Elliptic Curve for Secure Group Key Management in Distributed Network

Athira Vijay¹ Deepa S Kumar²

¹ PG Scholar, Dept. of Computer Science, College of Engineering Munnar,
² Associate professor, Dept. of Computer Science, College of Engineering Munnar

ABSTRACT

Group communication emphasis an important security criterion in the design of a distributed network. All the members of the group must agree to a common session key. The management of this session key refers to the group key management which is based on some group key agreement protocol. In this paper we propose a group key management method for secure group communication in a distributed network. Frequent change in group membership, and managing the key distribution for new members are the two main problems to be faced in group communication that too with minimal computation and communication overhead. Our system uses the concept of Elliptic curve Cryptography that provide same level of security as that of other cryptosystems with reduced key size. This results in less re-keying and re-distribution operations, thus reducing computation and communication overheads respectively.

Keywords: Computation cost, Communication cost, Distributed network, Elliptic Curve cryptography, Group communication, Key management, MANET, Security.

I. INTRODUCTION

With the sudden growth in technology and network, group communication has become an increased concern in terms of security. A secure group communication can be defined as a situation where a set of users communicate with each other through messages in such a way that anyone outside the set will be unable to gather any information even if they can obstruct the message. Confidentiality, integrity and authentication are the basic security services provided by any trustworthy group communication system. Group key management lay the basis for these services. So it is very necessary to ensure these services for efficient group communication.

Generally group key management protocol are classified into following three schemes:

1) Centralized: A centralized group key server take the responsibility of distributing and updating the group key to all the members in the network. But, a breakdown of key server will affect all the group application system. This is a major limitation of this scheme.

2) Decentralized: The groups are divided into different subgroups and a group key is shared between all group members. Every subgroup share a common subgroup key. Every group have a group key server (GK) which server all members in group and every subgroup have a subgroup key sever (SGK) which serve the subgroup members.

3) Distributed: In the distributed group management protocols, each individual member in the group is responsible for new group key generation and distribution. It is the most complex and difficult protocol. But, it is the best option for networks such as MANET.

4) Here we design an application of elliptic curve cryptography in a distributed network such as MANET. A MANET is a continuously self organizing and self configuring network. It is an application for distributed system with no particular infrastructure and base station. For confidentiality, members of the network exchange group keys for every membership change. This causes high computation overhead in the network. The key goal of this paper is to exchange these keys possibly in a MANET securely and efficiently with reduced computation overhead.

At times, several approaches have been proposed for distributed protocol to reduce the key size of group communication. Most of them were based on different types of Diffie-Hellman key agreement protocol. The main obstacle here is the large key size which will increase the computation overhead in the network.

In this paper, we propose a secure and authenticated group key management method in distributed network such as MANET using elliptic curve cryptography, to reduce the key size, re-keying computation and communication overhead compared to currently existing group key management approaches.
II. RELATED WORKS

Many works have been done in the field of group key management for secure communication. The papers enlisted from [4] to [14] discuss similar and typical approaches carried out in the field using different types of Diffie-Hellman key agreement protocols. The main drawback of these approaches is that, they either take high computation overhead or high communication cost or both.

Diffie-Hellman and Symmetric Algorithm (DHSA) [4] uses hierarchical approach to manage the keys logically. In this protocol, Diffie-Hellman key agreement and symmetric key are combined and assigned to the leaf nodes of the key tree and to intermediate nodes respectively. The drawbacks of this protocol are that key size is very large and modulo operation takes long computational time which makes the computation slow.

Paper describing secure group key agreement protocol using ECC (Elliptic Curve Cryptography) [8] is based on authenticated group key agreement protocol for wireless scenario. The protocol uses the concepts of elliptic curve cryptography to reduce the computation overheads and Asymmetric Encryption Standards(AES) to maintain efficiency. It consists of a set of users and a trustworthy server, where both of them contribute to create the group key. The performance and security analysis shows that the proposed protocol is secure and performs better in terms of computation cost. But as compared to the proposed system, the communication cost will be high due to larger key size, which accounts to the main drawback of this protocol.

Distributed group key agreement uses basically different variations of n-party Diffie-Hellman key agreement[9]. The main drawback of this system is that, the members must be synchronized to continuously calculate the parent’s key from that of children. If any member is slow in re-keying computation, the packet will be delayed and the process will be postponed or disrupted. Also, there are dependencies among node keys which may break all ancestral keys in case of any compromise in key computation.

Tree based Group Diffie-Hellman (TGDH) [13] approach again emphasis the use of two-party Diffie Hellman key agreement protocol to a group. It also introduces the concept of hierarchical group key management. The members at the leaf node maintains the key tree. Each intermediate node represents the key shared by its child nodes that are computed with single Diffie-Hellman key agreement protocol. The protocol initializes again with every membership change which lead to computational delay due to large cost of modular exponentiation.

III. PROPOSED SYSTEM

Every network is designed in such a way that every time a user join or leave the network, it causes some changes to the members in the network. Here, we consider group of users in the network to make the join and leave of new members and existing members reliable and secure. Since we use ECC for key calculation the system ensure decreased re-keying overhead during member join and leave operation.

The main purpose of ECC here is not the key delivery, but is the calculation of group key for the associated members of the group. Therefore, the main features of ECC that define the system are:

- In the hierarchical structure, leaf keys correspond to the public keys of the ECC key pair and all intermediate node keys are symmetric keys
- A periodically updated list of public key and corresponding parent node binary code on every membership change.
- Equally trusted and responsible group members with same capability

An elliptic curve over finite field with parameters a, b and P where, P is a prime and a point G on the curve whose order is larger than n is used. n is the private key which is a random number between 1 and P-1 and therefore n x G is obtained as the public key, which can be used to share common keys within members of the groups. For example, \( m_1 \) can share a key with \( m_2 \) by the common key \( n_1 \times n_2 \times G \)

ECC introduces two types of codes in it’s key tree, binary code, used for member position discovery and decimal code, used for intermediate node key calculation. Figure 1 illustrates a key tree with 8 members, \( \{m_1, ..., m_8\} \), and its corresponding binary code. A list is maintained to find the public key of the member in the network with whom a new incoming user wants to establish a connection. The list consist of the binary codes of intermediate node just above the leaf nodes and their corresponding member’s public key. The list is updated whenever there is a membership change and is informed to all the members through multicasting. Usually, the sibling member of affected branch is responsible to send the updated information to other members.
Table I shows the list of binary code and public key of corresponding members. As in the table, the public keys of \( m_1 \) is \( n_1 \times G \) and that of \( m_2 \) is \( n_2 \times G \) and their parent binary code is 000. Since \( m_2 \) doesn’t have a sibling node, the list just shows the public key of \( m_2 \) that is, \( n_2 \times G \) and the associated parent binary code, 00.

<table>
<thead>
<tr>
<th>Binary Code of Parent</th>
<th>Child Public Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>( n_1 \times G, n_2 \times G )</td>
</tr>
<tr>
<td>00</td>
<td>( n_3 \times G )</td>
</tr>
<tr>
<td>010</td>
<td>( n_5 \times G, n_6 \times G )</td>
</tr>
<tr>
<td>011</td>
<td>( n_7 \times G, n_8 \times G )</td>
</tr>
</tbody>
</table>

As mentioned before, other code is decimal which is used for intermediate node key calculation and is assigned to each intermediate node in the key tree. The intermediate node key is calculated by a one way hash function of the bitwise XOR of intermediate node code and the group key. It is defined in the following formula:

\[
\text{Key}_{\text{intermediate\_node}} = f([\text{Key}_{\text{group}} \oplus \text{Code}_{\text{intermediate\_node}}])
\]

And the intermediate node code is calculated by the formula:

\[
\text{Code}_{\text{child\_node}} = (\text{Code}_{\text{parent\_node}} \oplus \text{Random\_digit})
\]

In the figure 2, the node code management is illustrated with 8 members \( \{m_1, \ldots, m_8\} \). For example, if the intermediate node code is 02 and generated random digit is 6, the code of the child node will be 026. Therefore, the number of digits in a code represent, the number of nodes in that set of path.

During the join operation, the group key is sent to the new member, which is being encrypted using the shared key by the new member’s sibling node. The new members can calculate it by using a one way hash function on the previous key. If \( f \) is a hash function and \( K_0 \) is the previous group key, then new key can be calculate as:

\[
K'_f = f(K_{0})
\]

\[
K_{14} = f(K_{0} \oplus 02) \quad K_{5,8} = f(K_{0} \oplus 04)
\]

\[
K_{12} = f(K_{0} \oplus 025) \quad K_{5,6} = f(K_{0} \oplus 041)
\]

\[
K_{2,4} = f(K_{0} \oplus 028) \quad K_{7,8} = f(K_{0} \oplus 046)
\]

IV. DETAILED DESIGN

In the detailed design, we explain the join and leave operations of nodes to and from the network using the example figures 1 and 2 for join and 3 for leave operation. Members of the network commonly agree to a large
prime p and the primitive element G for each group. These values are selected during the initial stage of key tree establishment and are known publicly in the group.

During join operation of a new member to a group, he/she sends a hello message to discover the group members. Those members, who receive the signal, look up their corresponding list. The member who does not have a sibling member replies to the new node. But, if all the members have a sibling node, then the node with lowest parent binary ID replies to the node. The member then exchanges the public key generated using Elliptic Curve Cryptography. It is the sole responsibility of the member who replied to the new member to authenticate the incoming new member. Here, we assume the group members are equipped with some authentication methods. In advanced systems, an Elliptic Curve Digital Signature Algorithm can be used for this purpose.

Once the authentication is complete, the public key of new member and his/her corresponding parent binary code is updated in the list and this information is multicast to all other members. Now, the previous members and the new one can calculate the affected intermediate node keys by applying a given one way hash function to bitwise XOR of the intermediate node code and new group key.

![Intermediate node code and respective node key values](image)

**Figure 2:** Intermediate node code and respective node key values

### 4.1. Join Operation

Fig. 1 and 2 explains the procedures taking place while a new member \( m_4 \) joins the multicast group of 7 members \([m_1, m_2, m_3, m_5, m_6, m_7, m_8]\):

a. \( m_4 \) broadcast a hello message to all the members for member discovery.

b. The member who does not have a sibling node replies to the message. In our example, it is \( m_8 \).

c. \( m_3 \) shares the common key \( n_3 \times n_4 \times G \) with \( m_4 \).

d. \( m_3 \) downgrades it’s binary position from 00 to 001 and updates the member discovery table with its new parent binary code and public key of new member (Table II).

e. New intermediate node key is calculated by \( m_3 \) for it’s new parent using:

\[
\text{Code}_{K_{12}} = (02 || 08) = 028
\]

f. \( m_3 \) then generates new group key as below:

\[
K'_0 = f(K_0)
\]

g. \( m_3 \) then sends the new group key and node code to \( m_4 \) being encrypted using the common shared key between them.

\[
m_3 \text{ encrypts } (K'_0, 028) \times n_3 \times G
\]

h. All other existing members renew the group key as described in step f.

i. The members in the affected set of path calculate the intermediate node key by applying one way hash function to the bitwise XOR of the intermediate node code and and new group key.

\[
m_3, m_4: K_{12} = f(K'_0 \oplus 028)
\]
Elliptic Curve for Secure Group Key Management in Distributed Network

\[ m_1, \ldots, m_4: K_{1,4} = f(K'_2 \oplus 02) \]

Table II. Completely updated list after member join operation.

<table>
<thead>
<tr>
<th>Binary Code of Parent</th>
<th>Child Public Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>( n_1 \times G ), ( n_2 \times G )</td>
</tr>
<tr>
<td>00</td>
<td>( n_3 \times G ), ( n_4 \times G )</td>
</tr>
<tr>
<td>010</td>
<td>( n_5 \times G ), ( n_6 \times G )</td>
</tr>
<tr>
<td>011</td>
<td>( n_7 \times G ), ( n_8 \times G )</td>
</tr>
</tbody>
</table>

Here just one key is delivered to new member which is an important feature for distributed group communication in wireless network. Since members dynamically join/leave the network in addition to their self mobility, simultaneous join operation may take place here. Therefore, the ECC procedures are designed in such a way that, the overload of join operation is minimized.

4.2. Leave Operation

When a member leaves the group, it’s node is deleted from the key tree. The sibling node upgrades it’s position to the parent node position. And the sibling node of leaving node is responsible for deleting the public key from the list and updating the information to other members in the group. After every leave operation some intermediate node code and the group key need to be updated.

Figure 3 explains the procedures while a member \( m_8 \) leaves the multicast group of 8 members \( \{ m_1, m_2, m_3, m_4, m_5, m_6, m_7, m_8 \} \).

a. \( m_7 \) is promoted to it’s parent position.

b. \( m_7 \) changes the parent binary code and updates the member discovery list by removing the public key of the leaving member from the list. It also informs other members in the group about the new update.

c. Now, the new group key \( K'_G \) is generated by \( m_7 \) using symmetric algorithm.

d. \( m_7 \) checks the member discovery list and using elliptic curve key agreement, shares a key with one of the members in each sub group and then sends the new group key to the selected members.

e. \( m_4 \) and \( m_5 \) will now multicast the new group key to all the members in their respective sub group as follows:

\[ \text{multicast} \quad m_1 \rightarrow m_2, \ldots, m_4:(K'_G)K_{1,4} \]

\[ \text{multicast} \quad m_5 \rightarrow m_6, m_7:(K'_G)K_{3,6} \]

f. Finally, the members in the affected path set computer thrice new intermediate node code as follows:

\[ m_5, m_6, m_7: K_{57} = f(K'_2 \oplus 04) \]
Table III represents the updated member discovery list after a member leaves the group. This ensures backward secrecy by deleting the leaving nodes public key from the list.

**Table III: Updated list after member leave operation**

<table>
<thead>
<tr>
<th>Binary Code of Parent</th>
<th>Child Public Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>( n_1 \times G, n_2 \times G )</td>
</tr>
<tr>
<td>00</td>
<td>( n_3 \times G, n_0 \times G )</td>
</tr>
<tr>
<td>010</td>
<td>( n_5 \times G, n_6 \times G )</td>
</tr>
<tr>
<td>011</td>
<td>( n_7 \times G )</td>
</tr>
</tbody>
</table>

V. IMPLEMENTATION AND RESULTS

The proposed concept is implemented for Mobile Ad-Hoc Network using Network Simulator 2 (NS2). To provide a realistic mobile environment, NS2 is configured as follows: The proposed scheme is simulated by taking 50 nodes in a 1670m X 970m topology area. All of the nodes are mobile with no base station or particular infrastructure. In this mobility model, all the nodes move with a constant speed chosen between a minimum and maximum speed. The system is implemented with Ad-hoc On Demand Multipath Distance Vector routing protocol. Table IV shows the chosen simulation parameters.

**Table IV: Simulation Parameters**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channel Type</td>
<td>Channel/WirelessChannel</td>
</tr>
<tr>
<td>Radio-propagation model</td>
<td>Propagation/TwoRayGround</td>
</tr>
<tr>
<td>network interface type</td>
<td>Phy/WirelessPhy</td>
</tr>
<tr>
<td>MAC type</td>
<td>Mac/802_11</td>
</tr>
<tr>
<td>Interface queue type</td>
<td>CMUPriQueue</td>
</tr>
<tr>
<td>Antenna model</td>
<td>Antenna/OmniAntenna</td>
</tr>
<tr>
<td>Max packet in interface queue</td>
<td>300</td>
</tr>
<tr>
<td>Number of mobile nodes</td>
<td>50</td>
</tr>
<tr>
<td>routing protocol</td>
<td>AOMDV</td>
</tr>
<tr>
<td>Topology type</td>
<td>Flatgrid</td>
</tr>
<tr>
<td>Topology size</td>
<td>1670 x 970</td>
</tr>
<tr>
<td>Initial energy in Joules</td>
<td>100</td>
</tr>
</tbody>
</table>
5.1. Simulation and Analysis
In any network there are possible chances of intruders, which may affect the network performance and security. To analyse the performance of implemented network, we considered the following two metrics:

5.1.1. Throughput
Throughput can be defined as the actual rate at which data is transferred from a node to another. It is the ratio of total bytes received by time.

![Figure 4: Performance measurement using throughput](image)

5.1.2. End-to-End Delay
End-to-end delay can be defined as the time taken by a packet to reach the appropriate destination from the source. It is the difference between end time and start time.

![Figure 5: Performance analysis using delay](image)

VI. CONCLUSION
In this paper, we have tried to develop a system for a distributed network such as MANET. This protocol is based on logical key hierarchy because in this group members are arranged in hierarchical manner. The protocol uses symmetric cryptosystem for intermediate nodes and asymmetric cryptosystem for leaf nodes. For asymmetric key, Elliptic Curve Cryptography key agreement is introduced. Elliptic Curve Cryptography provides much stronger security with smaller key size.

So we conclude this proposal by highlighting the contributions that, by using elliptic curves in the system, we considerably reduced the key size, thus reducing the re-keying and re-distribution cost which further reduced the computation and communication costs respectively. Analysis of performance improvement is another
contribution that can be drawn from the simulation results by measuring the throughput and end-to-end delay of the network. The network have been simulated with NS2 and while simulating with AOMDV, throughput and end-to-end delay have improved drastically.

The future work of the system can be extended to the concept of providing individual authentication to all the users in the network. Since elliptic curve cryptography is used in the proposed protocol, Elliptic Curve digital Signature algorithm(ECDSA) can be used for this purpose. It can be ensured that, providing such an authentication facility will provide more secure system in the future.

REFERENCES