of SAKM is to organize dynamically over the time the multicast group into clusters of subgroups that use the same TEK. Mukherjee and Atwood proposed the SIM-KM protocol in [26] which uses proxy encryption [25] to transform data at the border of a subgroup. Proxy functions convert cipher text for one key into cipher text for another key without revealing secret decryption keys or clear text messages. This allows SIM-KM to do subgrouping with data transformation in order to limit the impact of re-keying, even thought intermediaries are not trusted entities. Huang and Mishra [17] proposed the Mykil protocol with fault tolerance and mobility support [18]. The Mykil

protocol combines the TEK per subgroup and common TEK approaches.

The ASGK protocol [7] (cf. figure 3), the group is organized into a hierarchy of administrative areas. Each area is managed by a local controller, that we call Area Security Agent (ASA). Each ASA is a member in its area and in its parent area. Thus, a ASA plays the role of a proxy for its area in the parent's area. When a ASA receives a message from the parent's area, it forwards it to the area under its control. Initially, all the areas use the same TEK, and thereby the task of the ASA would be simply to forward received messages. In this case, we say that the ASAs are in a passive state. In this configuration, a single membership change in any area would induce the distribution of a new TEK to the overall areas, because of using a common TEK. Hence, when one of the areas becomes dynamic, this area is isolated from the other areas in order to restrict the impact of re-keying to that dynamic area, and thereby reducing the *1-affects-n* phenomenon. In such a situation, the ASA of the dynamic area takes the decision to use an independent TEK for its area.

Therefore, upon receiving messages from the parent area (encrypted using the parent's TEK), the ASA will translate them to the TEK used within its area before forwarding them downward. In this case, we say that the ASA is in an active state. The fact that an ASA becomes active increases the computation power at the ASA and the delays before delivery of messages to its area and downward areas. The objective in ASGK is to design an adaptive architecture where the different ASAs switch from a state to another depending on the faced dynamism in

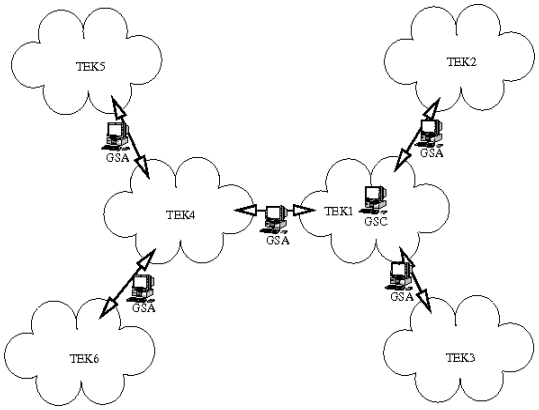

Figure 2. An example of a Iolus architecture

their areas and taking into consideration the induced computation overhead.

SDKM [13] is a decentralized architecture which uses the same architecture that ASGK, but the partitioning is made in a way that reduces both *1-affects-n* and delay in transmission due to key translation operations..

C. Distributed Key Agreement Protocols

In this approach, the group members cooperate to establish a group key. This improves the reliability of the overall system and reduces the bottlenecks in the network in comparison to the centralized approach. Ingemarson et al. [19] and Steiner et al. [38] proposed to extend Diffie-Hellman key agreement protocol [10] to group communication. In their schemes, group members perform intermediate DH exchanges, through the ring, and finally culminate into the common group key. Perrig et al. proposed in [27], [21] DH-LKH a distributed Diffie-Hellman implementation of LKH (cf. section II-A) through hierarchical collaboration. Rodeh et al. [33] and Waldvogel et al. [39] propose distributed versions of LKH (D-LKH) and CFKM (D-CFKM) protocols respectively following a hierarchical cooperation of group members. Seba et al. [35] proposed the Fault-Tolerant Group Diffie-Hellman (FTGDH) protocol where the key agreement is established only with members which are supposed correct. For that, failure detectors of Chandra and Toueg [8] are used. Indeed, the member Mi does not send its contribution to the successor Mi+1 but to correct (non faulty) successor. In FTGDH, each member can start the key agreement.

III. DISCUSSION

We notice that proposed solutions in the literature suffer from great concerns depending on group dynamism: protocols with common TEK suffers from the *1-affects-n* phenomenon, where a single group membership change (join or leave) results in a rekeying process that disturbs all group members to update the TEK. Moreover, centralized protocols are not scalable, and

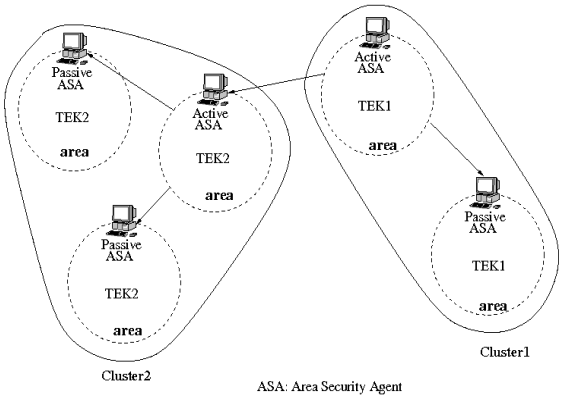

Figure 1. Adaptive clustering for Scalable Group Key management Architecture

distributed ones bring new challenges such as synchronization, conflict resolution and required time to construct the key. On the other hand, decentralized protocols with TEK per subgroup approach reduce the *1-affects-n* problem. This is advantageous for highly dynamic multicast groups. However, this

approach requires translation of sent messages whenever they pass from a subgroup to another, and this may not be supported in applications which are sensitive in transmission time and do not tolerate delays in packet delivery. Besides, this approach would not be worthy with relatively static groups because the multiple translations would induce avoidable delays and useless computation overheads. These shortcomings are due to the lack of dynamism awareness in existing group key management schemes.

IV. CONCLUSION

In this paper we have presented relevant group key management protocols. We have classified existing solutions into three main categories: centralized, decentralized and distributed. These categories can also be classified into two approaches: the common TEK approach and the TEK per subgroup approach. Both proposed approaches suffer from great concerns depending on group dynamism: the common TEK approach suffers from the *1-affects-n* phenomenon, where a single group membership change (join or leave) results in a re-keying process that disturbs all group members to update the TEK. Moreover, centralized protocols are not scalable, and distributed ones bring new challenges such as synchronization and conflict resolution. Time-driven rekeying protocols attempt to reduce the *1-affects-n* phenomenon by batch re-keying, but then cannot be used with critical applications that require taking into consideration the membership change instantly. On the other hand, the TEK per subgroup approach reduces the *1-affects-n* problem. This is advantageous for highly dynamic multicast groups. However, this approach requires transformation of sent messages whenever they pass from a sub-group to another, and this may not be tolerated by applications that are sensitive to packet delivery delay variations. Besides, this approach would not be worthy with relatively static groups because the multiple transformations would induce avoidable delays and useless computation overheads. These shortcomings are due to the lack of dynamism awareness in existing group key management schemes. Approaches presented in SAKM [6], ASGK [7] and SDKM [13] aim to reduce the impact of *1-affects-n* phenomenon and translation overhead with using adaptive clustering of encryption areas into clusters that use the same *Traffic Encryption Key*.

References

[1] A. Ballardie. Scalable Multicast Key Distribution, May 1996. RFC 1949.

[2] A. Ballardie. Core Based Trees (CBT version 2) Multicast Routing protocol specification, September 1997. RFC 2189.

[3] M. Baugher, R. Canetti L. Dondeti, and F. Lindholm. RFC4046: Multicast Security (MSEC) Group Key Management Architecture. IETF RFC 4046, April 2005.

[4] B. Briscoe. MARKS: Zero Side Effect Multicast Key Management Using Arbitrarily Revealed Key Sequences. In NGC '99: Proceedings of the First International Workshop on Networked Group Communication, pages 301–320, London, UK, November 1999. Springer-Verlag.

[5] R. Canetti, J. Garay, G. Itkis, D. Micciancio, M. Naor, and B. Pinkas. Multicast Security: A taxonomy and Efficient Constructions. IEEE INFOCOM, pages 708–716, March 1999.

[6] Y. Challal, H. Bettahar, and A. Bouabdallah. SAKM: a scalable and adaptive key management approach for multicast communications. ACM SIGCOMM Computer Communications Review, 34(2):55–70, 2004.

[7] Y. Challal, S. Gharout, A. Bouabdallah, and H. Bettahar. Adaptive clustering for Scalable Key Management in Dynamic Group Communications. Inderscience International Journal of Security and Networks (IJSN), 3(2), 2008.

[8] T. D. Chandra and S. Toueg. Unreliable failure detectors for reliable distributed systems. Journal of the ACM, 43(2):225–267, Mars 1996.

[9] B. DeCleene, L. Dondeti, S. Griffin, T. Hardjono, D. Kiwior, J. Kurose, D. Towsley, S. Vasudevan, and C. Zhang. Secure group communications for wireless networks. Military Communications Conference, 2001. MILCOM 2001. Communications for Network-Centric Operations: Creating the Information Force. IEEE, 1:113–117, October 2001.

[10] W. Diffie and M.E. Hellman. New directions in cryptography. IEEE Transactions on Information Theory, 22(6):644–654, November 1976.

[11] L. R. Dondeti, S. Mukherjee, and A. Samal. Scalable secure one-tomany group communication using dual encryption. Computer Communications, 23(17):1681–1701, November 2000.

[12] T. Dunigan and C. Cao. Group Key Management. Technical Report ORNL/TM-13470, 1998.

[13] S. Gharout, Y. Challal, and A. Bouabdallah. Scalable Delay-constrained Multicast Group Key Management. International Journal of Network Security (IJNS), 7(2):153–167, September 2008.

[14] B. Haberman and J. Martin. RFC5186: Internet Group Management Protocol Version 3 (IGMPv3) / Multicast Listener Discovery version 2 (MLDv2) and Multicast Routing Protocol Interaction . IETF RFC 5186, May 2008.

[15] T. Hardjono, B. Cain, and I. Monga. Intra-domain Group Key Management for Multicast Security. IETF Internet draft, September 2000.

[16] H. Harney and C. Muckenhirn. RFC2093: Group Key Management Protocol (GKMP) Architecture. IETF RFC 2093, July 1997.

[17] J. H. Huang and S. Mishra. Mykil: A Highly Scalable and Efficient Key Distribution Protocol for Large Group Multicast. IEEE GLOBECOM. Global Telecommunications Conference, 3:1476 – 1480, December 2003.

[18] J. H. Huang and S. Mishra. Support for Mobility and Fault Tolerance in Mykil. Proceedings of the 2004 International Conference on Dependable Systems and Networks (DSN04), pages 537 – 546, June/July 2004.

[19] I. Ingemarson, D. Tang, and C. Wong. A Conference Key Distribution System. IEEE Transactions on Information Theory, 28(5):714–720, September 1982.

[20] P. Judge and M. Ammar. Security Issues and Solutions in Multicast Content Distribution: A Survey. IEEE Network, 17(1):30–36, January/February 2003.

[21] Y. Kim, A. Perrig, and G. Tsudik. Simple and fault-tolerant Key Agreement for Dynamic Collaborative groups. 7th ACM Conference on Computer and Communications Security, pages 235–244, November 2000.

[22] D. A. McGrew and A. T. Sherman. Key Establishement in Large Dynamic Groups using One-way Function Trees. IEEE Transactions On Software Engeneering, 29(5):444 – 458, May 2003.

[23] S. Mittra. Iolus: a framework for scalable secure multicasting. In SIGCOMM '97: Proceedings of the ACM

SIGCOMM '97 conference on Applications, technologies, architectures, and protocols for computer communication, pages 277 – 288, New York, NY, USA, 1997. ACM Press.

[24] R. Molva and A. Pannetrat. Scalable multicast security in dynamic groups. In CCS '99: Proceedings of the 6th ACM conference on Computer and communications security, pages 101–112, New York, NY, USA, 1999. ACM Press.

[25] R. Mukherjee and J.W. Atwood. Proxy Encryptions for Secure Multicast Key Management. Proceedings. 28th Annual IEEE International Conference on Local Computer Networks, 2003. LCN '03., pages 377–384, October 2003.

[26] R. Mukherjee and J.W. Atwood. SIM-KM: Scalable Infrastructure for Multicast Key Management. IEEE Local Computer Networks- LCN'04, pages 335–342, November 2004.

[27] A. Perrig. Efficient Collaborative Key Management Protocols for Secure Autonomous Group Communication. In Proceedings of the 1999 International Workshop on Cryptographic Techniques and E-Commerce (CrypTEC '99), pages 192–202, July 1999.

[28] A. Perrig, D. Song, and J. D. Tygar. ELK, a New Protocol for Efficient Large-Group Key Distribution. Proceedings of the 2001 IEEE Symposium on Security and Privacy, S&P 2001, pages 247 – 262, May 2001.

[29] R. Poovendram, S. Ahmed, S. Corson, and J. Baras. A Scalable Extension of Group Key Management Protocol. 2nd Annual ATRIP Conference, pages 187–191, February 1998.

[30] S. Rafaeli and D. Hutchison. Hydra: A decentralised group key management. In WETICE '02: Proceedings of the 11th IEEE International Workshops on Enabling Technologies, pages 62 – 67, Washington, DC, USA, 2002. IEEE Computer Society.

[31] S. Rafaeli and D. Hutchison. A Survey of Key Management for Secure Group Communication. ACM Computing Surveys, 35(3):309–329, September 2003.

[32] R. Rivest. The MD5 Message-Digest Algorithm, April 1992. RFC 1321.

[33] O. Rodeh, K. Birman, and D. Dolev. Optimized group rekey for group communication systems. Network and Distributed System Security, pages 39–48, February 2000.

[34] P. Savola. RFC5110: Overview of the Internet Multicast Routing Architecture. IETF RFC 5110, January 2008.

[35] H. Seba, A. Bouabdallah, and N. Badache. A new approach to scalable and fault-tolerant group key management protocols. Journal of High Speed Networks, 13(4):283–296, 2004.

[36] S. Setia, S. Koussih, S. Jajodia, and E. Harder. Kronos: A scalable group re-keying approach for secure multicast. Proceedings of the 2000 IEEE Symposium on Security and Privacy, 2000. S&P 2000., pages 215–228, May 2000.

[37] C. Shields and J.J. Garcia-Luna-Aceves. KHIP-A scalable protocol for secure multicast routing. ACM SIGCOMM Computer Communication Review, 29(4):53–64, October 1999.

[38] M. Steiner, G. Tsudik, and M. Waidner. Diffie-Hellman key distribution extended to group communication. 3rd ACM Conference on Computer and Communications Security, pages 31–37, March 1996.

[39] M. Waldvogel, G. Caronni, D. Sun, N. Weiler, , and B. Plattner. The VersaKey Framework : Versatile Group Key Management. IEEE Journal on Selected Areas in Communications (Special Issues on Middleware), 17(8):1614–1631, August 1999.

[40] D. Wallner, E. Harder, and R. Agee. RFC2627: Key Management for Multicast : Issues and Architecture. IETF RFC 2627, June 1999.

[41] C. K. Wong, M. Gouda, and S. S. Lam. Secure Group Communications Using Key Graphs. ACM SIGCOMM, pages 68–79, 1998.

[42] C. K. Wong, M. Gouda, and S. S. Lam. Secure Group Communications Using Key Graphs. IEEE/ACM Transactions on Networking, 8(1):16–30, February 2000.

[43] Y. R. Yang, X. S. Li, X. B. Zhang, and S. S. Lam. Reliable Group Rekeying: A Performance Analysis. In SIGCOMM '01: Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications, pages 27 – 38, New York, NY, USA, 2001. ACM Press.

[44] C.K. Yeo, B.S. Lee, and M.H. Er. A survey of application level multicast techniques. Computer Communications, 27(15):1547–1568, September 2004.

[45] S. Zhu and S. Jajodia. Scalable group rekeying for secure multicast: A survey. In Proc. 5th International Workshop on Distributed Computing, 2918:1–10, 2004.