FRAMEWORK FOR SCALABLE SECURE SOURCE SPECIFIC MULTICAST

By

MURALI M. BRAHMADESAM

A THESIS PRESENTED TO THE GRADUATE SCHOOL OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE

UNIVERSITY OF FLORIDA

2002
To

My wonderful Mom & Dad
ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my advisor, Dr. Richard E. Newman, for giving me the opportunity to work on this interesting field of secure multicasting. This work would not have been possible but for his excellent advice and guidance through different stages of the thesis.

Thanks go to Dr. Randy Chow and Dr. Jonathan Liu for their suggestions during several research meetings.

Thanks go to Mr. Rahul Shah and Mr. Chirayu Shah of Protocols group, Ericsson IP Infrastructure, Rockville, MD, for their valuable guidance in multicast protocols development and deployment during my Internship in summer 2001.

Special thanks go to the people involved in IETF, with whom I have interacted regarding multicast protocols. But for their dedication, the Internet would not have grown this huge or robust.

Finally I would like to thank my parents for their continued and encouraging support throughout my period of study here.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACKNOWLEDGMENTS</td>
<td>iv</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>vii</td>
</tr>
<tr>
<td>KEY TO ABBREVIATIONS</td>
<td>viii</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td>x</td>
</tr>
<tr>
<td>CHAPTER</td>
<td></td>
</tr>
<tr>
<td>1 INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>2 KEY MANAGEMENT REQUIREMENTS FOR MULTICAST APPLICATIONS</td>
<td>3</td>
</tr>
<tr>
<td>3 MULTICAST ROUTING ARCHITECTURE</td>
<td>5</td>
</tr>
<tr>
<td>3.1 IGMPv3</td>
<td>5</td>
</tr>
<tr>
<td>3.2 PIM-SM</td>
<td>6</td>
</tr>
<tr>
<td>3.3 Summary</td>
<td>8</td>
</tr>
<tr>
<td>4 MULTICAST KEY DISTRIBUTION: EXISTING SOLUTIONS</td>
<td>9</td>
</tr>
<tr>
<td>4.1 Iolus Framework</td>
<td>9</td>
</tr>
<tr>
<td>4.1.1 Group Initialization and Member Addition</td>
<td>9</td>
</tr>
<tr>
<td>4.1.2 Member Deletion</td>
<td>9</td>
</tr>
<tr>
<td>4.1.3 Data Transmission</td>
<td>10</td>
</tr>
<tr>
<td>4.1.4 Merits of the Iolus Framework</td>
<td>10</td>
</tr>
<tr>
<td>4.1.5 Limitations of the Iolus Framework</td>
<td>10</td>
</tr>
<tr>
<td>4.1.6 Summary</td>
<td>12</td>
</tr>
<tr>
<td>4.2 Group Key Management Architecture (GKMA)</td>
<td>12</td>
</tr>
<tr>
<td>4.2.1 Overview</td>
<td>12</td>
</tr>
<tr>
<td>4.2.2 Detailed Description</td>
<td>13</td>
</tr>
<tr>
<td>4.2.3 Merits of GKMA</td>
<td>16</td>
</tr>
<tr>
<td>4.2.4 Limitations of GKMA</td>
<td>16</td>
</tr>
<tr>
<td>4.3 General Limitations of any GKMP</td>
<td>17</td>
</tr>
<tr>
<td>4.3.1 Group Secrets</td>
<td>17</td>
</tr>
<tr>
<td>4.3.2 Policy Complexity</td>
<td>17</td>
</tr>
<tr>
<td>4.4 Summary</td>
<td>18</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>An example of a group-shared and source-specific tree using PIM.</td>
<td>8</td>
</tr>
<tr>
<td>4.1</td>
<td>Data transmission in Iolus</td>
<td>11</td>
</tr>
<tr>
<td>4.2</td>
<td>Group Security Association Model</td>
<td>14</td>
</tr>
<tr>
<td>5.1</td>
<td>Key Update in a group of size 8 for departure of $c_5$</td>
<td>21</td>
</tr>
<tr>
<td>5.2</td>
<td>Example of departure of multiple members</td>
<td>21</td>
</tr>
<tr>
<td>6.1</td>
<td>Pragmatic Framework for Scalable Secure Multicast</td>
<td>27</td>
</tr>
</tbody>
</table>
KEY TO ABBREVIATIONS

ACL: Access Control List
CBT: Core Based Tree
DR: Designated Router
DSP: Data Security Protocol
DVMRP: Distance Vector Multicast Routing Protocol
GCKS: Group Controller / Key Server
GKMA: Group Key Management Architecture
GKMP: Group Key Management Protocol
GSA: Group Security Agents
GSC: Group Security Controller
GSI: Group Security Intermediary
IGMP: Internet Group Membership Protocol
IP: Internet Protocol
IPSEC: IP Security Protocol
KEK: Key Encrypting Key
LKS: Local Key Server
PIM-SM: Protocol Independent Multicast - Sparse Mode
RKP: Re-Key Protocol
RP: Registration Protocol
RP: Rendezvous Point
RSPT: Reverse Shortest Path Tree
SA: Security Association
SDP: Session Description Protocol
SPI: Security Parameter Index
SPKI: Simple Public Key Infrastructure
SSM: Single Source Multicast
TEK: Traffic Encrypting Key
Abstract of Thesis Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Master of Science

FRAMEWORK FOR SCALABLE SECURE SOURCE SPECIFIC MULTICAST

By

Murali M. Brahmadesam

December 2002

Chair: Richard E. Newman
Major Department: Computer and Information Science and Engineering

Multicast is a mode of message delivery in which a single message is delivered
to a group of recipients. Multicast delivery minimizes the demand on the sender
and network resources. With the standardization of several multicast routing
and group membership protocols, the task of multicast packet delivery is fairly
complete. Issues like group access control, source authentication, and secrecy have
been the major hurdles in world-wide deployment of multicast.

Secure multicast is the term used to indicate a multicast service providing
the aforementioned security features. Secure multicast is generally achieved by
providing a symmetric cryptographic key called a session key to the group members
and by sending the group data encrypted using the session key. Thus only the
legitimate group members will be able to decipher the group data.

The task of providing secure multicast is divided into two major subtasks,
namely, secure key distribution, and secure key management. Secure key distri-
bution involves distributing the session key securely to legitimate group members.
Secure key management involves generating the session key and facilitating key
updates in an efficient manner.
This thesis discusses various existing techniques to provide secure key distribution and management. Scalability is a key issue to enable large scale deployment of secure multicast. A special category of multicast, namely, Source Specific Multicast (SSM), is considered. In SSM the multicast group contains a single source and the remaining members are just receivers. A framework for scalable secure source specific multicast is proposed with the key distribution scheme providing a tree based solution where the group membership changes have only a local effect, and the key management scheme solving the local membership changes efficiently with the help of a local key server.
CHAPTER 1
INTRODUCTION

Multicasting enables group communication by delivering a single message sent to multiple receivers. Applications such as audio and video distribution, and several other “push” applications are the driving forces for multicast deployment. Reliability (timely and orderly delivery of messages) and security (group access control, source authentication and secrecy) have been the tallest hurdles in the path of commercial utilization of multicast applications.

Multicast packet delivery is a connectionless delivery scheme, hence its use in applications is done trusting that the source of packets is from the actual sender. Since it is not mandatory that only a member of a multicast group should send packets to a multicast group, there is no easy way to authenticate the packets delivered using ordinary multicast. Also, some applications require secrecy, e.g., a highly confidential video conference. Moreover, access control might be desired in some applications, including confidential collaboration and cases in which the service provider intends to charge the customer for content delivery.

Secure multicast is the term used to indicate a multicast service providing the aforementioned security features. Secure multicast is generally achieved by providing a symmetric cryptographic key called a session key to the group members, and by sending the group data encrypted using the session key. Thus only the legitimate group members will be able to decipher the group data. The essential key management requirements for multicast applications is explained in chapter 2. The task of providing secure multicast is divided into two major subtasks: secure key distribution and secure key management. Secure key distribution involves distributing the session key securely to legitimate group members. Secure key
management involves generating the session key and facilitating key updates in an efficient manner. This thesis explains the important contributions that have been made in the field of secure multicast in chapters 4, and 5.

Multicast packet delivery is based on the multicast delivery tree formed by the multicast routing protocol. Hosts and routers use a group management protocol to manage group membership information. Brief tutorials of the widely implemented multicast routing protocol, Protocol Independent Multicast - Sparse Mode (PIM-SM) [1], and the group management protocol, Internet Group Management Protocol version 3 (IGMPv3) [2], are provided in chapter 3 to draw a roadmap along which a deployable solution for secure multicast may be achieved. The proposed pragmatic framework for scalable secure source specific multicast is explained in chapter 6, and the possible future work and extensions, and conclusions are provided in chapter 7 concludes the thesis.
CHAPTER 2
KEY MANAGEMENT REQUIREMENTS FOR MULTICAST APPLICATIONS

Multicast applications use a Group Key Management Protocol (GKMP) to provide the legitimate members with the up-to-date cryptographic state they need for their secrecy and authenticity requirements. The following is the list of requirements for such a key management solution for multicast applications:

- **Join Secrecy**: this ensures that the new member cannot decipher past messages, thus mandates that a new session key be used for each member addition;

- **Leave Secrecy**: this ensures that an old member cannot decipher future messages, thus mandates that a new session key be used for each member deletion;

- **Source Authentication**: this enables each receiver to verify the authenticity of origin of each message;

- **Access Control**: this ensures that only legitimate members can access the group data;

- **Member de-registration**: this is essential for most commercial multicast applications, in which member de-registration is used for billing information; leave secrecy has to be maintained after member de-registration.

The following is the list of essential evaluation criteria for key management solutions. A more exhaustive list can be found in Moyer et al. [3], Canetti et al. [4].

- **Number of keys with a controller**: Apart from the common session key, the group controller/key server (GCKS) maintains more keys to enable efficient key updates;

- **Number of keys with each group member**: Apart from the common session key, each member maintains more keys to enable efficient key updates;
• Communication complexity for a join: This is an estimate of number of messages that have to be sent by the GCKS for each member join;

• Communication complexity for a leave: This is an estimate of number of messages that have to be sent by the GCKS for each member removal;

• Computation complexity for a join: This is an estimate of number of cryptographic operations that have to be done for each join. Separate estimates are given for GCKS and for a member.

• Computation complexity for a leave: This is an estimate of number of cryptographic operations that have to be done for each leave. Separate estimates are given for GCKS and for a member.

This list enables one to evaluate different key management and key distribution schemes for scalability.
CHAPTER 3
MULTICAST ROUTING ARCHITECTURE

Multicast packet delivery is based on the multicast delivery tree formed by a multicast routing protocol. A group management protocol is used by the hosts and the routers to manage group membership information. The following is a brief overview of IGMPv3 [2], which is the group management protocol for IPv4, and PIM-SM [1], which is the most widely implemented multicast routing protocol.

3.1 IGMPv3

Multicast addresses are class D IP addresses in IPv4. The address 224.0.0.0 is guaranteed not to be assigned to any group, and 224.0.0.1 is used as the group for all IP hosts; i.e., all the hosts and routers are members of this group. The following lists the messages sent by the hosts. All the IGMP messages are sent with TTL 1, and hence will not be forwarded outside the subnet.

- **Membership Reports**: The host can send a membership report to the router indicating the group address and/or list of the sources of interest for that multicast group. IGMPv3 adds support for source filtering; i.e., the ability for a system to report interest in receiving packets only from specific source addresses, or from all but specific source addresses, sent to a particular multicast address. The source filtering information is used by the multicast routing protocols to avoid delivering multicast packets from specific sources to networks where there are no interested receivers. Membership reports are sent either due to a membership query sent by the router or due to membership status changes in the host. A single membership report may contain the information pertaining different multicast groups, or that of just a single multicast group.

- **Leave Group**: IGMPv2 specifies a separate leave group message; in IGMPv3 a host can leave from a group by sending a membership report indicating that it excludes all the sources of a particular multicast group.

The following lists the messages sent by the routers.
• **General Query**: This is sent by a router to learn the complete multicast reception state of the hosts.

• **Group-Specific Query**: This is sent by the router to learn the reception state of the hosts with respect to a single multicast address.

• **Group-and-Source-Specific Query**: This is sent by the router to learn if any of the hosts intend to receive messages from a particular source of a particular group.

The General Query is sent periodically by the router to the multicast address 224.0.0.1, while the other two query messages are sent only to the particular multicast group address it intends to query, and are sent due to membership changes.

### 3.2 PIM-SM

The operation of PIM-SM can be explained with ease if the design considerations of this routing protocol are understood. These in turn are based on the merits and limitations of two other multicast routing protocols briefly explained below.

The Distance Vector Multicast Routing Protocol (DVMRP) is a multicast routing protocol that used a source-specific tree for data delivery. A source-specific tree is the shortest path tree built with the source for the multicast group as the root, the multicast routers as non-leaves and the members as leaves. The shortest path tree, which is a spanning tree rooted at the source, is referred to as Reverse Shortest Path Tree (RSPT), based on the method used to build such a tree. RSPTs are best suited for high data rate sources, and are to be constructed for each of the sources of the multicast group.

The Core Base Tree (CBT) is a multicast routing protocol that uses a shared tree built for the entire multicast group for data delivery. This single tree is used by all the sources of the multicast group. The shared tree is well suited for a large number of low data rate sources, and results in traffic concentration as all the packets from all the sources are delivered through the same shared tree.
PIM is designed to make use of both the shared tree and shortest path trees for data delivery. The choice of which of the trees to use depends on the data rate of the source. Initially, only the shared tree will be created for the entire group, and an RSPT will be created only if the data rate of a particular source at which the tree will be rooted exceeds a threshold. Thus creation of RSPTs is data-driven. After the creation of the RSPT for a source, this tree will be used to deliver packets from that source to the multicast group. Packets from the other sources will continue to use the shared tree of the multicast group. If the data rate of the source for the RSPT falls below the threshold, the shared tree will be used for subsequent packet delivery. PIM operation using an RSPT is referred as PIM-Dense Mode (PIM-DM), and the operation using the shared tree is called PIM-Sparse Mode (PIM-SM).

PIM-SM requires predetermined per-group Rendezvous Points (RPs) for the creation of shared trees. When a host sends the IGMP join message to its Designated Router (DR), the router sends the PIM join message towards the RP advertised for the particular group. Intermediate routers process this message, resulting in the creation of a branch from the RP to the DR used for packet delivery. A similar branch will be created between the source of the multicast group and the RP. Thus when a source sends packets at low data rates, the packets reach the RPs and are sent to the other DRs for delivery to the members. When the data rate exceeds a threshold, the creation of an RSPT rooted at the DR connected to the source is triggered at the DRs. PIM Prune messages are sent towards the RP and PIM join messages are sent towards the DRs forming the next hop of the RSPT. Figure 3.1 shows an example PIM-SM tree structure. The continuous lines represent edges of the shared tree and dashed lines represent the edges of the shortest path tree.
Figure 3.1: An example of a group-shared and source-specific tree using PIM

3.3 Summary

This chapter briefly explained IGMPv3 and PIM-SM. IGMPv3 is the group management protocol for IPv4, and is used by the routers and hosts to manage the group membership information. IGMPv3 queries and reports were explained briefly. Multicast packets are delivered using a multicast delivery tree formed by a multicast routing protocol. PIM-SM is the most widely deployed multicast routing protocol. PIM-SM uses a group shared tree for lower bit rate packet delivery, and a source specific tree for higher bit rate packet delivery.
CHAPTER 4
MULTICAST KEY DISTRIBUTION: EXISTING SOLUTIONS

This section explains important architectural frameworks for secure multicast. These architectural frameworks enable the session key to be distributed securely to legitimate group members. Existing solutions are also analyzed for ease of deployment.

4.1 Iolus Framework

Iolus [5] partitions the multicast group into a hierarchy of subgroups, each with relatively few members and its own separate multicast address. The tree is comprised of Group Security Agents (GSAs). The root of the tree is called the Group Security Controller (GSC), and the other GSAs are called Group Security Intermediaries (GSIs). The GSI of a subgroup manages the subgroup’s keys and mediates all the communication between its subgroups and other subgroups.

4.1.1 Group Initialization and Member Addition

At first, only the GSC exists, and all the GSIs are aware of the GSC’s unicast address. When there is a request to a GSA from an outsider to join the group, after checking with its Access Control List (ACL), it generates a new session key, sends it via secure unicast to the new member, and encrypted with the old key, it is multicast to the existing members. If the GSA is the GSI, and if the current request is the first join request, then the GSI joins a higher level multicast group, proceeding until a subtree is formed with GSC. The GSIs are supplied with the ACLs to process joins locally.

4.1.2 Member Deletion

On member deletion, the local GSI generates a new session key and gives it to the remaining $n$ members in the subgroup separately via secure unicast. Even
though the time complexity for this is $O(n)$ it is not bad, as the size of the entire multicast group can be much larger than $n$. If this removal results in no member for the subgroup, then the GSI leaves the tree.

4.1.3 Data Transmission

The source multicasts directly within its subgroup, and each GSI remulticasts to the other subgroup in which it is a member. Since the other subgroup uses a different session key, the GSI must decrypt the data with the key of one subgroup and encrypt with the key of the other subgroup before multicasting. Instead of doing this for the entire message, the source multicasts $\{K_{\text{rand}}\}K_{sk1}\{\text{Data}\}K_{\text{rand}}$, where $K_{\text{rand}}$ is the random key generated by the source for each message, and $K_{sk1}$ is the session key of the subgroup. The GSI of the subgroup decrypts $\{K_{\text{rand}}\}K_{sk1}$ and remulticasts $\{K_{\text{rand}}\}K_{sk2}\{\text{Data}\}K_{\text{rand}}$ to the other subgroup with $K_{sk2}$ as its session key. Thus, only the random key has to be encrypted each time with a different key. Figure 4.1 shows data transmission from a source in subgroup $G^{2A}$ to the entire group.

4.1.4 Merits of the Iolus Framework

The advantages of the Iolus framework are listed below.

- This architecture localizes the effect of group membership changes to one sub-group.

- It distributes the security over many nodes, eliminating any single point of failure except the top-level GSC.

- The use of the local session keys as Key Encrypting Keys (KEKs) in data delivery removes the burden of decrypting and encrypting the entire data payload at each GSA.

4.1.5 Limitations of the Iolus Framework

The drawbacks of the Iolus framework are listed below.

- The GSC still remains as a single point of failure.
Re-keying on member departure takes $O(n)$ messages, where $n$ is the number of remaining members in the subgroup. Even though this can be much smaller than the entire group size, there must be numerous GSIs to reduce the number of members in each subgroup.

The framework fails to mention where the GSIs have to be placed in the actual multicast delivery tree created by multicast routing protocols. This is very important because the semantics of operations become blurred if the GSIs can be outside the subnets and away from the actual delivery tree created by the multicast routing protocols. In this case the GSIs have to become the members of the global multicast tree but the routing protocols have to be changed so that the packets are not sent directly to the members but only through the GSIs.

Each GSI has to be a member of a separate subgroup. When an outsider intends to become a member of a global group, he has to join a local subgroup. The mapping between the local subgroup and the global group has to be defined.
• The requirement for different subgroups results in use of multiple multicast addresses for the single global multicast group. Since no routing protocols are designed this way, GSIs have to take responsibility for routing packets.

4.1.6 Summary

Iolus is the multicast key distribution framework which is highly scalable. The components and operation of Iolus is explained. The evaluation of Iolus with deployment as the core evaluation criteria is done and it indicates that this architecture cannot co-exist with the already deployed multicast routing protocols like PIM-SM.

4.2 Group Key Management Architecture (GKMA)

GKMA [6] intends to define a common architecture and design for different Group Key Management Protocols (GKMPs) for internet, transport, and application services. GKMP enables only the members (senders and receivers) of a secure multicast group to gain access and authenticate the group data. This design is based on a group controller model with a single group owner as the root of trust. The group owner designates a group controller for member registration and re-key.

4.2.1 Overview

The GKMP provides the group members with an up-to-date Data Security Protocol Security Association (SA) or Data SA, which contains the information necessary to secure the group data. The contents of the Data SA is given in 4.2.2. To achieve this goal, the GKMA consists of the following protocols:

• Registration Protocol;

• Re-key Protocol;

• Data Security Protocol.

Registration protocol (RP). The RP is a two-way unicast protocol between the Group Controller/Key Server (GCKS) and a joining member. The RP uses the Registration Protocol SA to ensure that the transfer of information from GCKS to
member is done in an authenticated and confidential manner. The RP facilitates mutual authentication between GCKS and a joining member, and allows verification of authorization of a joining member.

Upon successful authentication and authorization verification, the GCKS provides the joining member with sufficient information to initialize a Re-key Protocol SA with the joining member if the Group Security Policy (GSP) calls for it, and sufficient information to initialize a Data Security Protocol SA with the joining member.

Re-key protocol (RKP). RKP is an optional protocol and is used by GCKS periodically to send re-key information to the group members due to membership changes or expiry of the Traffic Encrypting Keys (TEK). Re-key messages are protected by the Re-key Protocol SA. The messages are either sent by multicast to group members or are unicast to a particular group member. RKP is optional as other means for managing the keys locally without interaction between the GCKS and member may be used. The re-key messages are to be delivered in a timely manner and should subjected to source authentication.

Data security protocol (DSP). DSP uses the TEK to secure the group data.

4.2.2 Detailed Description

Each group member uses the RP to obtain authorized, authenticated access to a particular group, its policies, and its keys. The two types of group keys are:

- **Key Encrypting Key** (KEK): this key is used by the re-key protocol, and is used to encrypt the TEK;

- **Text Encrypting Key** (TEK): this key is used by the Data Security Protocol (DSP) to protect the group data.

**Group Security Association (GSA).** The GCKS is a separate, logical entity that performs member authentication and authorization according to the group policy that is set by the group owner. The GCKS creates the KEK and TEK, and sends
Figure 4.2: Group Security Association Model

them to the joining member. Upon receipt of a TEK from a re-key message or a Registration Protocol exchange, the member’s group key management will provide a security association to a Data Security Protocol for the data sent from the sender to the receiver. In Figure 4.2, the Security Protocol SA protects the data sent on the arc labeled “DATA SECURITY PROTOCOL,” the Re-key SA protects the data sent on the arc labeled “RE-KEY PROTOCOL,” and the Registration Protocol SA protects the RP exchanges. These three SAs comprise the Group Security Association.

The policy and authorization infrastructures indicated in Figure 4.2 are external to GKMP. The group-policy will be distributed through both announcement and key management protocols [7]. The group-policy contains at least the following:
• group owner, authentication method, and delegation method for identifying a GCKS for the group;

• group GCKS, authentication method, and any method used for delegating other GCKSs for the group;

• group membership rules or list and authentication method.

It also contains two additional policy-related requirements external to group key management:

• authorization and authentication infrastructure such as Simple Public Key Infrastructure (SPKI)[8] or pre-shared key scheme in accordance with the group policy for a particular group;

• an announcement mechanism for secure groups and events that operates according to group policy for a particular group, (for example Session Description Protocol (SDP)).

Contents of Re-key SA. The Re-key SA protects the Re-key protocol. It contains the following:

• Re-key SA policy: the membership management algorithm that enforces forward and backward access control, the KEK encryption algorithm, the authentication algorithm, the control group address, the re-key server address;

• Group Identity: to identify the group if multiple groups can be initialized in a single invocation of the RP or RKP;

• KEKs and their lifetimes;

• GCKS authentication key: a symmetric key or public key for authentication of the re-key messages;

• sequence numbers: for replay protection;

• Security Parameter Index (SPI): the triple (Group Identity, SPI, an identifier for “Re-key SA”) that uniquely identifies an SA.

Contents of the Data SA. The Data SA protects the group data. It contains the following:

• Group Identity: to identify the group if multiple groups can be initialized in a single invocation of the RP or RKP;
• source identity: to identify the source for the group;

• traffic encrypting key;

• authentication key;

• sequence numbers: for replay protection;

• Security Parameter Index (SPI): the triple (Group identity, SPI, an identifier for “Re-key SA”) that uniquely identifies an SA;

• Data SA policy.

The Data SA policy includes encryption algorithm and parameters, the source authentication algorithm and parameters, the group authentication algorithm and parameters, and/or replay protection information.

4.2.3 Merits of GKMA

The implementation of the protocols is not dependent on the architecture, allowing the requirements of the application to dictate the protocol operation. For example, a re-key protocol for a small group could use multiple unicast transmissions with symmetric authentication, while that for a large group could use IP Multicast with packet-level forward error correction and source authentication. There is no need for a unicast exchange to provide data keys to existing members of the group. Data keys can be pushed in the Re-key Protocol. This allows fast rekeying for existing members. The authorization infrastructure can support delegation, as does SPKI. This hierarchically-organized key distribution with multiple GCKSs can handle “flash-crowds.”

4.2.4 Limitations of GKMA

Handling departing members. The following points list the limitations of GKMA in the way it handles departing members.

• If the departing member fails to report to the GCKS that it is no longer in the group, the GCKS must wait until the SAs time out before freeing up the associated data structures and timers.
• The timeouts for the SA expiry cannot be very small as this would pose considerable overhead for the GCKS, since it usually handles SAs of numerous members. If the group is for a paid service, the GCKS uses the time of SA removal for billing information. If the member fails to report its departure to GCKS the member will be expected to pay until the SA expires, even though it was not a member for the intervening time.

• In large-scale multicast applications, de-registration is an essential service (e.g., for billing information), and thus has the potential to cause implosion at the GCKS.

Handling flash crowds. Even though GKMA specification indicates that multiple GCKSs can be used by delegation to handle flash crowds, it lacks the means to handle a flood of departing members (discussed above) in a satisfactory way. Moreover, additional protocols are required between the different GCKSs to generate the same keys on member joins or leaves based on the group policy as the source will be using a single key to encrypt group data.

Dynamic groups. Dynamic groups involve multiple join and leave requests over a short period of time. A single GCKS cannot handle this as key generation requires a considerable amount of time. This delay may result in more leave requests worsening the situation. In the worst case scenario, the source ends up waiting for the key from the GCKS rather than sending group data.

4.3 General Limitations of any GKMP

This section discusses the general limitations of any GKMP [6].

4.3.1 Group Secrets

Group members are trusted to preserve the group secrets. It is very difficult to find if a legitimate member has disclosed the group secrets to an outsider, and if such an outsider is listening to the group data. Moreover, authentication by symmetric keys should be avoided unless all the group members can be trusted.

4.3.2 Policy Complexity

Unlike peer-to-peer security policy, which is an intersection of the policy of the individual peers, a group owner sets the group policy externally. The choice of the
group policy poses new risks to the members of the group. The group owner and members need to determine minimal acceptable levels of trust, authenticity and confidentiality for the group.

4.4 Summary

GKMA defines a common architecture and design for different GKMPs. The components and operation of GKMA is explained. The evaluation of GKMA with deployment as the core evaluation criteria indicates that it does not handle bursty and flash crowds for dynamic multicast groups. General limitations of any GKMP with respect to group secrets and policy complexity is explained.
CHAPTER 5
MULTICAST KEY MANAGEMENT: EXISTING SOLUTIONS

In secure multicast, join and leave secrecy must be maintained, which involves generating a new session key used for subsequent encryption of the group data, and distributing the new session key securely to the remaining members. Secure key management involves generating the session key and facilitating key updates in an efficient manner. In this section a naive key management scheme is explained first to emphasize the scalability requirements of the key management schemes. A detailed explanation of the tree-based solution using Boolean function minimization is given. Detailed explanations of other tree based solutions are covered in a previous survey of security issues in multicast communications [3].

Member de-registration, as explained earlier, involves preserving leave secrecy. Since most multicast applications involve bursty joins and leaves, it is essential to provide cumulative member removal for scalability reasons. The tree-based technique using Boolean function minimization provides such a solution. Other key management solutions are explained in Wallner et al. [9].

5.1 Naive Key Management

Here, the GCKS stores \( n + 1 \) keys: the common session key and one for each of the \( n \) members. For each join or leave, the GCKS securely unicasts the new group key to each member, so the computation and communication complexity in GKC are each \( O(n) \). Clearly, this scheme does not scale well for applications with numerous members and involving bursty joins and leaves.

5.2 Tree Based Solution Using Boolean Function Minimization

In the tree-based solution using Boolean function minimization [10], the GCKS maintains a complete balanced binary tree where the \( N \) leaves are attached to
another $N$ nodes representing the $N$ members of the group. Each of the $N$
members is given an unique UID represented by a binary string of length
$n = \lceil \log_2 N \rceil$. Each nonleaf node in the tree contains an auxiliary key. A member
with UID $X_{n-1}X_{n-2}...X_0$, receives the common session key $SK$, and the set of $n$
auxiliary keys $K_{n-1}, K_{n-2}, ..., K_0$, where $K_i$ is $k_i$ if $X_i = 1$, and $k'_i$ if $X_i = 0$. The
auxiliary keys are used to update the session key in secure fashion. For example,
in Figure 5.1, the member $c_0$ with UID 000 will have $SK$, $k'_0$, $k_1$, $k'_2$, and the
member $c_6$ will have $SK$, $k_2$, $k_1$, $k'_0$. So the GCKS maintains $2 \log_2 N$ auxiliary keys,
and each member will maintain $\log_2 N$ auxiliary keys.

5.2.1 Individual Member Removal

For each member removal the GCKS sends $\log_2 N$ messages. For example, to
remove a member $c_5$ with UID 101, the GCKS sends the new session key $SK_{\text{new}}$
encrypted with $k'_0, k_1, k'_2$. $c_5$ cannot decrypt any of these three messages as it has
only $k_0, k'_1$, and $k_2$. Figure 5.1 shows a visual interpretation of this re-keying
scheme. In the figure, the keys corresponding to the solid round nodes correspond
to the keys possessed by the departing member $c_5$. The hatched round nodes
represent the complementary set, that is, the keys not possessed by $c_5$. Since the
leaving member could listen to future key updates, the auxiliary keys are also
updated. To update a key $K_i$, a one way hash function $f$ is used that yields the
updated auxiliary key as follows $K_{i,\text{new}} = f(K_i, SK_{\text{new}})$. Since the leaving
member does not have the new session key $SK$, it will not be able to compute the
new auxiliary key.

5.2.2 Removal of Multiple Members

An efficient way to remove multiple members uses Boolean function
minimization. For example, if members $c_0$ and $c_4$ with UIDs 000 and 100,
respectively, are to be removed from the group, the new key $SK_{\text{new}}$ has to be given
securely to $c_1, c_2, c_3, c_5, c_6, c_7$. It is enough if GCKS sends $\{SK_{\text{new}}\}k_0, \{SK_{\text{new}}\}k_1$. 
The first message can be decrypted only by $c_1, c_3, c_5,$ and $c_7,$ and the second only by members $c_2, c_3, c_6,$ and $c_7$. Thus the remaining members can be updated with the new key using only two messages instead of six messages. Figure 5.2 depicts this scenario.

**Boolean function minimization.** Cumulative member removal is equivalent to grouping the remaining members that share common bits in the UID, and hence common keys, which are different from those possessed by the removed members. This problem is equivalent to Boolean function minimization. Methods for
minimizing the Boolean function can be used to find the minimum number of messages to be sent.

5.2.3 Collusion Attacks

In collusion attack, a set of removed members collude and by combining their sets of keys, and may be able to obtain the currently valid set of keys. This will enable them to listen to the group communication. It is impossible to eliminate the risk of a collusion attack with less than \(O(N)\) auxiliary keys [10]. The risk of collusion attack may be reduced with a large auxiliary key space and a sparse distribution of UIDs.

5.2.4 Performance Analysis

Detailed proof for the following performance analysis results is given in Chang et al. [10].

Worst case performance. Re-keying a secure multicast group of size \(2^n\) when two group members are to be removed requires at most \(n\) messages, and when \(2^{n-1}\) members are to be removed, it requires at most \(2^{n-1}\) messages.

Average case performance. The average number of messages required for aggregate removal of an arbitrary number of members from a multicast group is equivalent to average number of products in the minimum sum-of-products expressions of Boolean functions. Extensive proof of this is given in [10].

5.3 Summary

Secure key management involves in generating the session key and facilitating key updates in an efficient manner. It also helps in maintaining join and leave secrecy. A naive key management technique is explained to provide an insight on the complexity requirements of secure key management. A tree based solution using Boolean function minimization is explained as it solves the cumulative member removal in an efficient manner. Cumulative member removal is the most desirable feature for large dynamic groups in which the leaves are bursty. Other
operations are also done in logarithmic time complexity with fewer number of keys to be maintained in the GCKS and the members compared to other key management techniques.
The previous discussions of key distribution and key management were focussed mainly on scalability. Given the merits and limitations of various schemes, and with the multicast routing architecture in mind, this section proposes a framework for scalable key distribution and management for secure multicast.

6.1 Design Features and Requirements

Scalability is the driving force behind this framework. Much emphasis is given for the framework to be pragmatic by considering only the widely deployed protocols for routing and membership. The design features and requirements are given below.

**Scalability.** Core ideas from Iolus and GKMP, discussed in section 4 have been used in the proposed key distribution framework. The key management scheme using Boolean function minimization discussed in section 5.2 is used as it has many desirable features such as cummulative member removal, and better complexity than other schemes.

**Ease of deployment.** The framework is designed considering the working modes of widely deployed protocols such as PIM-SM and IGMPv3. The entire framework becomes more scalable by using the messages of these protocols. The framework is free of the limitations listed earlier in subsection 4.2.4.

**Why source specific multicast?** In source specific multicast [11] [12], there is only one source $S$, for the entire group $G$. The group data sent by the source to the group is said to form a stream/channel identified by the $(S,G)$ pair. The other group members can become subscribers of this channel by using IGMPv3 [2] for
IPv4 and MLDv2 [13] for IPv6 protocols. The multicast delivery tree will be built by PIM-SM [1], the most widely deployed multicast routing protocol. Based on the previous discussion of PIM-SM, SSM mostly results in using the source specific tree created for each (S,G). Any source multicast (ASM) is not considered in this proposal, where any member can be a source for the group. The semantics of operation of secure ASM is very much different from that of secure SSM. The presence of a single source to the group in SSM suits well with many of the applications having a potential to have multiple receivers and requiring access control.

6.2  Key Distribution

6.2.1  Framework

Large dynamic groups do not scale well if a single key server is used. Dynamic here denotes that the there is no restriction for the group members to join at a particular time of the group existence or to remain in the group for a period of time. Dynamic groups usually have bursty joins and leaves. Multicast groups are often denoted by a pair (*, G) or by (S, G). Here * represents any source for the multicast group G, and S represents a single source S for the multicast group G. A member of a multicast group G listening to the messages sent by a source S is said to be a subscriber of the (S, G) channel.

In this framework, a secure distribution tree like that of Iolus is used. The central GCKS may be replicated, and in such a case an election algorithm is used to select a primary GCKS. The election algorithm may be simple: the GCKS with the smallest IP address value is selected as the primary. The local key servers (LKSs) are sent the list of IP addresses of the GCKSs. Each LKS can pick a random GCKS and communicate with it. The LKS has to be present in the listening zone of IGMPv3. Listening zone here means that the LKS should be able to listen to the IGMPv3 messages sent by the members, outsiders, and by the
multicast-aware routers. This enables an LKS to find the presence or absence of
group members subscribing to an \((S, G)\) channel immediately. There can be
multiple LKSs within the listening zone, and an LKS can also be present as a part
of the multicast aware router. Since the LKSs are present in each of the listening
zones, the local changes in membership affect only the local session key and not the
global session key. The LKS here does the same operation as that of GSI in Iolus.
The list of LKSs is sent to the joining member after initial IGMPv3 join request,
then the joining member picks a random LKS from the list of LKSs sent by the
primary LKS and communicates with it. Figure 6.1 shows an example of the
proposed framework. In the figure, solid lines represent the group shared tree
created by PIM for a multicast group, the dashed lines represent the source-specific
tree created by PIM for a particular source S for a multicast group, and the dotted
lines represent the secure unicast connections established between the LKS and the
GCKS. The thatched circles represent the primary LKS/GCKS elected by an
election algorithm.

6.2.2 Operational Overview

The detailed description of the proposed framework is given below.

**Startup.** The framework requires the GCKS(s) to be available when the group is
started. The list of GCKSs, public key of GCKS (a single key is used by all the
GCKSs), the address of source, and the group address are distributed using a
Session Description Protocol (SDP). The GCKS maintains the ACL, which is used
to provide access control for the group data. Only those members who satisfy the
access control procedure will be provided the group key and hence will be able to
listen to the group data.

**Joins.** An outsider sends a IGMPv3 join message indicating the pair \((S, G)\) to its
multicast-aware router. The router uses PIM-SM to join the distribution tree
corresponding to \((S, G)\). The primary LKS listening to the IGMPv3 messages will
Figure 6.1: Pragmatic Framework for Scalable Secure Multicast

send a list of LKSs to the sender of the join message. The member picks a random LKS and establishes secure unicast channel with that LKS using protocols such as the IP Security Protocol (IPSEC). The member now uses this secure channel to send the required information to satisfy the access control procedure dictated by the group policy. For example, the message can be \{<username>, <password>\} \textit{PuK}_{GCKS}, where the \(<username>\), and \(<password>\) can be used by the GCKS to provide access to the group data. Since the LKS has no means to verify access control (dictated by the group policy), and it is not realistic to expect GCKS to provide such ACL related information to every LKS, the member encrypts this information with the public key of the GCKS. This ensures that only the GCKS
can decrypt this information as it alone has its private key which can be used to decrypt it.\(^1\)

The LKS uses the list of GCKSs sent via SDP, picks a random GCKS, and establishes a secure unicast channel using IPSEC. The LKS uses this secure channel to send the encrypted information sent by the (possible) member. The GCKS verifies and sends OK/NOK message to LKS. If the returned message is OK then the LKS now starts key management service and provides the member with the secure key. If not, the failure is indicated to the rejected member and the SAs corresponding to this sender are removed.

The LKS need not send the information sent by the possible member to GCKS immediately; rather it may collect many such requests to join, and send them as a bulk request to the GCKS. In this case the LKS assigns a nonce\(^2\) as the packet ID, and the GCKS replies with the same packet ID with OKs/NOKs in the same order as that of the corresponding request, so that the LKS can match the position of the requests it sent with that of the replies it has received and take appropriate actions. Usually the joins are bursty and hence this technique reduces the burden on LKS and GCKS. If the current request sent by the possible member is the first one, then the LKS establishes a secure unicast channel with a randomly picked GCKS and uses it for all future communication.

**Leaves.** When a member sends a IGMPv3 leave message indicating \((S, G)\) channel, the primary LKS listening to this message immediately removes the SA corresponding to this member and sends a member leave request to the GCKS,

\(^1\) In the private, public cryptographic key pair, if one is used for encryption the other one can be used for decryption

\(^2\) nonce denotes the non-repeating, random sequence used in secure communication to distinguish different messages
which in turn uses it for accounting purposes. The LKS also starts the key management service to provide the remaining members with the new session key.

The router processing the IGMPv3 leave messages sends messages to find if there are any other listeners for this \((S,G)\) channel, and if there is no response, removes itself from the global distribution tree using PIM-SM. The LKS listening to the IGMPv3 messages removes any remaining SAs for this \((S,G)\) channel, sends the list of members it has not heard from, and also indicates to the GCKS that it is no more in the key distribution tree as there are no more members it has to serve. Thus this technique enables the GCKS to compile nearly accurate billing information (it is mentioned as nearly accurate as IGMPv3 source- and group-specific queries are sent, and the conclusion that no member is present subscribing to this channel is deduced only after a timeout for getting a IGMPv3 report back from at least one member subscribing to this channel).

6.2.3 Data Transmission

The data transmission scheme is adopted from the Iolus scheme. The senders send the messages in this format: \(\{K_{\text{rand}}\}K_{LKS1}\{\text{Data}\}\{K_{\text{rand}}\}\), where \(K_{\text{rand}}\) is the random key generated by the sender, and \(K_{LKS1}\) is the local session key. The LKS decrypts and encrypts random key with next level of local session key. The message now will be: \(\{K_{\text{rand}}\}K_{LKS2}\{\text{Data}\}\{K_{\text{rand}}\}\). This technique reduces the necessity of decrypting and encrypting the whole message each time with the new key. The group members will decrypt the data after getting the random key using the local key.

The multicast-aware router does the decryption and encryption of the key if the LKS is a part of the router itself. If the LKS is a separate entity, then the multicast router sends the packet as a unicast to the primary LKS and the LKS multicasts the packet after decrypting and re-encrypting the random key with the local session key. The router, instead of sending the packet as multicast, uses the
link address of the primary LKS, so that LKS alone processes it before remulticasting.

6.3 Key Management

Key Management for the entire multicast group is subdivided into local key management tasks done by LKS to manage the actual members, and the key management task done by GCKS to manage the global session key distributed to the LKSs. The tree-based key management scheme using Boolean function minimization technique (see section 5.2) may be used as it has the following desirable features.

- **Cumulative member removal**: this is the most desirable feature as member de-registration in most dynamic multicast groups tends to be bursty.

- **Number of keys maintained in the LKS**: this measure is \((2\log_2 N) + 1\) where \(N\) is the number of members within LKS’s listening zone, which is comparatively fewer than the number that have to be maintained in other techniques.

- **Number of keys maintained in the member**: this measure is \((\log_2 N) + 1\) where \(N\) is the number of members within LKS’s listening zone, and is comparatively fewer than the number that have to be maintained in other techniques.

- **Better overall communication and computation complexities**: For \(N\) members in the listening zone of an LKS, the single member join or leave takes \(O(\log_2 N)\) computation and communication complexities, but cumulative member removal uses Boolean function minimization and results in fewer messages and hence fewer computations to be done by LKS and by the members.

The GCKS also uses Boolean function minimization to maintain the global session key, and distributes it to the LKSs. The GCKS need not be present along the global data distribution tree built by the multicast routing protocol PIM-SM; rather, it may be present as a separate entity visible to the LKSs. Thus the GCKS maintains \(2\log_2 N + 1\) keys, where \(N\) is the number of LKSs. The LKSs have to maintain \(\log_2 N + 1\) keys for this purpose, where \(N\) is the number of LKSs. This set
of keys is in addition to the set of keys maintained by each of the LKSs to manage keys for the actual members in their listening zones.

The analysis of the key management technique using Boolean function minimization indicates that the single member removal requires $O(\log_2 N)$ computation and communication complexities for $N$ members. A technique to reduce this logarithmic complexity to a constant complexity is proposed and is explained in section 6.3.1.

6.3.1 Detailed Description

The key management scheme proposed here is an improvement to the scheme discussed in 5.2. Here also the LKS maintains a key distribution tree, with the $N$ members attached to $N$ leaves. Each member is given an unique binary string representing its UID. The members have the $n = \log_2 N$ keys corresponding to the bits in their UIDs.

All the messages sent by the LKS use IPSEC, so that they can be authenticated. In this proposal, the LKS sends the new session key encrypted with an auxiliary key, and the members send the new session key encrypted by other auxiliary keys according to the algorithm given below. The auxiliary keys are updated locally by each of the member as in section 5.2.

6.3.2 Member Join

When a new member is added, the new member is given the new session key, and auxiliary keys are sent via secure unicast channel established between the LKS and the member. The new session key with the UID of the new member is sent in a single multicast packet encrypted with the old session key. All the existing members will receive the new session keys and will update the auxiliary keys according to the UID of the new member.
6.3.3 Individual Member deletion

In this proposal, responsibility of multicasting the new session key is shared with the members. Thus in the best case, the LKS sends only one key update message, and each of the members will send at most one key update message to the group. For example in Figure 5.1, for the departure of $c_5$, the LKS sends

$$\{UID(c_5)\{SK_{new}\}PrK_{LKS}\}k'_2,$$

where $UID(c_5) = 101$ is the UID of the member $c_5$, $SK_{new}$ is the new session key, $PrK_{LKS}$ is the private key of LKS, and $k'_2$ is the auxiliary corresponding the bit $X_2 = 0$ in the UID. Thus, with this message, $c_0, c_1, c_2, c_3$ can update their session and auxiliary keys. From the $UID(c_5) = 101$, it is known that one multicast with $k_1$, where $k_1$ is the auxiliary corresponding the bit $X_1 = 1$ in the UID, another two members can receive the session key. Both $c_0, c_1$ does not have $k_1$, but $c_2, c_3$ have $k_1$, so each of them picks a random time and backs off and when the timer expires sends $\{UID(c_5)\{SK_{new}\}PrK_{LKS}\}k_1$. If $c_2$ sends this message, $c_3$ does not send this message again. Since the new session key is signed by the private key of LKS, it is authentic. With this message $c_6, c_7$ receives the new session key. Again from the $UID(c_5) = 101$, it is known that one multicast with $k'_0$, where $k'_0$ is the auxiliary corresponding the bit $X_0 = 0$ in the UID, the remaining member can receive the session key. $c_0, c_2, c_6$ already have the new session key. Each of them picks a random time and backs off, and when the timer expires sends $\{UID(c_5)\{SK_{new}\}PrK_{LKS}\}k'_0$. If $c_0$ sends first $c_2, c_6$ refrain from sending the same message. Thus $c_4$ receives the new session key. Each member calculates the new auxiliary keys locally.

The key update messages are sent as multicast, so the LKS will also receive the key update messages. If there are no messages sent by the members, then the LKS has to traverse the tree and send the new session key encrypted with the key not held by the member being removed. Thus in the worst case LKS will be sending $O(\log_2 N)$ messages.
6.3.4 Cumulative Member Deletion

The technique using Boolean function minimization 5.2 results in minimal number of messages to be sent for cumulative member deletion. Thus this technique is used in the proposed framework.

6.3.5 Tree Growth

When a member join results in increase of the height of the KM tree in LKS by one, the LKS sends the growth message to the existing members with the a new auxiliary key which is common to all the members. The message is \( \{K_{newaux}\}SK \), where \( K_{newaux} \) is the new auxiliary key, and SK is the existing common session key. This message will also cause all the members to change their UID by prefixing a '0' to their existing UID. For example, if the UID of a member is \( X_{n-1}X_{n-2}...X_0 \) will be changed to \( X_nX_{n-1}X_{n-2}...X_0 \) where \( X_n = 0 \). Clearly, \( K_{newaux} \) corresponds to \( X_n = 0 \).

6.3.6 Tree Shrinkage

When a member deletion results in only one half of the tree (either left or right) to contain members, the height of the KM tree in LKS can be reduced by one. The LKS sends the shrinkage message to the existing members indicating them to discard the key corresponding to their MSB in UID. This also results in the removal of the MSB from the UID of all the members. For example if the UID of a member is \( X_nX_{n-1}X_{n-2}...X_0 \), then after removal of the MSB the UID changes to \( X_{n-1}X_{n-2}...X_0 \). The key corresponding to \( X_n \) is also discarded.

If the tree is sparsely populated by members, the half of the tree (either left or right) with fewer members is chosen to be removed from the tree. LKS sends rejoin messages to those members and hence removes them from the tree and assigns them new UID when they rejoin. If both the halves contain equal number of members, one half is chosen randomly.
6.4 Properties

**Early Detection of member absence.** Since the LKS is present in the listening zone of IGMPv3 messages, it can detect the absence of a member and hence the corresponding SAs can be removed. This is impossible if LKS is outside the listening zone, where the SAs are removed only after timeout. Here, SAs will be removed even if the member crashes; IGMPv3 messages discover such absences.

**No access control information leak.** LKS cannot decrypt this information sent by the member to be sent to the GCKS.

**Multiple License made easy.** Since the LKS can be placed locally, an organization can obtain multiple licenses for the entire organization and the LKS need not indulge in key management services for each member join or deletion. The local key can be kept constant, or the LKS can decrypt the entire message and send to the member.

**Collusion Attacks.** Collusion attacks by members still exist as the key management technique using Boolean function minimization is used.

6.5 Summary

A pragmatic framework for scalable secure source specific multicast is proposed with scalability, and ease of deployment as the core design requirements. The proposed framework is designed to co-exist with the multicast routing architecture with IGMPv3 and PIM-SM. The key distribution architecture is designed with desirable features from Iolus and GKMA. The proposed key distribution scheme localizes the group changes and hence the global session key remains mostly constant while the local session keys may change often. The key management scheme follows the technique using Boolean function minimization. The operations for individual member deletion has been modified so that in the best case, the LKS sends only one key update message, and each of the members will send at most one key update message to the group. Cumulative member deletion is possible as
Boolean function minimization is used. The features like early detection of member absence, no access control information leak, and ease of providing multiple license enable easy deployment. Collusion attacks remain as Boolean function minimization technique is used.
CHAPTER 7
CONCLUSION AND FUTURE WORK

7.1 Conclusion

This thesis document started with a brief explanation of the multicast routing architecture comprising multicast group management and multicast routing protocols. IGMPv3 is the group management protocol for IPv4, and is used by the routers and hosts to manage the group membership information. IGMPv3 queries and reports were explained briefly. Multicast packets are delivered using a multicast delivery tree formed by a multicast routing protocol. PIM-SM is the most widely deployed multicast routing protocol. PIM-SM uses a group shared tree for lower bit rate packet delivery, and a source specific tree for higher bit rate packet delivery.

Secure multicast is the term used to indicate a multicast service providing group access control, source authentication, and secrecy. Secure multicast is generally achieved by providing a symmetric cryptographic key called a session key to the group members, and by sending the group data encrypted using the session key. Thus only the legitimate group members will be able to decipher the group data. The task of providing secure multicast is divided into two major subtasks namely secure key distribution, and secure key management.

Secure key distribution involves distributing the session key securely to legitimate group members. Iolus is the multicast key distribution framework which is highly scalable. The components and operation of Iolus is explained. The evaluation of Iolus with deployment as the core evaluation criteria is done and it indicates that this architecture cannot co-exist with the already deployed multicast routing protocols like PIM-SM. GKMA defines a common architecture and design for
different GKMPs. The components and operation of GKMA is explained. The evaluation of GKMA with deployment as the core evaluation criteria indicates that it does not handle bursty and flash crowds for dynamic multicast groups. General limitations of any GKMP with respect to group secrets and policy complexity is explained.

Secure key management involves in generating the session key and facilitating key updates in an efficient manner. It also helps in maintaining join and leave secrecy. A naive key management technique is explained to provide an insight on the complexity requirements of secure key management. A tree based solution using Boolean function minimization is explained as it solves the cumulative member removal in an efficient manner. Cumulative member removal is the most desirable feature for large dynamic groups in which the leaves are bursty. Other operations are also done in logarithmic time complexity with fewer number of keys to be maintained in the GCKS and the members compared to other key management techniques.

A pragmatic framework for scalable secure source specific multicast is proposed with scalability, and ease of deployment as the core design requirements. The proposed framework is designed to co-exist with the multicast routing architecture with IGMPv3 and PIM-SM. The key distribution architecture is designed with desirable features from Iolus and GKMA. The proposed key distribution scheme localizes the group changes and hence the global session key remains mostly constant while the local session keys may change often. The key management scheme follows the technique using Boolean function minimization. The operations for individual member deletion has been modified so that in the best case, the LKS sends only one key update message, and each of the members will send at most one key update message to the group. Cumulative member deletion is possible as Boolean function minimization is used. The features like early detection of member
absence, no access control information leak, and ease of providing multiple license enable easy deployment. Collusion attacks remain as Boolean function minimization technique is used.

In this thesis, various issues in secure multicast have been highlighted. Important contributions in the area of secure key distribution and key management have been explained and reviewed with deployment as the core evaluation criteria. The review indicates that most of the solutions are not proposed with realistic implementations in mind. A framework for secure multicast is proposed to alleviate these shortcomings and with ease of deployment as the essential feature.

7.2 Future Work

Key management. The following is the list of improvements that can be done for key management:

- **Improved tree growth and shrinkage operations:** It is beneficial to build the tree before hand to a certain height, instead of increasing the height of the tree incrementally. The advantages are twofold: it minimizes the threat of collusion attacks as the incremental growth of the tree results in duals 1 more often as the LKS can assign UIDs randomly. This also reduces the overhead on the LKS during normal operation when there are lot of joins. It is also beneficial to maintain the KM tree at a certain height for a period of time before shrinking it. The reason is that, for dynamic groups, this may reduce the fluctuations of KM tree from growing and shrinking alternately. This also results in fewer keys being discarded and generated, the latter being a costly operation.

- **Early generation of keys:** LKS can generate multiple SKs beforehand, thereby improving the performance during normal operation of the LKS during key distribution.

---

1 duals are a pair of members whose UIDs are one’s complement of each other, resulting in the pair of members having all the keys in the KM tree
REFERENCES


BIOGRAPHICAL SKETCH

Murali M. Brahmathesam was born in Tiruchirappalli, Tamil Nadu, India, on July 5th, 1979. He earned his high school diploma from YWCA matriculation higher secondary school. He graduated with a Bachelor of Engineer degree with distinction in computer science and engineering in 2000 from Regional Engineering College, Tiruchirappalli. He came to Gainesville, Florida, to pursue a Master of Science in Computer and Information Science and Engineering Department, University of Florida.