SECURE AND EFFICIENT DECENTRALIZED GROUP KEY ESTABLISHMENT PROTOCOL FOR ROBUST GROUP COMMUNICATION

Rakesh Chandra Gangwar
Assistant Professor, Department of Computer Science
Beant College of Engineering and Technology,Gurdaspur (Punjab)-143521 India
{*rakeshgangwar1965*}*@gmail.com*

Abstract

Now-a-days Internet has become the common media of communication. Many group communication application such as pay-per-view, stock quote distribution, voice- and video-conferencing, white-boards, distributed simulations, and replicated servers of all types, etc can easily be conducted on the Internet. For conducting such applications, group key is often needed, which can be established by group key establishment protocol. Although many group key establishment protocols have been proposed in the literature, yet nothing has been done to establish their suitability for aforementioned applications. In this paper, we present the succinct description of different decentralized group key establishment protocols and analyze them against parameters such as key independence, one-affects-all, local rekey, data transformation.

Keywords: *Protocol, Key Management, Communication, Group Key Establishment Architecture*

1. INTRODUCTION

The explosive growth of the Internet has increased both the number and the popularity of applications that require a reliable group communication infrastructure, such as pay-per-view, stock quote distribution, voice- and video-conferencing, white-boards, distributed simulations, and replicated servers of all types. Secure group communication is crucial for building distributed applications that work in dynamic network environments and communicate over insecure networks such as the global Internet. Key management is the base for providing common security services (data secrecy, authentication and integrity) for group communication. There are several approaches to group key management.

In this paper we present the working of different decentralized group key establishment protocols briefly and subsequently do the complexity analysis of them to judge their suitability for secure and efficient conduct of aforementioned applications.

The rest of the paper is structured as follows: Section 2 presents the group key establishment architecture. Section 3 describes the decentralized group key establishment protocols, where rekeying is performed based on membership. Section 4 describes the decentralized group key establishment protocols, where rekeying is performed based on time. Section 5 present the complexity analysis and finally Section 6 concludes the paper

2. GROUP KEY ESTABLISHMENT ARCHITECTURE

Group key establishment in group communication can broadly be classified into two categories: first is distributory wherein the group is created by a single entity that is group controller and key server (GCKS), and second is contributory, wherein the group key is established by the equal contribution of all participants.
Distributory approach can further be classified into to categories such as centralized and decentralized. In centralized key establishment protocol, the key server is solely responsible for creation and distribution of group key. And in decentralized

group key establishment protocol, hierarchy of key servers share the burden of distributing the group key to group members in order to avoid bottlenecks and single point of failure. We distinguish two subcategories of decentralized group key establishment protocols, where the group key is modified after each membership change (membership based), or systematically after each slot of time (time based). The complete group key establishment architecture is shown in Fig. 1.

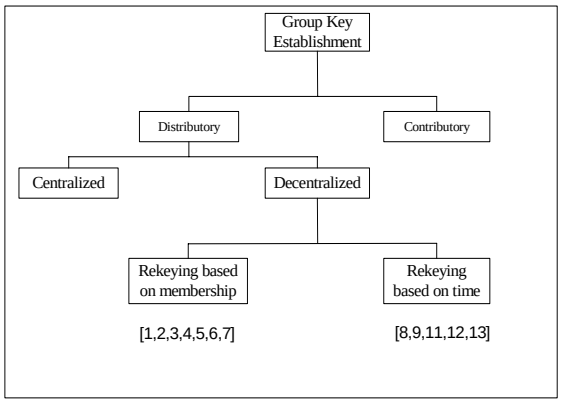

Figure 1: Group Key Establishment Architecture

3. REKEYING BASED ON MEMBERSHIP

In this subcategory of protocols, the group key is changed each time a join or a leave operation occurs in the group. We present some protocols based on this approach.

3.1. Scalable Multicast Key Distribution

Ballardie[1] proposed the scalable multicast key distribution (SMKD) protocol, which uses the tree built by the Core Based Tree (CBT) multicast routing protocol [2,3] to deliver keys to multicast group members. In the CBT architecture, the multicast tree is rooted at a main core. Secondary cores can exist eventually. The main core creates an access control list (ACL). Group key and key encryption key (KEK) are used to update the group key. The ACL, the group key and the key encryption key are transmitted to secondary cores and other nodes, when they join the multicast tree after their authentication. Any router or secondary core authenticated with the primary core can authenticate joining members and use the ACL to distribute the keys, but only the main core generates those keys. The SMKD protocol does not provide the forward secrecy when a member leaves the group. It has to execute afresh each time when a member departs.

3.2. Intra-domain Group Key Management Protocol

DeCleene et al. [4,5] proposed the Intra-domain Group Key Management Protocol IGKMP. Architecture divides the network into administratively scoped areas. There are a Domain Key Distributor (DKD) and many Area Key Distributors (AKDs). Each AKD is responsible for one area. Fig. 2 exemplifies this architecture. The group key is generated by the DKD and is propagated to the group members through the AKDs. The DKD and AKDs belong to a multicast group called All-KD-Group. The DKD uses this group to transmit re-key messages to the AKDs who re-key in turn their respective areas.

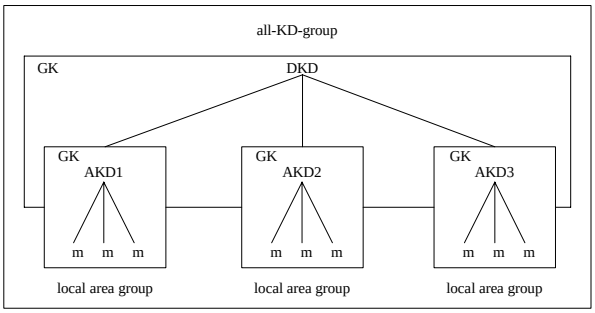

Figure 2: Intra-domain Group Key Management Protocol architecture

This architecture suffers from a single point of failure, which is the DKD that is the entity responsible for generating the group key. Besides, in case of an AKD failure, members belonging to the same area will be not able to access the group communication.

3.3. Hydra protocol

Rafeli and Hutchison [6] proposed Hydra protocol, wherein the group is organized into smaller subgroups and a server called the Hydra server (HS_i) controls each subgroup *i*. If a membership change occurs at subgroup *i*, the corresponding HS_i generates the group key and sends it to the other HS_j involved in that session. In order to have the same group key distributed to all HSs, a special protocol is used to ensure that only a single valid HS is generating the new group

key whenever required. Fig 3 depicts the Hydra architecture.

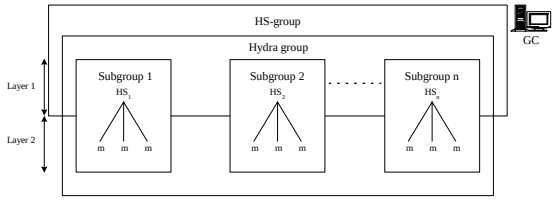

Figure 3: Hydra architecture

3.4. Baal protocol

Baal Chaddoud et al. [7] proposed a protocol that is known as Baal protocol, which defines three entities:

1. The Group Controller (GC): It maintains a participant list and creates and distributes the group key to group members via local controllers.
2. Local Controllers (LC): The GC delegates a LC to each subnet (generally a local network) to manage the keys within its subnet. When a LC receives a new group key, it distributes it to the members connected to its subnet. Besides, a LC can play the role of the GC by generating and distributing new group keys after membership changes following some coordination rules.
3. Group member: It belongs to participation list.

When a membership change occurs at a subnet, the corresponding LC can generate a new group key and distribute it to its subnet and to the other members via their LCs. To assure that a single LC generates a new group key at a time, the GC assigns a priority to each LC and when many LCs distribute simultaneously a new group key, the LCs are instructed to commit to the group key issued by the LC having the highest priority.

4. REKEYING BASED-ON-TIME

In this subcategory of protocols, the group key is changed after each specific period of time. Thereby, the departing members are not excluded immediately from having access to the secure content. Similarly, new members are delayed up to the beginning of a new interval of time. Some protocols based on this concept are as follows:

4.1. Kronos protocol

Setia et al. [8] proposed the Kronos protocol, which is driven by periodic re-keying rather than membership changes that means a new group key is generated after each time interval rather than after each membership change. In Kronos protocol, each domain is divided into many areas managed by different AKDs as in IGKMP protocol. However, in Kronos, the DKD does not multicast the group key each time to the AKDs. Instead of that, each AKD generates independently the same group key whenever required and re-keys the members belonging to its area. To implement this scheme, the AKDs' clocks should be synchronized, and the AKDs have to agree on a re-key period. Second, the DKD transmits secret factors K and G_0 to AKDs using secure channels. To generate the group key G_{i+1}, AKDs calculate after each period of time: $G_{i+1} = E_K(G_i)$, which is the encryption of the previous group key (G_i) with the encryption algorithm E using the secret key K.

4.2. MARKS protocol

In MARKS protocol, Briscoe [9] 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 leaves in a binary hash tree that is generated from a single seed. A blinding function, such as MD5 [10] is used to create the tree nodes. Fig. 4 depicts an example of the generated binary tree whose leaves are the keys that correspond to the different slices.

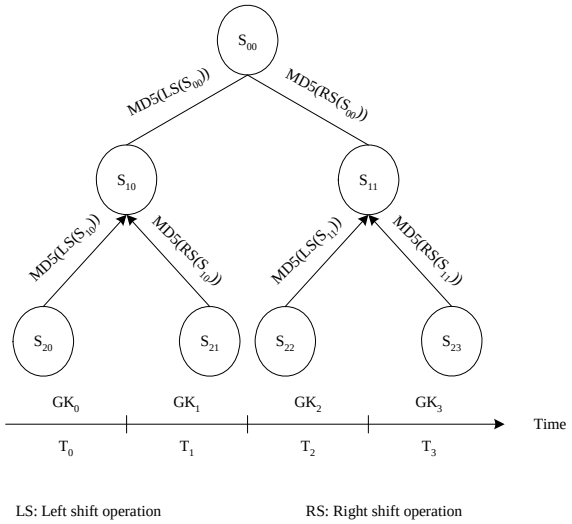

Figure 4: MARKS key generation tree

Each intermediate node (including the root) is allowed to generate two children (left and right children). The left node is generated by shifting its' parent one bit to the left and applying the

blinding function on it. The right node is generated by shifting its' parent one bit to the right and applying the blinding function on it. Users willing to access the group communication receive the seeds needed to generate the required keys.

The system cannot be used in situations where a membership change requires the change of the group key, since the keys are changed as a function of the time. The distribution of the seeds and the management of receivers' queries are assured by a set of key managers.

4.3. Dual Encryption Protocol

A common drawback of most of decentralized protocols is the involvement of a high number of intermediary parties. In practice it is difficult to assume trustiness for all of these entities. In order to solve the problem of trusting third parties, Dondeti et al. [11,12,13] proposed the Dual Encryption Protocol (DEP). In their work, they suggest hierarchical subgrouping of the group members where a sub-group manager (SGM) controls each subgroup. There are three type of KEKs and one Data Encryption Key (DEK), KEK_{i1} is shared between a SGM_i and its subgroup members, KEK_{i2} is shared between the Key Server (KS), and the group members of subgroup *i* excluding SGM_i. Finally, KS shares KEK_{i3} with SGM_i. In order to distribute the DEK to the group members, the KS generates and transmits a package containing the DEK encrypted with KEK_{i2} and encrypted again with KEK_{i3}. Upon receiving the package, SGM_i decrypts its part of the message using KEK_{i3} and recovers the DEK encrypted with its subgroup KEK (KEK_{i2}), which is not known by the SGM_i. SGM_i encrypts this encrypted DEK using KEK_{i1} shared with its subgroup members and sends it out to subgroup *i*. Each member of subgroup *i* decrypt the message using KEK_{i1} and then, decrypting the message using KEK_{i2} (shared with KS) and receives DEK. Therefore, the DEK cannot be recovered by any entity that does not know both keys. Hence, although there are third parties involved in the management (SGMs), they do not have access to the group key (DEK). When the membership of subgroup *i* changes, the SGM_i changes KEK_{i1} and sends it to its members. Future DEK changes cannot be accessed by members of subgroup *i* that did not received the new KEK_{i1}.

5. COMPLEXITY ANALYSIS

Table 1 presents the comparison of above protocols based on the parameters such as one-affects-all , key independence, local rekey (membership changes in a sub-group should be treated locally), data transformation (data is transformed using some means when messages pass from a sub-group to another).

6. CONCLUSION

The drawback of Kronos protocol is key independence because it generates new keys based on old ones. In such scenario, if any previous key is compromised, all successive keys are disclosed. The same thing happens with MARKS if a seed is compromised, all keys are compromised. Although all aforementioned protocols divide the whole group into subgroups, yet they suffer from one-affects-all phenomena because of using the same group key for all subgroups. Hydra and Baal protocols show the best results for different parameters.

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] T. Ballardie, I.P. Francis, and J. Crowcroft, "Core Based Trees: an Architecture for Scalable Inter-domain Multicast Routing", ACM SIGCOMM, pages 85–95, 1993.

[4] 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", MILCOM, June 2001.

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

[6] S. Rafaeli and D. Hutchison., "Hydra: a decentralized group key management", 11th IEEE International WETICE: Enterprise Security Workshop, June 2002.

[7] G. Chaddoud, I. Chrisment, and A. Shaff, "Dynamic Group Communication Security",

6th IEEE Symposium on computers and communication, 2001.

[8] S. Setia, S. Koussih, S. Jajodia, and E. Harder., "Kronos: A scalable group re-keying approach for secure multicast", IEEE Symposium on Security and Privacy, May 2000.

[9] B. Briscoe, "MARKS: Multicast key management using arbitrarily revealed key sequences", 1st International Workshop on Networked Group Communication, November 1999.

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

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

[12] L.R. Dondeti, S. Mukherjee, and A. Samal, "Comparison of Hierarchical Key Distribution Schemes", IEEE Globcom Global Internet Symposium, 1999.

[13] L.R. Dondeti, S. Mukherjee, and A. Samal, "Survey and Comparison of Secure Group Communication Protocols", 1999. Technical Report.

Table 1: Comparison of Decentralized Group Key Establishment Protocols

Protocol	one-affects-all	Key independence	Local rekey	Data Transformation
SKMD	Yes	Yes	No	No
IGKMP	Yes	Yes	No	No
Hydra	Yes	Yes	Yes	No
Baal	Yes	Yes	Yes	No
Kronos	-	No	No	No
MARKS	-	No	No	No
DEP	Yes	Yes	No	No