Group Key Management in Multicast Security

Firdaus Mah
Helsinki University of Technology

Abstract

One of the major obstacles for the deployment of multicast is the lack of security. And in multicast security, key management for securing group communication is an important area that needs to be addressed. This paper will give an overview on group key management that are standardized and proposed within Internet Engineering Task Force (IETF) Multicast Security Workgroup.

Group key management is mainly build upon the trust model developed by Group Key Management Protocol (GKMP). The trust model gives an overview of the relationship between entities in a multicast security system. The Group Security Association (GSA) that associates cryptographic attributes in a multicast group. Unlike unicast SA, GSA requires to take into consideration protecting multiple users at the same time and able to scale from small to large number of users. Therefore, it requires multiple SAs that are not only used for protecting data traffic but also for other means such as registration and rekeying. This paper will examine the main components of the trust model and GSA. Base on that, the paper will layout the architecture and design requirements needed for group key management protocol. There are several group key management protocols that are proposed, the paper will however elaborate mainly on Group Security Association Key Management Protocol (GSAKMP).

An important component for protecting group secrecy is rekeying. In the event of the group being compromise due to members leaving and joining the group or an unauthorized access to a cryptographic key, a secure group key management protocol requires to handle it by performing rekeying operation. The paper will explain on a rekey algorithm called Logical Key Hierarchy (LKH) that is generally recommended for performing group rekeying operation.

1 Introduction

As internet continues to grow phenomenally, the desire for more efficient ways of distributing information across network is increasing. Multicast technology is used for distributing data to a group of participants by conserving bandwidth more efficiently than traditional unicast mechanism. This is done by replicating IP streams in the router at the same time thus achieving better delivery to multiple users [31]. This would mean conservation of computational resources of the sender and bandwidth efficiency in the network. A group membership can be performed using the Internet Group Multicast Protocol (IGMP) protocol [1]. It provides admission control operation such as "join" and "leave". Some examples of applications that take advantage of multicast technology are video conferencing, digital broadcasting, software distribution and electronic learning. The Figure 1 depicts an example of a multicast distribution tree where information from a single sender traversing to multiple receivers.

![Multicast distribution tree](image)

Security in the other hand is a critical element for the deployment of IP multicast technology. According to the recommendation from International Standards Organization (ISO), criterias [13] for designing a secure system are confidentiality, integrity, authentication, non-repudiation and access control. Cryptography is fundamental to these criterias as it involves asymmetric and symmetric key operations. Therefore, management of these keys plays an important role in designing multicast security.

On the standardization front, IETF has formed Multicast Security (MSEC) Workgroup [23] to standardize protocols for securing group communication over the internet. The workgroup has made it an important objective to standardize group key management architecture. This paper will incorporate many of the features documented in MSEC Group Key Management Architecture [11], Multicast Security Architecture [7], Group Security Association Key Management Protocol [10], Group Domain of Interpretation [6] and other related group key management documentations within IETF multicast security workgroup.

2 Group Key Management

Group Key Management refers to the process of managing cryptographic keys in a secure multicast group [2]. The han-
dling of these keys in multicast security is complex because it has to operate in a very dynamic environment. Typically in a unicast key management mechanism usually works only between two hosts. Multicast in the other hand requires handling scenarios that involves one-to-many or many-to-many communication. Consequently, these may require more than one keys required for a session. Ideally, due to handling of more cryptographic keys, a trusted entity is needed to manage them. Therefore, multicast security workgroup proposals are mainly base on the Group Controller and Key Server (GCKS) trust model developed in GKMP [3]. It provides a high-level overview on the relationship of the entities involved in multicast security that is centered on GCKS. Additionally, Group Security Association (GSA) [6] is an important element for the construction of a secure multicast group. It provides a way to associate cryptographic attributes so that all members in the group can communicate together securely. The trust model and GSA forms the basis for group key management. The following subsection will elaborate on them.

2.1 Trust Model

The trust model used as the building block for group key management are mainly base on the Group Controller and Key Server (GCKS) model developed in GKMP [3]. Figure 2 shows the components of this trust model. The model provides a high-level overview on all entities relating to group key management. The idea behind this is a trusted entity called GCKS.

![Figure 2: Group Controller and Key Server Trust Model](image)

The Group Controller/Key Server (GCKS) is responsible for the issuance and management of cryptographic keys. GCKS also plays an important role for admission control for the multicast group and performs user authentication when a member joins the group. Since this component is so integral to the trust model, all components require interfacing with GCKS. Consequently, any compromise to GCKS can possibly lead to the security integrity breakdown of the multicast system.

In figure 2, the Sender is the entity that transmits data to the multicast group while the Receiver is the entity that receives the data. In a multicast group, there can be a 1-to-N multicast group where only a single sender is allowed to transmit data while in a M-to-N multicast group more than one sender is allowed to transmit data to the group. In some cases, all members of the groups are allowed to be senders. The senders and receivers require performing authentication, admission control and keys downloading from GCKS. Policies in the other hand has to be

The Policy server shown in figure 2 is responsible for creation and management of the policies. The policies are managed, disseminated and accessed via GCKS to group the members. Policies are important in multicast security group key management, as it will decide on how keys are to be managed and cryptographic operation to be performed.

The relationship of these entities illustrated in the model provides a good reference framework for an architectural group key management design.

2.2 Group Security Association (GSA)

In typical unicast operation in IPsec [4], Security Association (SA) is used as a way to map identical security attributes so that two host can communicate securely. These attributes include cryptographic keys, algorithm, identifier and other related attributes used to associate with the security material. Multicast however needs to handle more complex scenarios where it has to communicate with more than one host at a time. Due to its complexity, it requires more than one key to keep a multicast session secure. Therefore, group key management introduces the concept of Group Security Association (GSA) [6] to group multiple SAs. In multicast security system, a GSA can contain three or more SAs.

GSA is constructed in two distinct ways. They are the superset and aggregation that is illustrated in Figure 3. Superset is when GSA forms the superset of an SA. An example would be signed credentials to certify group membership if it will be allowed new keys when a join or leave operation occurs. Aggregation is when a GSA comprises multiple SAs where each of them may be used for different purposes such as Key Encryption Keys (KEK) and Traffic Encryption Keys (TEK) that will be explained in dynamic secrecy section.

![Figure 3: Relationship of GSA to SA](image)

SAs in GSA according to multicast group security architecture [7] are categorized into three types. They are the registration SA, rekey SA and data security SA. The registration SA is used separately between GCKS and individual group members. The rekey SA on the other hand is the multicast SA used between GCKS and all group members. And finally, the data security SA is the multicast SA between each multicast source sender and the group receivers. The estab-
3 Architecture

As mentioned earlier in the paper, multicast security workgroup has made it an important objective to standardize the group key management architecture. It is described and documented in the internet draft [11]. Its purpose is to be used as a guideline for designing group key management protocols supporting different types of applications, transport and network layer security protocol.

3.1 Requirements

According to group key management architecture [11], it has listed out twelve requirements for group key management protocol. The following is the list of the requirements.

First, the requirement suggests that group members should receive SA that includes cryptographic keys for encryption and authentication, the policies to describe these keys and attributes for referencing the SA. Second, GCKS requires providing access control mechanism in regards to the policies associated with the group keys. These access control mechanism may include definition and enforcement for group membership, key management functionalities and policies. Third, when keys are created, it requires to have a predetermined lifetime. The keys may also be periodically updated to maintain group secrecy. Fourth, key material should always be delivered in a secure manner to group members. It has to be protected using integrity and verifiable mechanism such as public key cryptography or keyed hash. Fifth, the protocols designed for group key management should be able to protect against replay attacks and denial of service (DOS) attacks. Sixth, the protocol should provide the ability for the addition or removal of group members. In the event of addition, group members can be optionally denied access to previous group session keys. On the other hand, member removal of the group will cause the member to lose access to the key material. Seventh, the protocol should allow support for scalable group rekeying. Unicast rekeying operations is undesirable as it will overload GCKS managing a large multicast group. Eight, when designing the protocol, it should be compatible with current infrastructure and performance requirements of the data security application. Examples of these applications are IPsec security protocols such as Authentication Header (AH) [26] and Encapsulating Security Protocol (ESP) [27] and application layer security protocols such as Secure Real Time Protocol (SRTP) [29] and Transport Layer Protocol (TLS) [30]. Ninth, the protocol should offer framework for updating transforms, authorization and authentication system. Tenth, it should be designed to protect against collusion among excluded members or non-members. Collusion is the term used when members collaborate to deduce keys that they are not privileged to have. Eleventh, the protocol should provide a mechanism to be able to recover from some of the compromised key material securely if not all. Twelve, the key management protocol should support deployment issues such as Network Address Traversal (NAT) [28] and legacy systems that are widely in used.  

These requirement however is not intended to be an exhaustive list nor applicable to all applications. So, an adaptive approach should be used when incorporating these requirements in the design.

3.2 Design

Base on the trust model, the management of keys in principle is between GCKS and the group member. Figure 4 illustrates the design of a block diagram for group key management between GCKS and the group member that is documented in the group key management architecture [11]. The group key management protocol is intended to create GSA that consists of Data SA, an optional Rekey SA and a Registration SA. From the diagram, GKM is the Group Key Management entity that manages the key in the host. It also illustrates two arrows where the GCKS and member GKM communicates with each other. They are the Registration and Rekey protocol used to establish its SA. Figure 4 also shows functional blocks that can be peculiar to different types of operating system where the specification can have support for IPsec Security Association Database (SAD) and Security Policy Database (SPD). The diagram also shows "CRED" block that stands for credential stores that can be defined by different vendors. The block "Control" task is to instruct GCKS to establish group, admission control and manages joining and leaving operations. "Control" also includes authorization that is in accordance with the policy that is GCKS specific. Besides that, "Control" can perform large-scale announcement to perform group rekeying in the event of keys being compromised.

4 Group Key Management Protocols

There are several varieties of group key management protocol. Within multicast security workgroup, I have identified three standards and draft relating to group key management protocol. They are Group Security Association Key
Management Protocol (GSAKMP) [10], Group Key Distribution Protocol (GKDP) [12] and Multimedia Inter KEY-ing (MIKEY) [8]. GKDP is supposed to be an alternative to GSAKMP as it reuses Internet Key Exchange version 2 (IKEv2) [9] functionality. MIKEY in the other hand is designed for real-time application. GSAKMP is being proposed as the generic key management protocol although there are some limitations on what it could achieve. In the following subsection, this paper describe parts of GSAKMP that are of interest.

4.1 GSAKMP

Group Security Association Key Management Protocol (GSAKMP) [10] defines the procedure on how the GSA can be established. This is facilitated by providing policy distribution and enforcement and key management functionality for multicast security. Key management functionalities includes group establishment for access control, rekeying operation and authentication mechanism. GSAKMP is build upon Internet Security Association Key Management Protocol (ISAKMP) [5] flexible functionality for managing security association. However, according to GKDP documentation [12], the functionality of GSAKMP differs a bit from those of IKEv2 [9] code base. In any case, the Group Domain of Interpretation (DOI) [6] provides the definition for ISAKMP protocol to operate in a multicast system.

4.1.1 Group Establishment

Establishing a cryptographic group is used to support secure communication among group members. A cryptographic group according GSAKMP [10] is a set of entities sharing or desiring to share a common GSA. These entities relates to the group members in a multicast system. Group establishment requires the support of a policy token. The policy token is a data structure that contains policies that are disseminated to group members. These policy tokens require to undergo verification procedure for the purpose of proving authenticity between GCKS and the group members.

Group establishment in GSAKMP consist of three mandatory messages that is required to implement. They are request to join, key download and key download acknowledgement or failure messages. Figure 5 is the GSAKMP Ladder Diagram used to illustrate group establishment process.

![GSAKMP Ladder Diagram](Image)

The diagram consists of GCKS and the group member communicating with each other. Dotted lines represent optional feature while the rest of the arrows are mandatory feature. Lets begin with the first process that is the "Request to Join" or in short RTJ process. This process is shown in step 1 where group member sends a "Request To Join" message to GCKS to request admission. The message consists of key creation material, data freshness, an optional selection of mechanism and signature of the group member. Step 2 is the "Key Download" process. A key download message contains the identifier to the group member, freshness data, key creation material, encrypted keys and finally the encrypted policy token. The message is sent from GCKS to the group member as a response to an accepted RTJ message and containing the group cryptographic keys. Consequently, an optional step 3 is sent in an event of a rejected RTJ. This process is also known as the RTJ error process. The message however is not signed by GCKS due 2 reasons. Firstly, a signature would be meaningless to the group member since it has no knowledge of the authorized GCKS. Secondly, it could possibly lead to denial of service attack. The next process is called "Key Download Acknowledgement or Failure" which is shown in step 4. Here, "key download Ack/Failure" message is sent from the group member to GCKS. The message is signed by the group member and used as an indication of the receipt status for the previous key download message. Optionally, the Lack of Acknowledgement message as shown in step 5 can be sent to indicate an invalid or absent "Key Download Ack/Failure" message. It is a signed message and used to warn group member of imminent removal from the group if step 4 is not sent. Once these processes are completed, the group will have shared keys for use in the multicast session. At this point, the GSA is established for the group member to use.

4.1.2 Group Maintenance

In GSAKMP, the maintenance of a multicast security groups covers rekeying events, policy updates and group destruction. Rekeying event requires an update of the keys used in a multicast session. These could possibly happen in the event of the keys being compromised or the expiration of the keys. Compromised indication in multicast security is when group member join and leave the group or in the case of third party gaining unauthorized access to the keys. In the event of rekeying, GCKS must provide signed messages to the group containing the rekeying information for preventing replay attacks. GSAKMP, supports rekeying algorithm such as Logical Key Hierarchy (LKH) that is explained in section 5. During a rekey event, policies are also updated in the form of policy token and disseminated to the group members. In the case of destruction of the group, it is also accomplished using the rekeying messages. Group destruction can be used as a way of maintaining group secrecy by removing keys and re-updating them.

4.1.3 GSAKMP State Diagram

As a reference for GSAKMP protocol implementation, GSAKMP has provided a state diagram and documented the transitions between the states. Figure 6 is an illustration of GSAKMP state diagram. The states are separated into two
sections. On the right hand side are the steps that are marked with the letter 'a' while on the left hand side are steps that are marked with the letter 'b'. Steps 'a' are related to group creation and steps 'b' are related to group joining.

The state starts from an "Idle" state. Let's begin from group creation transition that is on the right side of the diagram. Transition 1a is the create group command. At that stage, the state is waiting for GCKS event. Transition 2a is a loop-back transition in the event either receiving a bad "Request to Join", receiving a valid command to change group membership, sending compromise message for x number of times or member deregistration. Transition 3a is on receiving a valid "Request to Join" and the state changes to waiting for response from key download. Then, transition 4a happens in either the event of receiving "Ack", receiving "Nack" or caused by a timeout. Finally, transition 5a is the delete group command and returns back to the "Idle" state.

And on the left side of the diagram is the group joining transitions. It begins with transition 1b that performs the "Joining group command". The state has now changed to "waiting for group membership". Transition 2b happens when an acknowledgement is sent and then state changes to waiting for group membership event. A loopback transition 3b happens in the event of receipt of group management messages. Transition 4b returns back to "Idle" state in the event of deletion of group command or deregistration command. And finally, transition 5b is caused either by time out events, message failure or error events.

The commonly recommended algorithm to handle dynamic secrecy is Logical Key Hierarchy (LKH) [3]. The reasoning behind the recommendation is LKH allows support for secure removal, transmission and storage efficiency. Group key management protocol such as GSAKMP supports LKH for rekeying procedure. Figure 7 provides an overview of LKH. It is organized in a logical tree that is maintained by the key server. LKH presents the concept of root key and in the diagram K1 represents the root key. The root key in this case is also known as the Traffic Encryption Key (TEK). TEK is the commonly used key shared by all the users for the purpose of encrypting data traffic in the multicast group.

Let's construct the tree from the bottom up. The square denotes the user and circles for keys. Each user constructs a pair wise key with the key server and it is not shared with other users represented as the leaf nodes. In Figure 7, keys K5-K13 are considered the leaf nodes. Intermediary keys K2-K4 are also known as Key Encryption Keys (KEK) that are used to facilitate distribution of the root key. Therefore, with this organization, the user will only have its own path to the root of the tree. So, in the event of joining or removing of the group, all keys along the path have to be updated to maintain dynamic secrecy. LKH storage requirements for the user at the depth of d would be equivalent to d+1. So in the example as shown in figure 7, the user is at depth 2 that makes it 3 keys. This comprises of the TEK root key that is shared by all, intermediary KEK key and the unique pair wise key.

5.1 Joining the group

Now, let's consider a use case scenario when user 9 decides to join the group. User 9 generates a pair wise key with the key server. Then, the key server will now replace K1 and K4 with Ka and Kb. To distribute these new keys to all the user, we start from the bottom of the tree. The new key Kb is encrypted with the respective pairwise key of user 7 to 9 and distributed to these users. Then, the root key Ka will need to be encrypted with K2 for user 1 to 3, encrypted with K3 for user 4 to 6 and encrypted with Kb for user 7 to 9. It is also possible to send the intermediary keys and root key in a single message encrypted by the user pairwise key. In this way, only one large message is required as compared to the multiple transmission.
5.2 Leaving the group

Suppose that user 4 leaves the group. The key server then requires that Keys K1 and K3 changes to Kc and Kd and distributed to the remaining users. This activity is to prevent the current session from being compromised. So, from bottom up, first Kd will be encrypted by pairwise key of K9 and K10 and then multicast to user 5 and 6. This is proceeded by the next level where the new root key Kc is encrypted by each intermediary keys, K2 for user 1 to 3, Kd for user 5 and 6 and finally K4 for user 7 to 9. It is then multicasted to the whole group and decrypted accordingly.

5.3 Scalability

LKH is considered to be highly scalable and efficient algorithm for group rekeying compared to naive unicast approach [18]. According to [18], suppose N is the group size and d is the degree of the key tree, the cost of communication for LKH is \( O(\log(d) N) \) compared to the naive approach \( O(N) \). However, when dealing with very large group, LKH may not perform so well. There are other approaches that maybe well suited for that kind of setup for example One-way Function (OFT) [25] tree and periodic batch rekeying [17].

6 Conclusion

Group key management itself is a complex topic. The paper has focused mainly on the trust model and GSA as a reference for group key management. Since multicast security needs to support deployment issues, a single protocol alone so far is incapable to support the various scenarios in multicast security. GSAKMP, MIKEY and GKDP are examples of protocols developed to support the difference kind of use cases. Although policies are not mentioned much in this paper, it is also an important component of group key management as it is responsible for many decision-making roles. They are essential for deployment of secure multicast system. On the other hand, rekeying algorithms such as LKH helps in protecting group secrecy in a dynamic multicast environment.

In light of this study, multicast security still have some way to go for us to experience a full fledge secure system. Issues are complex that needs to be studied and standardized. And from there, it requires effort from the industry to implement it so that it eventually becomes a reality.

References


[19] G. Caronni, M. Waldvogel, D. Sun, B. Plattner  Efficient Security for Large Dynamic Multicast Groups  Computer Engineering and Networks Laboratory (TIK), ETH Zurich, Switzerland


[31] Internet Protocol Multicast Internetworking Technologies Handbook, Cisco