Network Security Project

Submitted by:Sahil Bafna(CS21M056)

Title:Key Establishment in Multicast Node Network

Abstract:

The purpose is to show how any user in the multicast group can compose the group keys and use the group Diffie Hellman key-exchange protocol (GDH) to securely multicast data from the multicast source to the rest of the multicast group in wireless ad hoc networks. Then, according to the measures of GPS parameters, it allows the multicast source to restructure representation of network topology, and computes a multicast tree. Hence the multicast source can utilize the group set information with the Prufer number of the multicast tree for the multicast header. As a result, we deliver the group key based on the network topology and the multicast routing. It not only provides the multicast routing information, but also fits for the wireless networks and reduces the overhead for the security management.
In Accordance with above I have Implemented BlockChain Based Key Sharing In Multicast System

Introduction:

1. The Project First Starts to take User Input to create the graph and then select MultiCast network node.
2. Then it takes the multicast network in the form of graph and form a Steiner Tree.
3. Then the Steiner tree helps in to create Prufers's Sequence code and Also in Group Deffie Hellman Key Exchange
4. Pruefers Sequence helps to create a Unique Sequence for the steiner tree
5. Pruferes Sequence generation takes help of following algorithm: Single source shortest path , All pair shortest path, K-subset Partitions
6. GDH Key Exchange extends the conventional diffie hellman key exchange to N nodes for key establishment among all of them.
7. Next I create Blockchain based node creation.
8. Following it we connected all the nodes with accordance with their respective hash keys

System/ security protocols/algorithm:

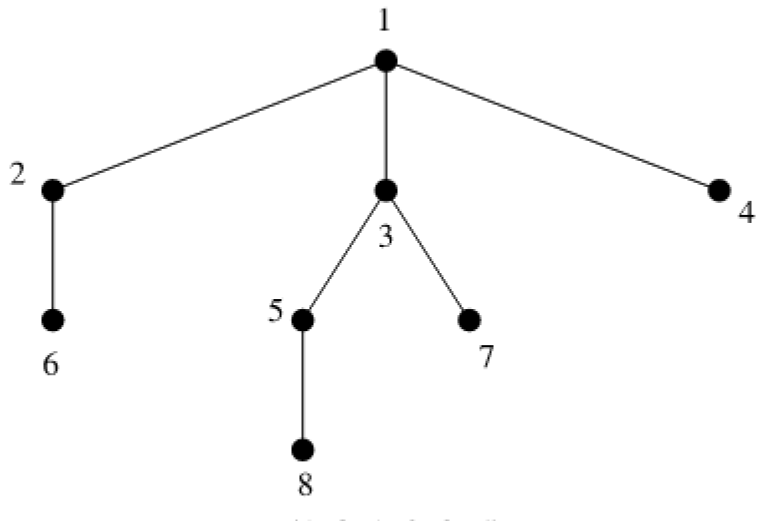
Steiner Trees:

-Steiner trees can be seen as a combination and generalization of two problems Shortest paths and minimum spanning trees
-Given *N* points , the goal is to connect them by lines of minimum total length in such a way that any two points may be interconnected by line segments either directly or via other points and line segments. It may be shown that the connecting line segments do not intersect each other except at the endpoints and form a tree, hence the name of the problem.

Pruferes Sequence:

-It is a unique code associated with a labeled tree
-for n node tree it has n-2 length
-One can generate a labeled tree's Prüfer sequence by iteratively removing vertices from the tree until only two vertices remain.

-consider a labeled tree *T* with vertices {1, 2, ..., *n*}. At step *i*, remove the leaf with the smallest label and set the *i*th element of the Prüfer sequence to be the label of this leaf's neighbour.
Example Given:



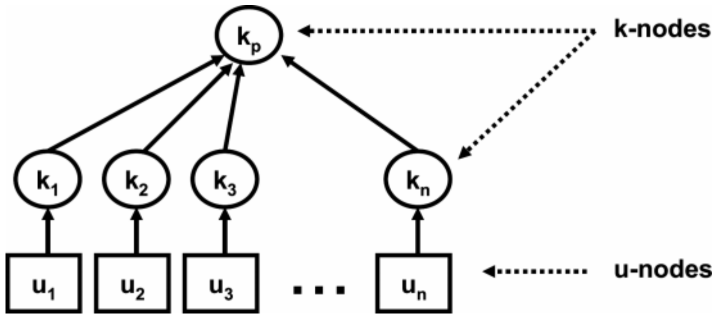
{1, 2, 1, 3, 3, 5}

<u>Secure Groups</u>:

-We begin by formalizing the notation of a secure multicast group as a triple (U,K,P), where U denotes a set of multicast group users, K denotes a set of keys for the users, and P denotes a Prüfer-key relation, which specifies keys related by each user in U. In particular, each user is given a subset of keys which includes the user's individual key and a group key (Prüfer-key). We next illustrate how to get the information by using the Prüfer decoding algorithm from a group of users in a tree. Then, we can give users additional keys.

-Then we use Prufers algorithm to Encode and Decode the topology graph
-The key Distribution graph model using prufer decoding algorithm is as follows:

1. Each multicast-user node in G corresponds to a unique u-node.
2. Each individual key corresponds to a unique k-node.
1. The Prüfer-key (a group key) has a directed path from all the k-nodes



Key Exchange strategies and secure multicast protocol

Maintaining a multicast tree for the purpose of multicast packets expends great effort due to the bandwidth constraint. We partition **g** multicast groups from the n nodes of an ad hoc network. An application needs to send a multicast packet securely from a multicast source to all the nodes of a multicast group.

For ex:For Source Node 2 with multicast group as ={2,3,6,9,13,16}

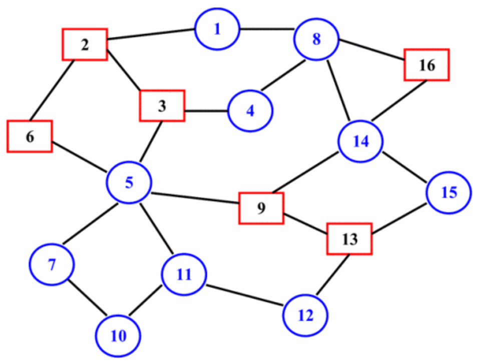
Since ad hoc networks have no fixed routers; all nodes are moveable and can be connected dynamically in an arbitrary manner. For our purpose; the cost of all links in an ad hoc network is We adapt the Prim's algorithm to compute a minimum cost multicast tree, also called a Steiner tree

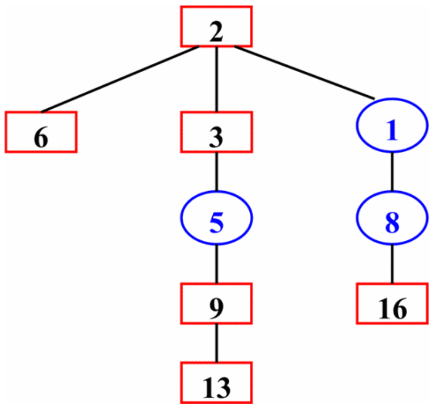
There can be following cases or packet delivery from source 2:

1.Packet delivered to leaf node(6) so it doesn't forward any further

2.Packet delivered to a node not in MG and the steiner tree(4) ,it will discard the packet

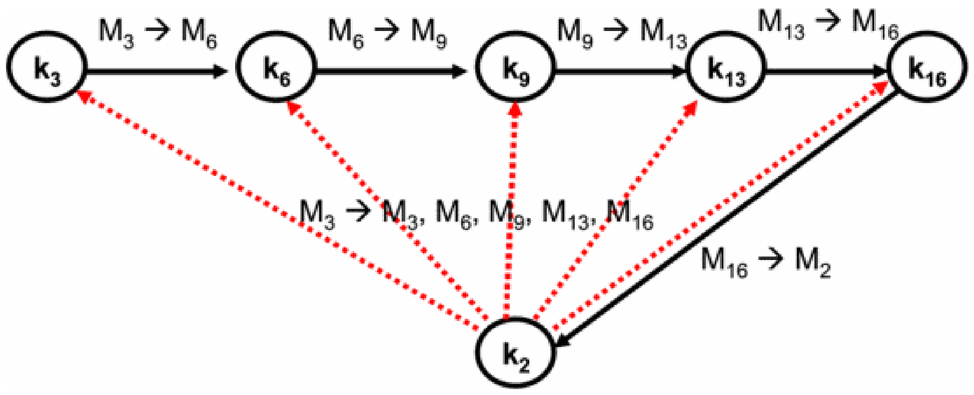
3.Packet delivered to an interior node(3),it decodes the multicast tree with Prüfer sequence number, and transmits the packet with the encoding information of its corresponding subtree(5,9)





Key-Distribution graph model using GDH protocol

Here we still need the internodes, 1, 5, and 8, which are not the members of the multicast group to relay the multicast message. Therefore emergence of the ad hoc networks raises a new challenge for the security of group multicast, since those internodes are difficult to prevent from attacking. So we can use the GDH protocol as we mentioned before to only encrypt the payload data with the group key generated from all multicast group members except the relay nodes.



<u>BlockChain for Multicast Network Simulation</u>:

Here we have generated/Simulated multi node block chain network from scratch using Python, and develop a decentralized data sharing application based on this simulated block chain network.

We create nodes with secret Key using HAsh functions

An ideal hash function has the following characteristics:

- Should be easy to calculate
- Should be deterministic, always generating the same hash for the same input data
- Should be uniform randomness, a slight change in the input data will also result in a significant change in the output hash

There are currently many popular hash functions, the following is a Python example using the SHA-256 hash function.

<u>Development Environment</u>

The following are the details

1.Language Used:Python3.0

2.IDE:Google Collab

3.Libraries:Collections,itertools,random,graphviz,sympy,random

4.Inspires Function:KSubset,Shortest_Path_Single_Source(Reference added at last)

Performance Result

The following are the performance of various algorithm:

Steiner Tree:O(Nlog(N))

Prufer Sequence:O(V*E) –V-Vertices,E--Edges

Advantages of Deffie Hellman:

1. The sender and receiver dont need any prior information of each other
2. The sharing of secret key is safe
3. Once the keys are exchanged, the communication of data can be done through an insecure channel

Drawbacks:

1. The algorithm can not be sued for any asymmetric key exchange
2. Since it doesn’t authenticate any party in the transmission, the Diffie Hellman key exchange is susceptible to a man-in-middle-attack

Conclusion:

Successfully implemented all the mentioned part and in addition I have added and extra concept of blockchain in multi nodes.

Learning Experience

The project Helped me to understand the following:

1. Trees and its variants can be very useful in Network communications.
2. Communication needs a reliable trusted unique identity like prufer sequence
3. Diffie hellman can be extended to n nodes unlike conventional 2 node
4. Communication is not just key algorithms but also how we present the data to any algorithm
5. BlockChain can be useful Multicast communication as well.

6. The subject material was extended in the project so understanding the basics were equally important before advancing.
7. The subject taught made it easier to implement the above mentioned algorithms

Suggestions

The course met all my expectations and provided good learning throughout ,be it from lecture or challenging assignments and projects.

References:

https://codereview.stackexchange.com/questions/1526/finding-all-k-subset-partitions

https://ieeexplore.ieee.org/document/1240393

https://patentimages.storage.googleapis.com/e2/ca/7e/e6b525046f7648/WO2020051486A1.pdf