EFFECTIVE KEY MANAGEMENT IN WIRELESS MESH NETWORK

Project Report submitted in partial fulfilment of the requirement for the degree of

Bachelor of Technology.

in

Computer Science & Engineering

under the Supervision of

Dr. Hemraj Saini

By

Kainaat Singh (111203)

to

Jaypee University of Information and Technology,
Waknaghat, Solan – 173234, Himachal Pradesh

Certificate

This is to certify that project report entitled "**EFFECTIVE KEY MANAGEMENT IN WIRELESS MESH NETWORKS**", submitted by **Kainaat Singh** in partial fulfillment for the award of degree of Bachelor of Technology in Computer Science & Engineering to Jaypee University of Information Technology, Waknaghat, Solan has been carried out under my supervision.

This work has not been submitted partially or fully to any other University or Institute for the award of this or any other degree or diploma.

Date:

Dr. Hemraj Saini
Assistant Professor(Senior Grade)

Acknowledgement

First of all I am grateful to **The Almighty GOD** for establishing me to complete this project. I wish to express my sincere thanks to Prof. Dr. RMK Sinha, Dean (CSE and IT), for providing me with all the necessary facilities.

I place on record, my sincere gratitude to Prof. Dr. Satya Prakash Ghrera, FBCS, SMIEEE Professor, Brig (Retd.) and Head, Dept. of CSE, for his constant encouragement.

I also thank Dr. Hemraj Saini, Assistant Professor (Senior Grade), Guide, Dept of CSE. I am extremely grateful and indebted to him for his expert, sincere and valuable guidance and encouragement extended to me.

I take this opportunity to record our sincere thanks to all the faculty members of Department of Computer Science and Engineering, for their help and encouragement. I also thank my parents for their unceasing encouragement and support.

I also place on record, my sense of gratitude to one and all who, directly or indirectly, have lent their helping hand in this venture.

Date: Kainaat Singh

Table of Content

LIST OF FIGURES

ABSTRACT

This project report talks about wireless mesh network functions, characteristics, security issues and a security key management scheme. Wireless Mesh Networks are replacing the Wireless Infrastructure networks in many areas due to their lower cost and higher flexibility. The wireless mesh networks provide network access for mesh as well as conventional clients with the help of mesh routers and mesh clients. Communication across these networks is formed via the bridge functions. Mesh routers also provide the minimal possible mobility and forms the backbone of WMNs.

Wireless mesh network is a rapidly deployed, self organisable and multi-hop network. The wireless and distributed natures of these networks make them prone to various different kinds of attacks, due to which great challenges are raised in securing these networks. Most existing security mechanisms use cryptographic keys where higher degree key management services are in demand. In this report, I present an effective key management scheme such that the induced network is very well connected and protected against possible attacks. This is done by introducing the ant colonization optimization technique to one of the existing algorithms available.

CHAPTER-1

1.1 INTRODUCTION

Wireless mesh networking (WMNs) is an attractive, emerging and a new way of communication due to its lower cost and its scalable wireless internetworking solutions for the near future, which is the main reason that it is becoming little popular in the communication sector. In all kinds of networks security it is one of the major factors for reliable and trusted communications.

WMNs have various advantages over other wireless networks.

1) They provide very simple settings, broadband capabilities and inherent fault tolerance in the case of any network failures. Deployment of WMNs is also very easy. They are dynamically self-configurable and self organisable with the existing nodes in the network as they automatically establish and maintain mesh connectivity between the nodes, so they bring reliable service coverage in the network.

2) Due to their cost effective solution they have been proposed in different networks. Mesh networks are seen as a type of mobile ad hoc network (MANET). Data can be transmitted to destination nodes by using multiple hops and it also provides the backbone nodes that are usually not mobile.

The IEEE 802.11 working groups have provided many standards for communication and now they are focusing on 802.11s standard because of its dynamic path configuration as well as topology learning. WM networking is a way to route the data, voice and instructions between the nodes. Sometimes WMNs provide local 802.11g access to clients and connects all neighbours using 802.11a backhaul but not always since requirements vary like peak data rate and coverage range etc.

3) The nodes automatically establish an ad hoc network and maintain the connectivity due to which the network provides dynamically self-organisable and self-healing and self-configurable and selects the optimal path back to the wired network. WMNs do consist of mesh routers and mesh clients. Mesh routers also provides network access for mesh as well as conventional clients. Mesh routers help form the mesh backbone and provide the minimum possible mobility. They provide the same coverage as conventional routers do but with the low transmission power. Usually they have multiple wireless interfaces but have similar hardware.

4) They provide the additional routing function for wireless mesh networking. Also the mesh client must have the necessary mesh functions for behaving like the mesh router and for Security in WMNs transmission of data in the networks. They have only one such wireless interface for connectivity. Gateway/bridge functions do not exist in these nodes. Clients are being interconnected using a wired backbone network with the help of wireless access points in WLAN deployments so because of that wireless networks can have only a single hop for the end to end path. For connectivity clients are required to be within a single hop range of wireless access points. To achieve more coverage they are required to have more number of fixed access points. In the large scale the deployment of WLAN is costly and very time consuming.

5) But with WMNs one can achieve wireless network coverage of a larger area without a dedicated access point and also without having to rely on wired backbone infrastructure. Mesh router provides network access to the wireless clients in WMNs and also by involving multiple wireless hops, communication between these mesh routers is achieved. Multiple mesh routers can serve as a gateway to internet connectivity in mesh networks or nodes. The main difference between the two is that the wired backbone network can be replaced by a wireless mesh network.

1.2 GOAL

Security in every kind of network is a huge challenge. Different types of threats and attacks can be caused to show network failure and they can disturb and change the routing data, updates and also decrease the network's performance. This project should present a method of a secure the key distribution to prevent such attacks in a wireless mesh network.

1.3 MOTIVATION

The main motivation of choosing WMNs is that nowadays around the globe WMNs are being deployed to allow internet access and offering many other services to the many users in the cities. Presently, there are already various applications for wireless networks. Wireless mesh technologies are a critical part in the infrastructure of wireless networks, and also in the performance of wireless applications. WMNs provide high coverage to all users. Due to these reasons, vendors and the research community have been actively investigating new ways to improve the performance of wireless mesh networks.

1.4 WIRELESS MESH NETWORK

The term ‘wireless mesh networks’ describes wireless networks in which nodes can talk directly or indirectly with one or more neighbouring nodes. The term mesh describes that all the nodes were connected to all the other nodes directly but mostly in the modern meshes it connects only a small set of nodes to each other. There are two types of nodes in WMN’s:

1.4.1-Mesh routers
1.4.2-Mesh clients

Both types of nodes can function as a host or a router too. Then packets are advanced on behalf of other nodes that may not be inside direct wireless transmission range of their destinations.

1.4.1 Mesh Routers

Mesh routers are static devices mainly. Using multi-hop technology it can achieve the similar coverage to a conventional router but with much lower power. It has additional routing functions that help in supporting mesh networking. It immensely helps the users by connecting them through with wireless mesh routers using Ethernet even though they do not wireless NIC, hence user can always be online. Therefore with the help of gateway or bridge functions they integrate with different wireless networks like cellular and wireless-fidelity.

POWER PC ROUTER

ARM EMBEDDED ROUTER

1.4.2 Mesh Clients

Mesh clients can be mobile or static. Mesh clients possess the required mesh functions and they can work as a router but they do not possess gateway or bridge functions. They only possess one wireless interface. We possess variety of devices that act as mesh clients.

LAPTOP

CELL PHONE

1.5 Wireless Mesh Networks Architecture

A Wireless Mesh Network can be categorized into three network architectures based on the network topology and functions of the nodes.

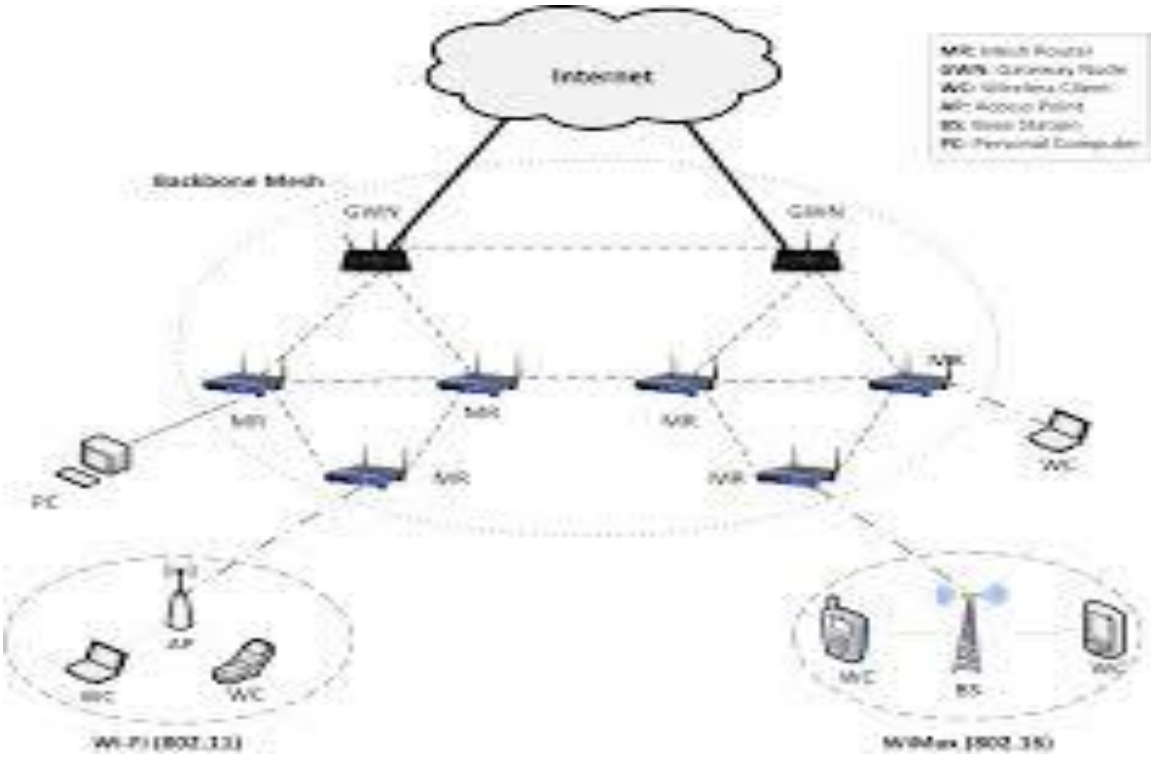
1.5.1 Infrastructure/backbone WMNs

In this the network is formed by connecting various types of nodes that are routers as well as clients. Each and every node is at the similar level that is of its neighbours. They are mesh routers that form an infrastructure required for clients that connect to the routers. The infrastructure can be built using various types of radio technologies. There are two types of radios which are used in routers, one used for backbone communication and other used for user communication. The direction antennas are provided for longer range communication in backbone communication. The backbone networks are being formed by applying self-configurable, self-healable links between themselves. Mesh networks have the strength to build themselves and configure as well. As any end node is powered on, it listens to and finds all the neighbour nodes and sends them a request to be a part of the networks and then nodes get into the network after fulfilling the network security requirements. Automatically path or routes will be made using the end node as the information that it transmits, gets relayed by neighbouring nodes until and unless it reaches the central node. But if one or more end nodes change their location then self healing function provides the reorganization

for the nodes in mesh networks and keeps those nodes working in the network. Self healable function provides the redundancy in the mesh network as, if a node is removed in the network then message can be sent within the network through other nodes. Self-configurable capability provides no human interference for re-routing of messages to the final nodes. Because of gateway function, the mesh routers are connected to the Internet that provides backbone for conventional clients. Users having Ethernet interface are connected to mesh routers using Ethernet links. The routers make a mesh by connecting with each other and are responsible for routing client's data. Data may travel using multiple router hops before reaching its final node.

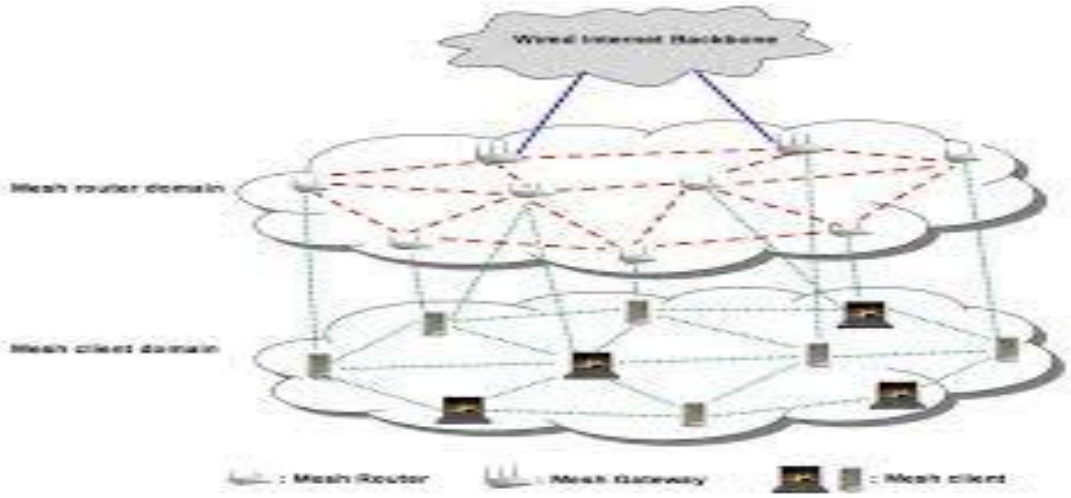
For example:

Community and neighbourhood networks can be made using this, the mesh routers are placed on the roof which serves as access point for user even if they are home users or they are using it on road.



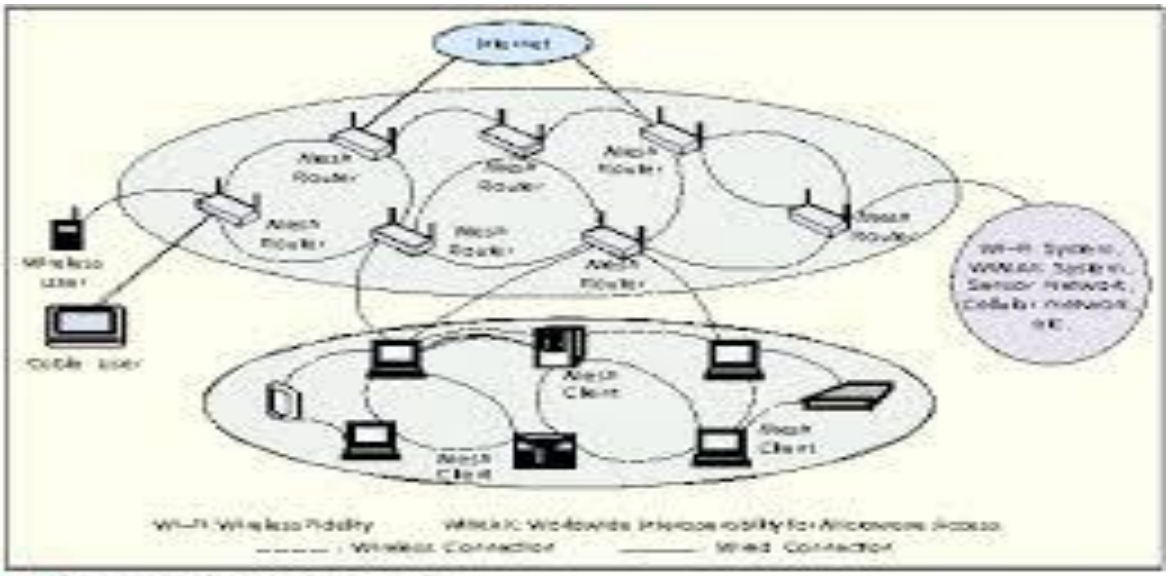
1.5.2 Client WMNs

They provide the peer to peer networks between the clients. The client nodes perform the routing as well as other configuration functions as well as provide end-user apps to the customers. The clients also perform the responsibilities and uphold the network connection.



1.5.3 Hybrid WMNs

This is the combination of all the mesh architectures. In Hybrid mesh using mesh routers or using directly meshing clients with all other mesh clients can use the network.



▲ Figure 1. Hybrid WMN architecture

1.6 Characteristics of WMNs

1. WMNs are multi-hop networks that help in providing much coverage range.
2. They are capable of self healable and self forming and self organisable and help in providing support for Ad Hoc Networking.
3. As there is multi hopping so it has higher throughput, and better frequency re-use.
4. They provide lower cost for installation due to the decrease in the number of access point so the main advantages of WMNs are that they are easy to deploy.

1.7 SECURITY CHALLENGES AND ISSUES IN WMNs

A wireless mesh network is prone to the basic threats that are common for both wired as well as wireless networks, thus the messages in such networks can be intercepted, delayed, modified, replayed, or some new messages can be inserted. However, wireless mesh networks are difficult to give full protection to.

1.8 APPLICATION SCENARIOS

- *Broadband Home Networking*: Mesh networking is required for resolving the location of all the access points in the home network. The access points can be replaced using wireless mesh routers with a mesh connection established between them. Thus, the communication among these nodes becomes so much more flexible.
- *Building Automation*: All access points can be replaced using WMNs and thus the deployment cost will be quite decreased. The process to deploy is much simpler due to the mesh connections among wireless routers.
- *Enterprise Networking*: When wireless mesh network is used, multiple backhaul access modems are to be shared by each and every node in the entire network, thus improving the robustness and resource utilization of the networks. Wireless mesh networks can increase easily as the size of organization expands.
- *Metropolitan Area Networks*: If compared to all cellular networks, wireless mesh MANs do support much high data rate, and the communication among the nodes does not have to rely on a wired backbone. If compared to wired networks, wireless MAN is a very economic alternative to broadband networking.

CHAPTER-2

2.1 LITERATURE REVIEW

2.1.1 A Key Management Scheme in Distributed Sensor Networks Using Attack Probabilities

- By Siu-Ping Chan, Radha Poovendran and Ming-Ting Sun.
- PURPOSE

  Establishing a secure connection in distributed sensor networks.

- INTRODUCTION

  1. Distributed sensor networks are ad-hoc MNs that may also include thousands of sensor nodes as well as they have limited computation and communication capabilities.
  2. Because of their limited calculation and communication capabilities, the sensor nodes have difficulty to bootstrap a secure communications infrastructure from a group of sensor nodes.

- PROPOSED SCHEME

  1. Subgrouping and random key predistribution.
  2. Probability of node compromise for a subgroup.

- METHODOLOGY

  1. There is a probability that there is a shared key existing between two sensor nodes within a subgroup.
  2. The resilience and the fraction of the attacked communications for a subgroup after z nodes in that subgroup have been attacked.
  3. The relation between the probability of node being attacked and the size of key pool in a node for a subgroup.

2.1.2 A Secure Key Management Scheme In Wireless Mesh Networks

- By Farah Kandah, Weiyi Zhang, Xiaojing Du, Yashwi Singh.
- PURPOSE

  To provide a wireless network that has high resilience against malicious eavesdropping attack.

- INTRODUCTION

  With more attentions on wireless mesh networks lately, the issues of security become very important and urgent in such cases for managing and deploying in wireless networks. Flexible deployment nature and also the lack of static infrastructure makes wireless mesh networks suffer from different types of security attacks, where such attacks might prevent their potential advantages to take place and wide scale deployment of this wireless network technology.

- PROPOSED SCHEME

  1. The way in which keys are distributed among every other node in the network can make the network resilient or prone to security attacks.
  2. Not all the keys that are generated in the pool will be used, nor all the keys provided in some nodes.

2.1.3 An Efficient Key Management Scheme with Key Agreement to Mitigate Malicious Attacks for Wireless Mesh Network

- By Vijay H. Kalmani, Sojwal S. Kulkarni and H.M. Rai
- PURPOSE

  The given network is to be connected and protected from malicious eaves dropping attacks.

- INTRODUCTION

  With more attentions on wireless mesh networks lately, the issues of security become very important and urgent in such cases for managing and deploying in wireless networks. Flexible deployment nature and also the lack of static infrastructure makes wireless mesh networks suffer from different types of security attacks, where such attacks might prevent their potential advantages to take place and wide scale deployment of this wireless network technology.

- PROPOSED SCHEME

  1. Distributing encryption keys between all nodes to be as different as possible so that the security attacks can be reduced.
  2. The Key Distribution Centre is provided which keeps the keys of each and every node.

2.1.4 RANDOM KEY PREDISTRIBUTION SCHEME

By Eschenauer and Gligor

They gave a random key pre-distribution algorithm. Let n denote the number of unique cryptographic keys that can be on a sensor node. The algorithm works as following. Before sensor nodes can be deployed, there is an initialization phase. In this phase, the basic algorithm picks up a random pool of keys out of the total possible keys. For each and every node, n keys are selected randomly from the key pool and saved into the node. These n key set is called the node's key ring.

After the sensor nodes have been deployed, there is a key-setup phase. Thus the nodes first perform key discovery to see as to with which of their neighbours they share a common key. Such discovery can be performed using a short identifier for each key prior to their deployment, and having each node to broadcast identifiers. Nodes which can discover that they possess a common shared key in their key ring can then prove that their neighbour actually has the key using a challenge-response protocol.

After key setup is terminated, there is a connected graph of secure links formed. Nodes can then set up their path keys with all the nodes that are in their vicinity and with whom they did not happen to share keys inside their key ring. But if the graph is connected, then a path can be found from a source to its neighbour. The source can then generate a random path key and send it using the path to the final node.

CHAPTER-3

3.1 Proposed Work

Various cryptographic algorithms need random or pseudorandom inputs at many different points, like for auxiliary quantities that are used in generating random digital signatures, and also for producing random challenges in the authentication algorithms. Therefore the security of an algorithm is dependent on keys. If a process that is cryptographically weak is used to produce keys then the entire system will become weaker. Therefore only the key should be secret. Various cryptographic algorithms depend on the secrecy of keys as algorithms are in the public domain. So to engage in secure communications what is required is to securely distribute the secret key or the public key. All keys should be dynamic. Therefore a key that is cryptographically strong must be generated and hence, the key management becomes a major problem in cryptography. I have proposed the use of ant colony optimization technique along with a basic key distribution scheme already proposed to assure that a strong and efficient key management is produced for wireless mesh networks that can be employed in the WMNs everywhere.

3.2 ALGORITHM

KEY DISTRIBUTION ALGORITHM

1**:** for every node $v \in G$ *do*

2: *keys(v) = NULL;*

3: end for

4: for every node in *G do*

5: for every node $v \in G$ *do*

6: Find *NIR(v);*

7: Calculate |*NIR(v)*|*;*

8: end for

9: Choose a node $v \in G$ *with the highest* |*NIR(v)*|*;*

10: for every node $u \in NIR(v)$ *do*

11: //Assign keys between node *u and node* $v \in NIR(v)$ *based on:*

12: if *keys(v) = NULL and keys(u) = NULL then*

13: Choose *k as the least used key from K set;*

14: Add *k to keys(v) and keys(u);*

15: else if *keys(v) = NULL and keys(u) = NULL then*

16: Choose *k as the least used key from K not in keys(x), where x is a neighbor of v, if applicable, else choose k as the least* used key from *K set;*

17: Add *k to keys(v) and keys(u);*

18: else if *keys(v) = NULL and keys(u) = NULL then*

19: Choose *k as the least used key from K not on x where* $x \in NIR(v) \cup NIR(u)$*, if applicable otherwise you can choose the least used* key from *K set;*

20: Add *k to keys(v) and keys(u);*

21: end if

22: end for

23: end for

ANTS ALGORITHM

1. Calculate a linear lower bound LB to the problem.

Initialize $\tau\ \iota\ \phi$ $(\forall\ \iota, \phi)$ with the primal available variable values

2. For m=1,k (k= number of ants) do

repeat

2.1 compute $\eta\ \iota\ \phi$ $\forall(\iota\ \phi)$

2.2 choose the probability of the state to move into

2.3 append the selected move to the *m*-th ant's tabu list until and unless ant *m* has completed its solution

2.4 take the solution to its local optimum

end for

3. For every ant move $(\iota\ \phi)$,

calculate $\Delta\ \tau\ \iota\ \phi$ and update trail

4. If not them end_test and goto step 2.

CHAPTER-4

4.1 SOFTWARE REQUIREMENTS

MATLAB R2012a

MATLAB is a higher-level language that has an interactive environment for every possible numerical calculation, visualization, and programming languages. Also by using MATLAB, you can analyze all types of data, develop all types of algorithms, and create many different models and applications. The languages and tools, and built-in maths functions provide you the ability to explore different approaches and also reach a solution in an easier way than with traditional programming languages, like C/C++ or Java. Also, you can use the MATLAB for a huge range of applications, also including the signal processing and communications, image and video processing, etc. More than a million engineers in industry and academia have to use MATLAB everyday, the language of technical computing.

HISTORY

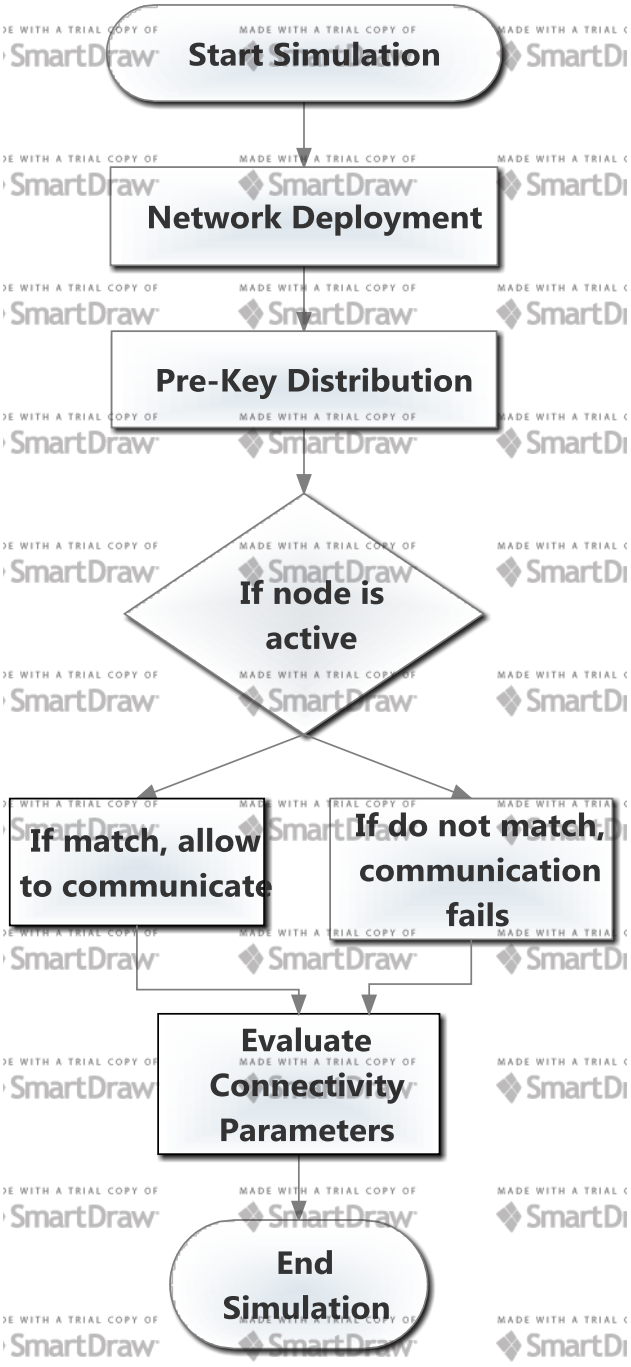
The chairman of the CSE department at the University of New Mexico, Cleve Moler, had started working to develop MATLAB during the late 1970s. He designed so that he could give all his students accessibility to LINPACK and EISPACK regardless of them having to learn Fortran. And it soon spread like fire to other universities also and therefore found a huge audience among the applied mathematics community. There was an engineer, Jack Little, who was luckily exposed to it during the time Moler visited the Stanford University during 1980's. He helped recognize its commercial potential, therefore he joined Moler and Steve Bangert in the development team. They again wrote MATLAB in C and thus founded MathWorks in 1984 so that it could continue its development. Also, these again written libraries were also called as JACKPAC. Hence, in 2000 MATLAB was again written so that a newer set of libraries could be used for matrix manipulation.

4.2 Hardware Requirements

- Processor: x86 compatible processor
- RAM: 512 MB or greater
- Hard Disk: 20 GB or greater
- Monitor: VGA/SVGA
- Keyboard: 104 keys standard
- Mouse: 2/3 button. Optical/ Mechanical

CHAPTER-5

5.1 Flow Chart



5.2 IMPLEMENTATION

INSTALLING MATLAB

You can get administrator privileges for the computer on which you are going to install MATLAB software.

Use WinRAR so as to extract RAR file

Step1: Start up the installer

Step 2: Choose the option to Install Without Using the Internet

Step 3: Check review the License Agreement

Step 4: Enter the File Installation Key that is provided

Step 5: Choose the Installation Type that you want

Step 6: Specify the Installation Folder correctly

Step 7: Specify Products to Install (Custom Only)

Step 8: Specify the Location of the License File that has been provided

Step 9: Specify Installation Options (Custom Only)

Step 10: Confirm Your Choices once again and Begin Copying the Files

Step 11: Complete the Installation by clicking on Finish

Step 12: Set Environment Variables

CODE

finalwsn.m

```
function varargout = finalwsn(varargin)


gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @finalwsn_OpeningFcn, ...
                   'gui_OutputFcn',  @finalwsn_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
```

```
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

function finalwsn_OpeningFcn(hObject, eventdata, handles, varargin)

handles.output = hObject;

guidata(hObject, handles);

function varargout = finalwsn_OutputFcn(hObject, eventdata, handles)

varargout{1} = handles.output;

function pushbutton1_Callback(hObject, eventdata, handles)

clear all
close all
clc

tic
disp('Creating Nodes and Clusters Please Wait.....');
h = waitbar(0,'Creating Nodes and Clusters Please Wait...');
for m=1:100,
    waitbar(m/100)
    pause(0.1);
end
close(h);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% PARAMETERS %%%%%%%%%%%%%%%%%%%%%%%%%%%%

xm=100;
ym=100;

sink.x=0.5*xm;
sink.y=0.5*ym;

n=100;
p=0.1;
Eo=0.5;

ETX=50*0.000000001;
ERX=50*0.000000001;

Efs=10*0.000000000001;
Emp=0.0013*0.000000000001;

EDA=5*0.000000001;
mwsn=1;

m=0.1;
a=1;
rmax=9999;

%%%%%%%%%%%%%%%%%%%%%%%%% END OF PARAMETERS %%%%%%%%%%%%%%%%%%%%%%%%

do=sqrt(Efs/Emp);

figure(1);
for i=1:1:n
```

```
    S(i).xd=rand(1,1)*xm;
    XR(i)=S(i).xd;
    S(i).yd=rand(1,1)*ym;
    YR(i)=S(i).yd;
    S(i).G=0;

    S(i).type='N';

    temp_rnd0=i;

    if (temp_rnd0>=m*n+1)
        S(i).E=Eo;
        S(i).ENERGY=0;
        plot(S(i).xd,S(i).yd,'o');
        hold on;
    end

    if (temp_rnd0<m*n+1)
        S(i).E=Eo*(1+a)
        S(i).ENERGY=1;
        plot(S(i).xd,S(i).yd,'+');
        hold on;
    end
end
figure;
clc;



n=100;
Eo=0.5;
m=0.1;

mu1 = [1 2];
sigma1 = [3 .2; .2 2];
mu2 = [-1 -2];
```

```
sigma2 = [2 0; 0 1];

%%%%%%%%%%%%%%%%%%%%%%%%%%%% END OF PARAMETERS %%%%%%%%%%%%%%%%%%%%%%%%%%%%




X.P = [mvnrnd(mu1,sigma1,n);mvnrnd(mu2,sigma2,n)];

for i=1:1:n
  temp_rnd0=i;

    if (temp_rnd0>=m*n+1)
        X.E=Eo;
        X.ENERGY=0;
    end
end

options = statset('Display','final');
gm = gmdistribution.fit(X.P,2,'Options',options);
clc

idx = cluster(gm,X.P);
cluster1 = (idx == 1);
cluster2 = (idx == 2);



scatter(X.P(cluster1,1),X.P(cluster1,2),n,'r+');
hold on
scatter(X.P(cluster2,1),X.P(cluster2,2),n,'bo');
hold off
legend('Cluster 1','Cluster 2','Location','NW')
```

```
disp('Processing Network Protocol Please Wait.....');
h = waitbar(0,'Processing Network Protocol Please Wait...');
for m=1:100, % computation here %
    waitbar(m/100)
    pause(0.1);
end
close(h);
clc;
close all;
clear all;
xm=100;
ym=100;
sink.x=0.5*xm;
sink.y=0.5*ym;
n=100 ;
p=0.1;
Eo=0.5;
ETX=50*0.000000001;
ERX=50*0.000000001;
Efs=10*0.000000000001;
Emp=0.0013*0.000000000001;
EDA=5*0.000000001;
rmax=1000;
do=sqrt(Efs/Emp);
ms=0;
Et=50;
disp('Processing Mesh Network Please Wait.....');
h = waitbar(0,'Processing Mesh Network Please Wait...');
for m=1:100, % computation here %
    waitbar(m/100)
    pause(0.1);
end
close(h);
clc
```

```
for i=1:1:n
   S(i).xd=rand(1,1)*xm;
   XR(i)=S(i).xd;
   S(i).yd=rand(1,1)*ym;
   YR(i)=S(i).yd;
   S(i).G=0;
   S(i).E=Eo;
   S(i).type='N';

end

S(n+1).xd=sink.x;
S(n+1).yd=sink.y;

algo
countCHs=0;
cluster1=1;
flag_first_dead=0;
flag_teenth_dead=0;
flag_fifth_dead=0;
flag_all_dead=0;
dead=0;
first_dead=0;
teenth_dead=0;
fifth_dead=0;
all_dead=0;
[x,y,d,t,h,iter,alpha,beta,e,m,n,el]=ants_information;
for i=1:iter
   [app]=ants_primaryplacing(m,n);
   [at]=ants_cycle(app,m,n,h,t,alpha,beta);
   at=horzcat(at,at(:,1));
   [cost,f]=ants_cost(m,n,d,at,el);
   %o=at
   [t]=ants_traceupdating(m,n,t,at,f,e);
   costoa(i)=mean(cost);
```

```
    [mincost(i),number]=min(cost);besttour(i,:)=at(number,:);
    iteration(i)=i;
end

[~,l]=min(mincost);
for i=1:n+1
    X(i)=x(besttour(l,i));
    Y(i)=y(besttour(l,i));
end
for i=1:n

end


% Initial Parameters for PSO
noP=30;
Max_iteration=500;
Dim=10;
vel=zeros(noP,Dim);
p=zeros(noP,Dim);
pBestScore=zeros(noP);
pBest=zeros(noP,Dim);
gBestScore=inf;
gBest=zeros(1,Dim);
fitness=rand(1,Max_iteration);
% Initialization
for i=1:size(p,1) % For each Particle
    for j=1:size(p,2) % For each dimension
          p(i,j)=rand();
        vel(i,j)=0.3*rand();
    end
end

 gBestScore=inf;
if fitness(p(1)==1)>=fitness(pBest(1)==1)
```

```
    pBest=p;
end

p = p + vel;
overlimit=p<=1;
underlimit=p>=0;
p=p.*overlimit+not(overlimit);
p=p.*underlimit;
Iteration=1:Max_iteration;
fitness(1,Iteration)=gBestScore;
O_gbest=mean(Iteration);

allive=n;

packets_TO_BS=0;
packets_TO_CH=0;
for r=0:1:rmax
    r ;

 % if(mod(r, round(1/p) )==0)
    for i=1:1:n
        S(i).G=0;
        S(i).cl=0;
    end
  %end

  E1(r+1)=0;
  for i=1:100
      E1(r+1)=S(i).E+E1(r+1);
  end
  Ec2(r+1)=Et-E1(r+1);

dead=0;
for i=1:1:n
```

```
    if (S(i).E<=0)
      dead=dead+1;

       if (dead==1)
         if(flag_first_dead==0)
           first_dead=r;
           flag_first_dead=1;
         end
       end

       if(dead==0.1*n)
         if(flag_teenth_dead==0)
           teenth_dead=r;
           flag_teenth_dead=1;
         end
       end
        if(dead==0.5*n)
         if(flag_fifth_dead==0)
           fifth_dead=r;
           flag_fifth_dead=1;
         end
        end
       if(dead==n)
         if(flag_all_dead==0)
           all_dead=r;
           flag_all_dead=1;
         end
       end
    end
    if S(i).E>0
       S(i).type='N';
    end
end

STATISTICS.DEAD(r+1)=dead;
```

```
DEAD(r+1)=dead;
STATISTICS.ALLIVE(r+1)=allive-dead;
ALLIVE(r+1)=allive-dead;
countCHs=0;
cluster1=1;
for i=1:1:n
 if(S(i).E>0)
   temp_rand=rand;
   if ( (S(i).G)<=0)

           countCHs=countCHs+1;
           packets_TO_BS=packets_TO_BS+1;
           PACKETS_TO_BS(r+1)=packets_TO_BS;
            S(i).type='C';

             end
    end
end
STATISTICS.COUNTCHS(r+1)=countCHs;
COUNTCHS(r+1)=countCHs;

for i=1:1:n
   if ( S(i).type=='N')
      if ( S(i).E>0 )
    if(cluster1-1>=1)
      min_dis=sqrt( (S(i).xd-S(n+1).xd)^2 + (S(i).yd-S(n+1).yd)^2 );
      min_dis_cluster=0;
      for c=1:1:cluster1-1
          temp=min(min_dis,sqrt( (S(i).xd-C(c).xd)^2 + (S(i).yd-C(c).yd)^2 ) );
           if ( temp<min_dis )
             min_dis=temp;
             min_dis_cluster=c;
          end
      end
```

```
            min_dis;
            if (min_dis>do)
               S(i).E=S(i).E- ( ETX*(4000) + Emp*4000*( min_dis * min_dis * min_dis *
min_dis));
            end
            if (min_dis<=do)
               S(i).E=S(i).E- ( ETX*(4000) + Efs*4000*( min_dis * min_dis));
            end


         if(min_dis>do)
            S(C(min_dis_cluster).id).E = S(C(min_dis_cluster).id).E- ( (ERX + EDA)*4000
);
          PACKETS_TO_CH(r+1)=n-dead-cluster1+1;
         end


         S(i).min_dis=min_dis;
        S(i).min_dis_cluster=min_dis_cluster;

        packets_TO_BS=packets_TO_BS+1;
           % plot(packets_TO_BS)
     end
   end
    end
end
STATISTICS.PACKETS_TO_CH(r+1)=packets_TO_CH;
PACKETS_TO_CH(r+1)=packets_TO_CH;
STATISTICS.PACKETS_TO_BS(r+1)=packets_TO_BS;
PACKETS_TO_BS(r+1)=packets_TO_BS;
end
%ALive=ALLIVE+allive1

first_dead;
teenth_dead;
```

```
all_dead;
STATISTICS.DEAD(r+1);
STATISTICS.ALLIVE(r+1);
STATISTICS.PACKETS_TO_CH(r+1);
STATISTICS.PACKETS_TO_BS(r+1);
STATISTICS.COUNTCHS(r+1);
r=0:1000;
close all;
clc;
clear all;
rmsn=10;
xm=100;
ym=100;
sink.x=0.5*xm;
sink.y=0.5*ym;
msn=5;
p=0.1;
Emsn=.5;
ETX=50*0.000000001;
ERX=50*0.000000001;
Efs=10*0.000000000001;
Emp=0.0013*0.000000000001;
EDA=5*0.000000001;
rmax=10000
do=sqrt(Efs/Emp);

for i=1:1:msn
   S(i).xd=rand(1,1)*xm;
   XR(i)=S(i).xd;
   S(i).yd=rand(1,1)*ym;
   YR(i)=S(i).yd;
   S(i).G=0;
   S(i).E=Emsn;
   %initially there are no cluster1 heads only nodes
   S(i).type='msn';
```

```
end

S(msn+1).xd=sink.x;
S(msn+1).yd=sink.y;

countCHs=0;
cluster1=1;
flag_first_dead1=0;

flag_all_dead1=0;
dead1=0;
first_dead1=0;

all_dead1=0;

allive1=msn;

packets_TO_BS=0;
packets_TO_CH=0;

for r=rmsn:1:rmax
   r

 if(mod(r, round(1/p) )==0)
   for i=1:1:msn
      S(i).G=0;
      S(i).cl=0;
   end
 end

dead1=0;
for i=1:1:msn

   if (S(i).E<=0)
```

```
        dead1=dead1+1;


        if (dead1==1)
          if(flag_first_dead1==0)
            first_dead1=r;
            flag_first_dead1=1;
          end
        end



        if(dead1==msn)
          if(flag_all_dead1==0)
            all_dead1=r;
            flag_all_dead1=1;
          end
        end
    end
    if S(i).E>0
        S(i).type='msn';
    end
end
STATISTICS.DEAD1(r+1)=dead1;
DEAD1(r+1)=dead1;
STATISTICS.ALLIVE1(r+1)=allive1-dead1;
ALLIVE1(r+1)=allive1-dead1;
countCHs=0;
cluster1=1;
for i=1:1:msn
 if(S(i).E>0)
   temp_rand=rand;
   if ( (S(i).G)<=0)

        if(temp_rand<= (p/(1-p*mod(r,round(1/p)))))
```

```
            countCHs=countCHs+1;
            packets_TO_BS=packets_TO_BS+1;
            PACKETS_TO_BS(r+1)=packets_TO_BS;
             S(i).type='C';
            S(i).G=round(1/p)-1;
            C(cluster1).xd=S(i).xd;
            C(cluster1).yd=S(i).yd;
           distance=sqrt( (S(i).xd-(S(msn+1).xd) )^2 + (S(i).yd-(S(msn+1).yd) )^2 );
            C(cluster1).distance=distance;
            C(cluster1).id=i;
            X(cluster1)=S(i).xd;
            Y(cluster1)=S(i).yd;
            cluster1=cluster1+1;

           distance;
            if (distance>do)
               S(i).E=S(i).E-          (          (ETX+EDA)*(4000)          +          Emp*4000*(
distance*distance*distance*distance ));
            end
            if (distance<=do)
               S(i).E=S(i).E- ( (ETX+EDA)*(4000)  + Efs*4000*( distance * distance ));
            end



        end

    end
    % S(i).G=S(i).G-1;
     end
end

STATISTICS.COUNTCHS(r+1)=countCHs;
COUNTCHS(r+1)=countCHs;
```

```
for i=1:1:msn
  if ( S(i).type=='msn')
     if  S(i).E>0
    if(cluster1-1>=1)
     min_dis=sqrt( (S(i).xd-S(msn+1).xd)^2 + (S(i).yd-S(msn+1).yd)^2 );
     min_dis_cluster=0;
     for c=1:1:cluster1-1
        temp=min(min_dis,sqrt( (S(i).xd-C(c).xd)^2 + (S(i).yd-C(c).yd)^2 ) );
         if ( temp<min_dis )
            min_dis=temp;
            min_dis_cluster=c;
        end
     end
        %Energy dissipated by associated cluster1 Head
         min_dis;
         if (min_dis>do)
             S(i).E=S(i).E- ( ETX*(4000) + Emp*4000*( min_dis * min_dis * min_dis *
min_dis));
         end
         if (min_dis<=do)
             S(i).E=S(i).E- ( ETX*(4000) + Efs*4000*( min_dis * min_dis));
         end


       %Energy dissipated
       if(min_dis>do)
          S(C(min_dis_cluster).id).E = S(C(min_dis_cluster).id).E- ( (ERX + EDA)*4000
);
        PACKETS_TO_CH(r+1)=msn-dead1-cluster1+1;
       end


       S(i).min_dis=min_dis;
      S(i).min_dis_cluster=min_dis_cluster;
```

```
      packets_TO_BS=packets_TO_BS+1;
%
   end
      end
  end
end
dne=10;
STATISTICS.PACKETS_TO_CH(r+1)=packets_TO_CH;
PACKETS_TO_CH(r+1)=packets_TO_CH;
STATISTICS.PACKETS_TO_BS(r+1)=packets_TO_BS;
PACKETS_TO_BS(r+1)=packets_TO_BS;
end
    vo = 1500 ;
Tdelay =30;
bitsDroped=0;
bitsReceived=0;


PacketSize=2;
NoOfPackets=1;
PreDefinedDelay = 3;
delay=0;
bitsSend = 0;
 bitsdroped=0;
W=20;

 n = 10 ;

L = 100 ;
B = 100 ;
D = 100 ;


%-topo function----------
destination.x = L;
```

```
destination.y = B;
destination.z = D;


forwarding.X = .9*L;
forwarding.Y = .9*B;
forwarding.Z =  .9*D;

source.X = 1;
source.Y = 1;
source.Z = 1;

figure(1);

y=rand(n,1)-0.5;
x=rand(n,1)-0.5;

h=plot(x,y,'.','Markersize',10);
axis([-1 1 -1 1])
axis square
set(gca,'XTick',[])
set(gca,'YTick',[])
%-topology function----------


    distanceDS = sqrt((destination.x -source.X)^2+ (destination.y -source.Y)^2 +
(destination.z -source.Z)^2);
    RoutingTable=zeros(n);

  CosTheta =zeros(n);
  CosBeta =zeros(n) ;
       u=0;
        R= 25; % in meters
```

```
alfac=2;



%----------Energy Consumption Pattern----------------
SumRx= 0 ;
Erx = .01;
SumTx= 0 ;
Etx = .02;
SumDx= 0 ;

SumAx= 0 ;
EAx = .05;

SumBx = 0 ;
EBx = .04;

SumIx = 0 ;
EIx = .03;

mode = 1;

EgPerSen = 10;
totaleng = n*EgPerSen;

%------------------------------
TrafficType ='cbr';

%------------------------------
dns= zeros(totaleng);

while u < 2 ,
```

```
    u=u+1;
        SumIx= SumIx+EIx;




      disp('::::::::::::::::::')

     close all

Total_Mobile_Anchors=10;
n=Total_Mobile_Anchors;
MobAnchor=1:1:n;
Total_Anchor_Pos=1:3;
i=MobAnchor;
dne_wsn=1;
if i==n
   Node(nodeID) && Total_Anchor_Pos<3
   XCrood(nodeID(Total_Anchor_Pos))=XCrood(MobAnchor(i));
   YCrood(nodeID(Total_Anchor_Pos))=YCrood(MobAnchor(i));
else if Total_Anchor_Pos==3
      j=1:3;
      newXCrood(nodeID)=(1/3)*sum(XCrood(nodeID(j)));
      newYCrood(nodeID)=(1/3)*sum(YCrood(nodeID(j)));
   end
end
   mAph = min(RoutingTable);
    Rdelay=0;
    for k=1:size(RoutingTable)
       tic;

       if(RoutingTable(k)==mAph)
            disp('Found Route :Sending RReq:')
            for j=1:1:NoOfPackets
```

```
 Ta =rand(1) ;

% Cycle Simulation
 Ta= Ta + 1.5;

 SumTx = SumTx+ Etx;


 if(Ta<=2.5)
 pause(Ta);
bitsSend=bitsSend+1;
disp('Status:Packet Send');
delay =Ta+delay;
 if (rand(1,1)>0.411);
 disp('Status:Packet Received:RRep');
  bitsReceived=bitsReceived+1;
    SumRx = SumRx+ Erx;


else
    SumDx = SumDx+ EIx;
    disp('Status:Packet Droped');
     bitsdroped=bitsdroped+1;
end

 else
    SumDx = SumDx+ EIx;

     disp('Packet Droped')
      bitsdroped=bitsdroped+1;
 end

end
```

```
            SumBx = SumBx+EBx;
        else

            % Syncho Mode

            ta =2 * rand(1);


            if  ta<1.1

              SumBx = SumBx+EBx;
              SumIx = SumIx+EIx;
            else

              disp('No Energy Consumption :::')
              disp('Listing Mode ')

            end %----------End of Mode Logic

                end

    end



    %-------------------------------------------%

    totaleng =  SumTx+SumRx+SumDx+SumIx + SumBx; % Total Consumption

    %---------Total Eng Calculations ------------%

end

     PDR = bitsSend/bitsReceived ;
     TT = SumRx;SumTx+SumRx+SumDx+SumIx + SumBx ;
```

```
    txndrx=toc;
librarysupport;

clc
h=plot(i1,o2,'-b',i5,o5,'-g',i,o1,'-.w',i,o11,i,o12);grid on;
set(h,'markersize',5,'linewidth',2.6);
xlabel('Time (hours)');
ylabel('Number of Nodes alive');
set(gca,'YTickLabel','|5|10|15|20|25|30||');
set(gca,'XTickLabel','0|5|10|15|20|25|30|35|40|');
title('Network lifetime comparison amongst Previous and Proposed Work');
legend('Previous Work','Proposed Work');
intial=1.5;
final=8;
%signal_range=4;%figure(1);
%h=plot(intial,signal_range,'O-b',final,signal_range,'O-b',input,o1,input,o2);
%set(h,'markersize',10,'linewidth',10);hold on;axis off;
%title('Time-of-Arrival for WSN','fontsize',15,'FontWeight','bold','color','m');
%text(intial-0.3,signal_range+0.7,'Tx-Node','fontsize',14,'FontWeight','bold','color','r');
%text(final-0.3,signal_range+0.7,'Rx-Node','fontsize',14,'FontWeight','bold','color','r');
for t=intial:1:final;
%pause(0.5);axis off;
% hold on;s=plot(t,signal_range,'x-g','markersize',4,'linewidth',6);
% legend(s,'Signal',4);
end


Vp=1.0003;% Signal propagation speed for the air
dist=final-intial;
sending_time=starting_time;
TOA_value=(dist+sending_time*Vp)/Vp;
TOA_value=TOA_value/60;
value=sprintf('The value of Time of arrival :%g',TOA_value);
%msgbox(value,'Time of Arrival');
%display(TOA_value);
```

```
tot_time=2+TOA_value;
comp_t=tot_time;
  E=sprintf('Computational Time: :%g%',comp_t);
    msgbox(E);
%oa=85;
%ir=txndrx;
%f = figure('Position',[100 100 600 110]);
%title('Comparison of data delivery time between Previous and our algorithm','fontsize',12,'FontWeight','bold','color','b');axis off;
%data = [oa ir];
%cnames = {'Previous Work',' Proposed Work '};
%rnames = {'Time'};
%uitable('Parent',f,'Data',data,'ColumnName',cnames,...
 %          'RowName',rnames,'Position',[110 20 360 60]);
%figure;
%y = [oa ir];
%bar(2, y(1),0.5,'FaceColor','g');
%hold on;bar(3, y(2),0.5,'FaceColor','c');grid on;
%title('Delivery Time','fontsize',16,'color','m','fontweight','bold');
%xlabel('Algorithms','fontsize',12,'fontweight','bold');
%ylabel('Time','fontsize',12,'fontweight','bold');
%set(gca,'XTickLabel','                    Previous Work                    Our Algorithm');


clc;
clear all;

function pushbutton2_Callback(hObject, eventdata, handles)

clc
clear all
close all
```

algo.m

```
u=200;
v=300;
w=400;
Nodes=[u v w];
Nu=Nodes(1,1);
Nv=Nodes(1,2);
Nw=Nodes(1,3);
C=1500;
NCA=mode(2*C*Nu);
MEA=max(NCA);pause(2);
disp('**********************************');
%disp('Please enter encryption key');
disp('**********************************');
%nn = input('Enter encryption key: ', 's');
nn='25';
wt = waitbar(0,'Please wait simulation in progress...');
steps = 1000;
for step = 1:steps
   waitbar(step / steps)
end
close(wt);
xm=10;
ym=10;
sink.x=1;
sink.y=1;
n=1;
p=0.1;
Eo=0.5;
ETX=50*0.000000001;
ERX=50*0.000000001;
Efs=10*0.000000000001;
Emp=0.0013*0.000000000001;
EDA=5*0.000000001;
yd1=33;
```

```
m=0.1;
a=2;
rmax=9000;
do=sqrt(Efs/Emp);
Et=0;
for i=1:1:n
   S(i).xd=rand(1,1)*xm;
   XR(i)=S(i).xd;
   S(i).yd=rand(1,1)*ym;
   YR(i)=S(i).yd;
   S(i).G=0;
   S(i).type='N';
   temp_rnd0=i;
   if (temp_rnd0>=m*n+1)
      S(i).E=Eo;
      E(i)=S(i).E;
      S(i).ENERGY=0;
   end
   if (temp_rnd0<m*n+1)
      S(i).E=Eo*(1+a);
      E(i)=S(i).E;
      S(i).ENERGY=1;
   end
   Et=Et+S(i).E;
end
S(n+1).xd=sink.x;
S(n+1).yd=sink.y;
countCHs=0;
rcountCHs=0;
KMS=1;
countCHs;
rcountCHs=rcountCHs+countCHs;
flag_first_dead=0;
allive=n;
packets_TO_BS=0;
```

```
packets_TO_CH=0;
for r=0:1:rmax
   pnrm=( p/ (1+a*m) );
   padv= ( p*(1+a)/(1+a*m) );
   if(mod(r, round(1/pnrm) )==0)
      for i=1:1:n
         S(i).G=0;
         S(i).cl=0;
      end
   end
   if(mod(r, round(1/padv) )==0)
      for i=1:1:n
         if(S(i).ENERGY==1)
            S(i).G=0;
            S(i).cl=0;
         end
      end
   end
   dead=0;
   dead_a=0;
   dead_n=0;
   for i=1:1:n
      if (S(i).E<=0)

         dead=dead+1;

         if(S(i).ENERGY==1)
            dead_a=dead_a+1;
         end
         if(S(i).ENERGY==0)
            dead_n=dead_n+1;
         end
      end
      if S(i).E>0
         S(i).type='N';
```

```
        if (S(i).ENERGY==0)
        end
        if (S(i).ENERGY==1)
        end
    end
    STATISTICS.DEAD(r+1)=dead;
    STATISTICS.ALLIVE(r+1)=allive-dead;
end
if (dead==1)
    if(flag_first_dead==0)
        first_dead=r
        flag_first_dead=1;
    end
end
countCHs=0;
KMS=1;
for i=1:1:n
    if(S(i).E>0)
        temp_rand=rand;
        if ( (S(i).G)<=0)
            if( ( S(i).ENERGY==0 && ( temp_rand <= ( pnrm / ( 1 - pnrm *
mod(r,round(1/pnrm)) )) ) ) )
                countCHs=countCHs+1;
                packets_TO_BS=packets_TO_BS+1;
                PACKETS_TO_BS(r+1)=packets_TO_BS;
                S(i).type='C';
                S(i).G=100;
                C(KMS).xd=S(i).xd;
                C(KMS).yd=S(i).yd;
                distance=sqrt( (S(i).xd-(S(n+1).xd) )^2 + (S(i).yd-(S(n+1).yd) )^2 );
                C(KMS).distance=distance;
                C(KMS).id=i;
                X(KMS)=S(i).xd;
                Y(KMS)=S(i).yd;
                KMS=KMS+1;
```

```
                distance;
                if (distance>do)
                    S(i).E=S(i).E-       (       (ETX+EDA)*(4000)       +       Emp*4000*(
distance*distance*distance*distance ));
                end
                if (distance<=do)
                    S(i).E=S(i).E- ( (ETX+EDA)*(4000)  + Efs*4000*( distance * distance
));
                end
            end
            if( ( S(i).ENERGY==1 && ( temp_rand <= ( padv / ( 1 - padv *
mod(r,round(1/padv)) )) ) )  )
                countCHs=countCHs+1;
                packets_TO_BS=packets_TO_BS+1;
                S(i).type='C';
                S(i).G=100;
                distance=sqrt( (S(i).xd-(S(n+1).xd) )^2 + (S(i).yd-(S(n+1).yd) )^2 );
                X(KMS)=S(i).xd;
                Y(KMS)=S(i).yd;
                KMS=KMS+1;
                distance;
                if (distance>do)
                    S(i).E=S(i).E-       (       (ETX+EDA)*(4000)       +       Emp*4000*(
distance*distance*distance*distance ));
                end
                if (distance<=do)
                    S(i).E=S(i).E- ( (ETX+EDA)*(4000)  + Efs*4000*( distance * distance
));
                end
            end

        end
    end
  end
end
```

```
NIR=sum(Nodes);
K=str2num(nn);
pause(2);disp('**********************************');
%disp('Please enter encryption key');
disp('**********************************');
%jj = input('Enter encryption key: ', 's');
jj='25';
pwd=str2num(jj);
 if pwd==K
    %disp('CORRECT KEY');
    h = waitbar(0,'Please wait...');
steps = 2000;
for step = 1:steps
   waitbar(step / steps)
end
close(h);
matlablib;
 else
    error 'Invalid Key';
end
```

ants_cost

```
function [cost,f]=ants_cost(m,n,d,at,el)

for i=1:m
   s=0;
   for j=1:n
      s=s+d(at(i,j),at(i,j+1));
   end
   f(i)=s;
end
cost=f;
f=f-el*min(f);%elimination of common cost.
```

ants_cycle

```
function [at]=ants_cycle(app,m,n,h,t,alpha,beta);
for i=1:m
   mh=h;
   for j=1:n-1
      c=app(i,j);
      mh(:,c)=0;
      temp=(t(c,:).^beta).*(mh(c,:).^alpha);
      s=(sum(temp));
      p=(1/s).*temp;
      r=rand;
      s=0;
      for k=1:n
         s=s+p(k);
         if r<=s
            app(i,j+1)=k;
            break
         end
      end
   end
end
at=app;% generation of ants tour matrix during a cycle.
```

ants_information

```
function [x,y,d,t,h,iter,alpha,beta,e,m,n,el]=ants_information;
iter=10;%number of cycles.

m=20;%number of ants.
x=[8 0 -1 2 4 6 3 10 2.5 -5 7 9 11 13];
y=[2 4 6 -1 -2 0.5 0 3.7 1.8 1 0 4 3 2];%take care not to enter iterative points.
n=length(x);%number of nodes.
for i=1:n%generating link length matrix.
```

```
    for j=1:n
        d(i,j)=sqrt((x(i)-x(j))^2+(y(i)-y(j))^2);
    end
end
e=.1;%evaporation coefficient.
alpha=1;%order of effect of ants' sight.
beta=5;%order of trace's effect.
for i=1:n%generating sight matrix.
    for j=1:n
        if d(i,j)==0
            h(i,j)=0;
        else
            h(i,j)=1/d(i,j);
        end
    end
end
t=0.0001*ones(n);%primary tracing.
el=.96;%coefficient of common cost elimination.
```

ants_primaryplacing

```
%global m;
function [app]=ants_primaryplacing(m,n);
rand('state',sum(100*clock));
for i=1:m
    app(i,1)=fix(1+rand*(n-1));%ants primary placing.
end
```

ants_traceupdating

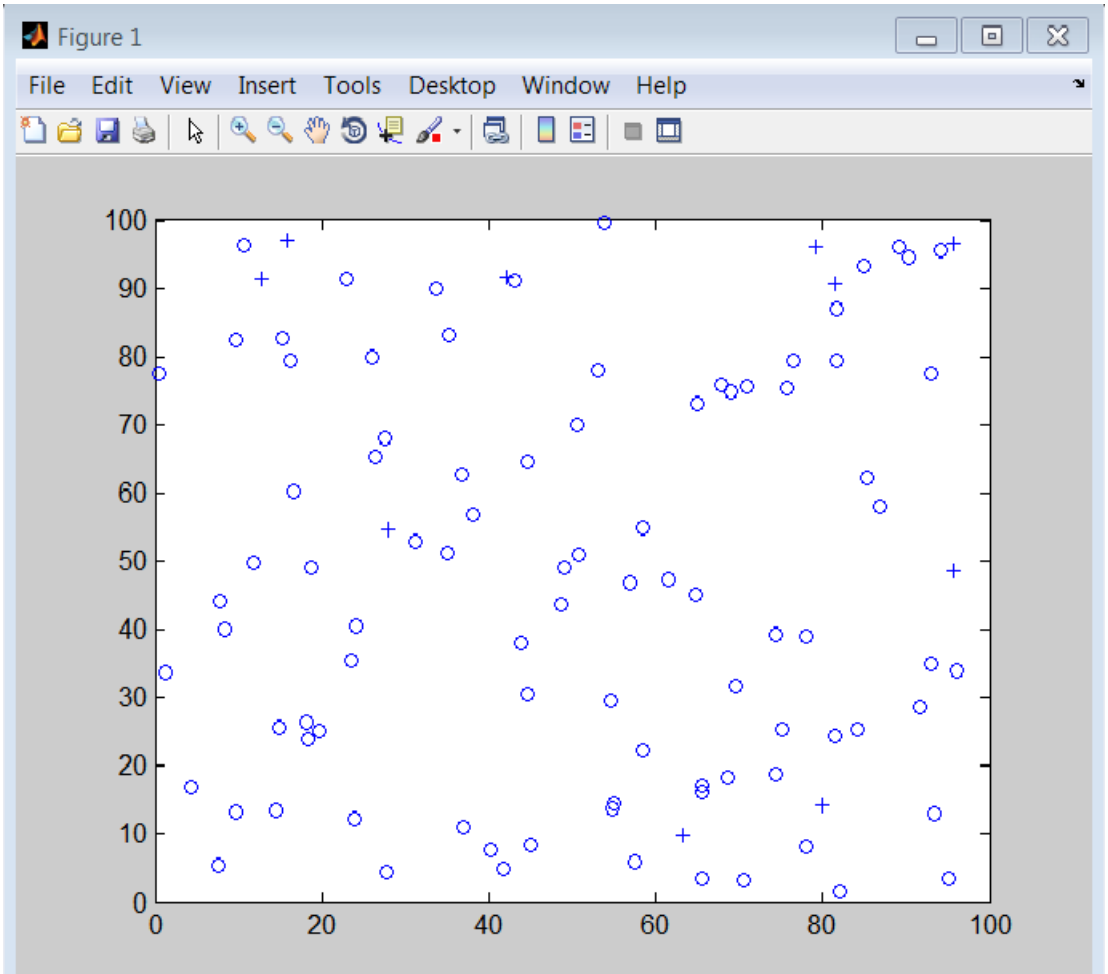
```
function [t]=ants_traceupdating(m,n,t,at,f,e)

for i=1:m
    for j=1:n
```

```
        dt=1/f(i);
        t(at(i,j),at(i,j+1))=(1-e)*t(at(i,j),at(i,j+1))+ dt;          %updating traces.
    end
end
```

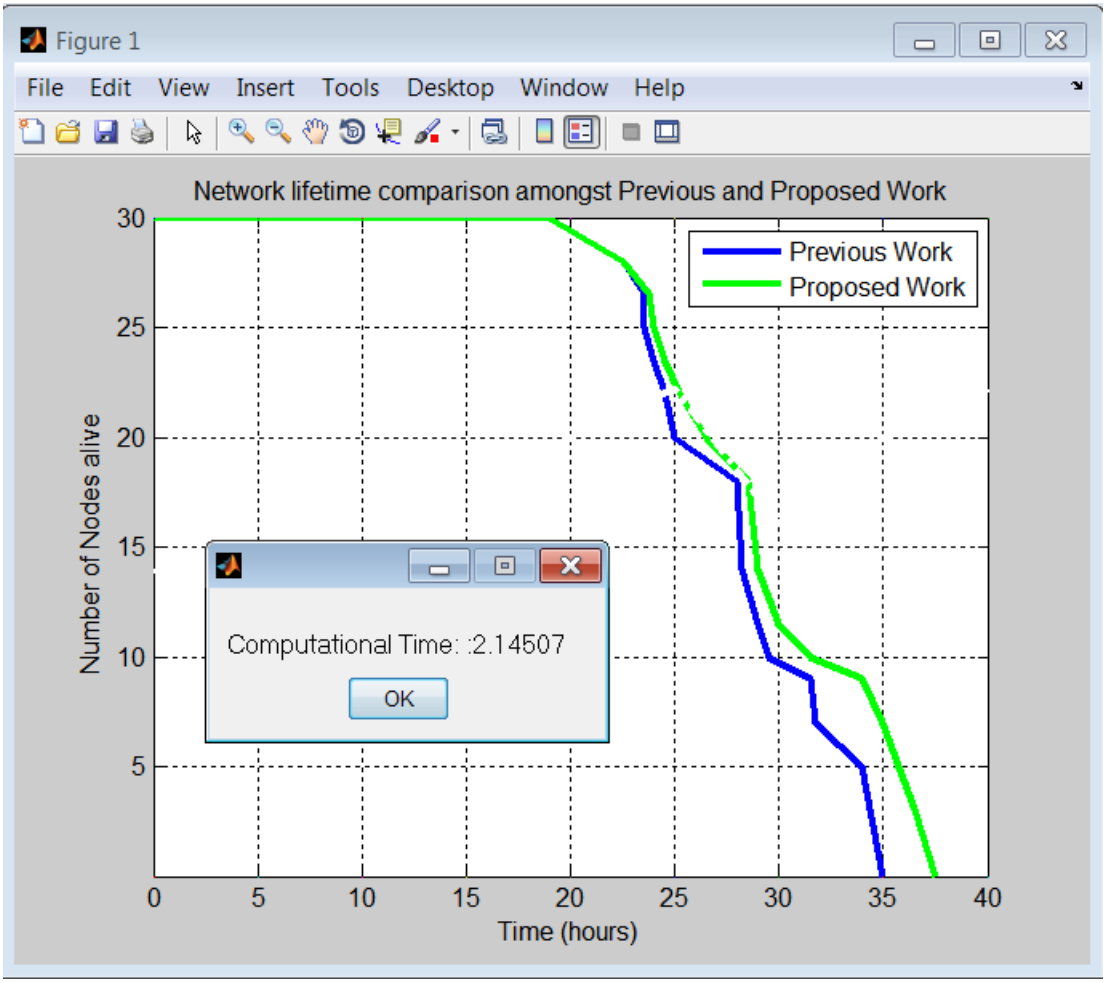
starting_time

```
function [set] = starting_time()
time=cputime/60;
set=time;
end
```

OUTPUT



WIRELESS MESH NETWORK



NETWORK LIFETIME COMPARISON AND COMPUTATIONAL TIME OF PROPOSED ALGORITHM

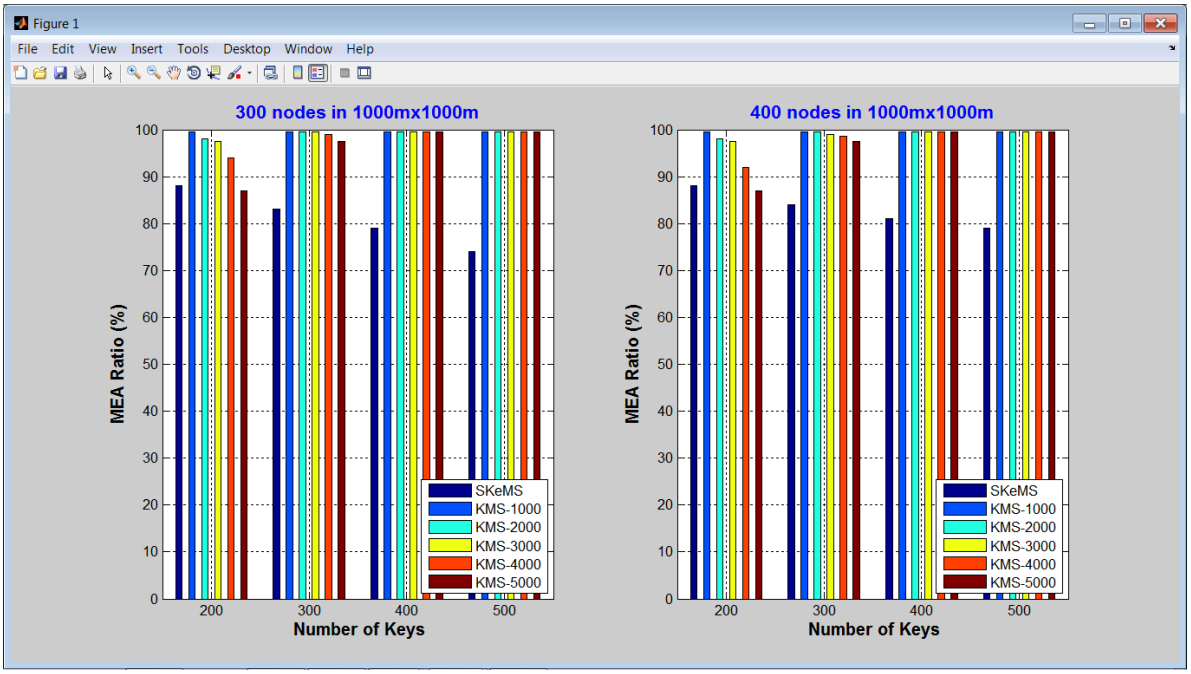


FIGURE 1

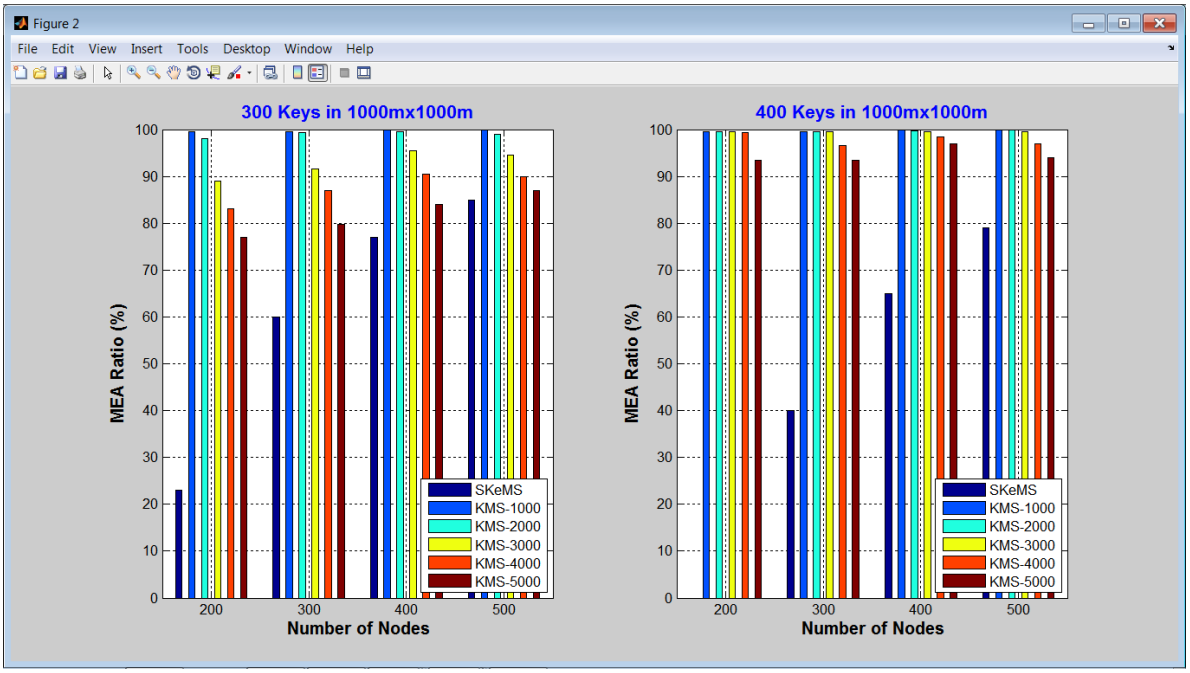


FIGURE 2

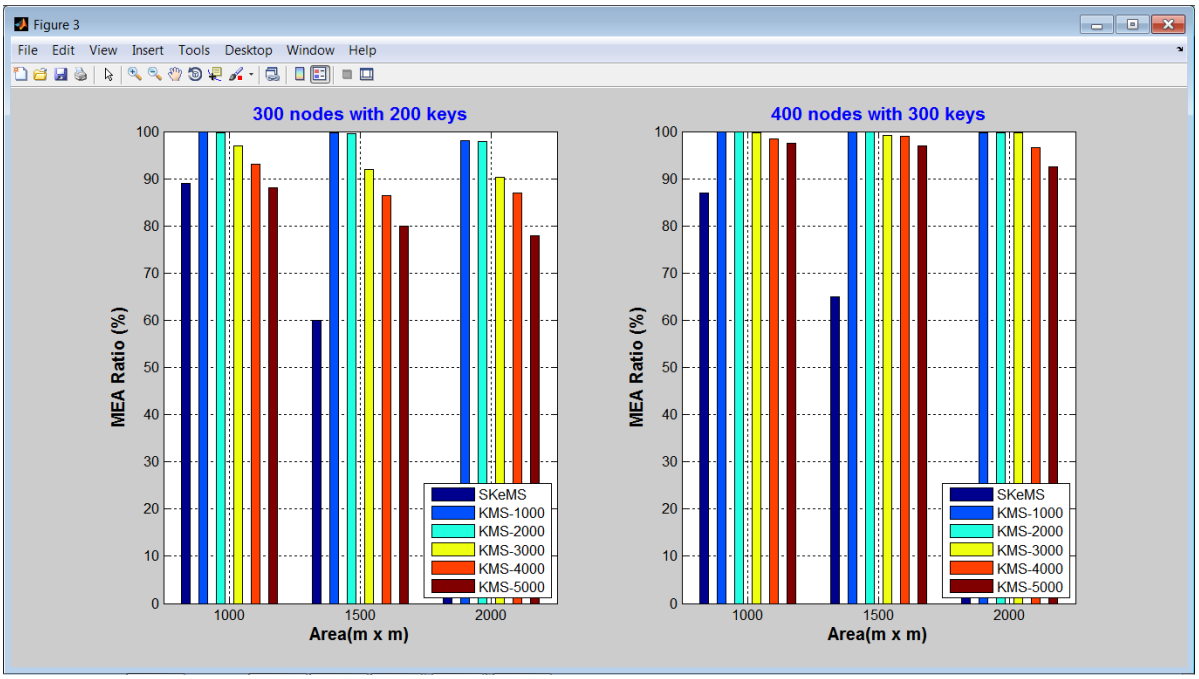


FIGURE 3

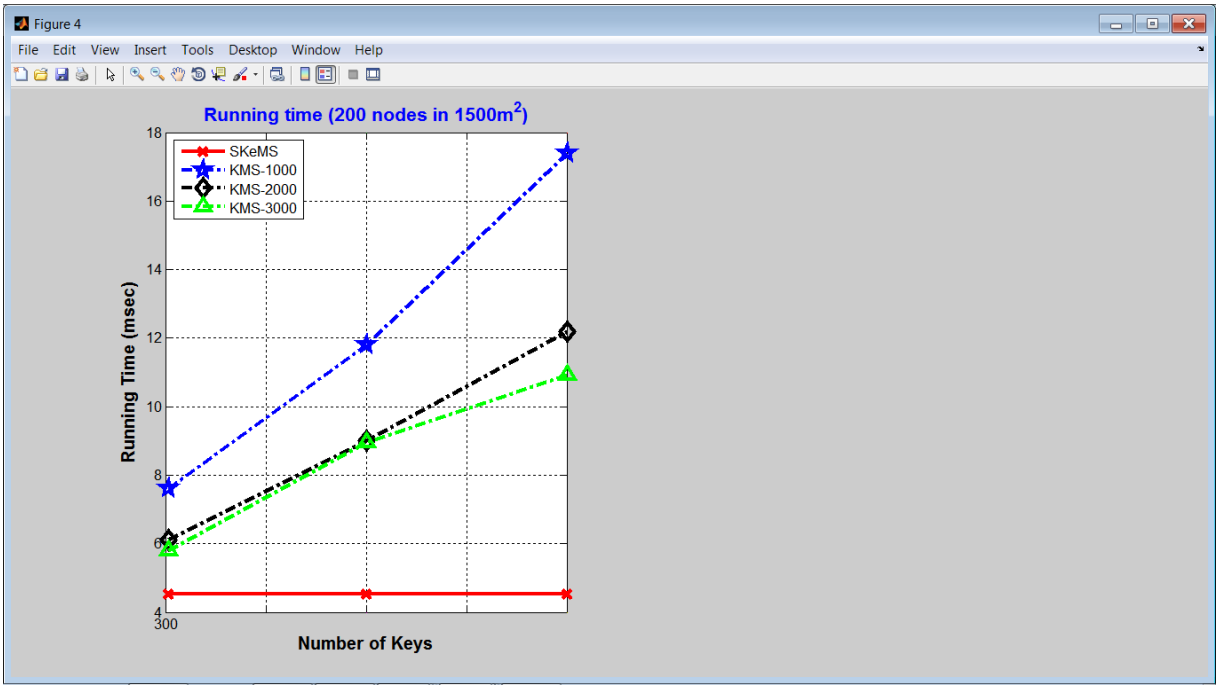


Figure 4
File Edit View Insert Tools Desktop Window Help
Running time (200 nodes in 1500m2)
SKeMS
KMS-1000
KMS-2000
KMS-3000
Running Time (msec)
Number of Keys
18
16
14
12
10
8
6
4
300

FIGURE 4

RESULT ANALYSIS

To show how our scheme performs, I implemented my algorithm (SKeMS in the figures above), and used it to compare it with previous work (KMS in the figures above). I took a static wireless mesh network with n nodes uniformly distributed in a square ground. The results shown are the average of 5 test runs for various scenarios.

The first metric that I have used to evaluate the performance is *malicious eavesdropping ability ratio* (MEA ratio in the figures above), which is computed as the neighbour compromise ability divided by the number of neighbouring nodes that are prone to eavesdropping attack. Having smaller MEA ratio proves that the network is more secure and is also more resistant against the malicious eavesdropping attacks. In the first scenario, I randomly distributed 300 and 400 nodes in a 10x10 square meters ground.

For the KMS algorithm an increase in the pool size minimizes the MEA ratio. For example, with 300 keys chosen from a pool of 1000, we get an MEA ratio of 98%, while with 5000 keys pool with the same amount of keys we get a ratio of 87%. Compared to my algorithm, the results show that my algorithm outperforms the KMS algorithms in all different tested pool sizes. The same results can be seen in Fig. 1(b), where I distribute 400 nodes in a 1000m square field.

Fig. 2 shows the results of the second scenario where I have studied the algorithm's performance with different number of nodes in the same area size. In Fig. 2(a) I have tested the performance with different number of nodes. I tested the effect of giving 300 keys on the MEA ratio on the given network. The results showed that, by applying my algorithm, the MEA ratio increases while the number of nodes increases because of more common shared keys among the nodes in the neighbourhood. But my algorithm's ratio is better as compared to the ratio when applying the previous algorithm.

In Fig. 3 the effect of the network density is tested on both algorithms by changing the area size, but keeping the number of nodes fixed. Fig. 3(a) displays the scenario where 300 nodes are distributed in different area sizes using 200 available keys. According to the results, in a sparse network the no. of neighbouring nodes decreased with increasing area size, this led to a decrease in the number of nodes that are affected by the eavesdropping attack. The same trend can be noticed in Fig. 3(b) with 400 nodes and 300 keys.

The second metric used is the running time that is defined as the running time that the algorithm took to distribute the keys to the nodes. I tested my algorithm and compared it with the previous algorithm in one scenario shown in Fig.4. I took 200 nodes that were randomly distributed in a 1000 square meters ground. I assigned 300, 400 and 500 keys for both the algorithms. Since the previous algorithm depends on the pool size, it can be seen that it takes more time to distribute the keys from 1000 keys pool and find the MCA for the network, because of having more number of neighbouring nodes that have more common keys. Compared to my algorithm, it can be seen that my algorithm outperforms the previous work in all tested cases.

CONCLUSION

In this project, I have proposed a Secure Key Management Algorithm, and thus provided with an effective algorithm that helps provide a key assignment to a WMN. My algorithm is resilient against eavesdropping attacks. The simulation gave results that proved that my algorithm performs good in terms of smaller MEA ratio and lesser running time. To conclude, in this project, I showed that a good key management algorithm can guarantee a more secure network.

FUTURE WORK

The power of self healable and self organisable is the crucial factor in wireless mesh networks that helps reduce the network maintenance cost and complexity. It provides with the backbone power using which a user can connect to the internet. Wireless mesh networks are a very promising technology for wireless networking of the next generation.

Wireless mesh networks have increased the capability and reliability of the modern ad hoc networks. But we still have many problems in wireless mesh networks that need to be improved upon. Though the existing schemes are effective at particular layers but there is a requirement to have a detailed mechanism which can prevent from the security attacks at all protocol layers.

REFERENCES

[1] Farah Kandah, Weiyi Zhang, Xiaojiang Du, Yashaswi Singh, A Secure Key management Scheme in Wireless Mesh Networks, Communications (ICC), 2011 IEEE
International Conference, June 2011.

[2] I. F. Akyildiz, X. Wang, W. Wang, Wireless mesh networks: a survey; Elsevier Journal of Computer Networks, vol.47, Issue.4, pp.445-487, 2005.

[3] N. Asokan, P. Ginzboorg, Key Agreement in Ad Hoc Networks; Computer Communications, vol.23, pp.1627-1637, 2000.

[4] M. Cagalj, J. Hubaux, C. Enz, Minimum-energy broadcast in all-wireless networks: NP-completeness and distribution issues; ACM MobiCom'02, Atlanta, Georgia,
USA.

[5] S. P. Chan, R. Poovendran, M. T. Sun, A key management scheme in distributed sensor networks using attack probabilities; IEEE GLOBECOM' 05, vol.2, pp.5, St. Louis, MO, USA.

[6] X. Du, Y. Xiao, M. Guizani, H. H. Chen, An effectivekey management scheme for heterogeneous sensor networks; Ad Hoc Networks, Special Issues in Sensor
and Ad Hoc Networks, vol.5, Issue.1, pp.24-34, 2007. network; IEEE INFOCOM'05, vol.3, pp.2223- 2234, Miami, Fl., USA.

[7] L. Eschenauer, V. D. Gligor, A key-management scheme for distributed sensor networks; ACM CCS'02, Washington, DC, USA.

[8] P. Loree, K. Nygard, X. Du, An efficient post-deployment key establishment scheme for heterogeneous sensor networks; IEEE GLOBECOM'09, Honolulu, Hawaii, USA. [15] Dr. M.S.Aswal, Paramjeet Rawat, Tarun Kumar, Threats and Vulnerabilities in Wireless Mesh Networks, International Journal of Recent Trends in Engineering, Vol 2, No. 4, November 2009

[9] A. Raniwala, T. Chiueh, Architecture and algorithms for an IEEE 802.11- based multi-channel wireless mesh

[10] A. Raniwala, K. Gopalan, T. Chiueh, Centralized channel assignment and routing algorithms for multi-channel wireless mesh networks; ACM MobiCom'04, Vol.8, Issue.2, pp.50-65, Philadelphia, PA, USA.

[11] J. Shi, R. Zhang, and Y. Zhang, Secure range queries in tried sensor networks; IEEE INFOCOM'09, pp.945-953, Rio de Janeiro, Brazil.

[12] J. Tang, G. Xue, W. Zhang, Interference-Aware Topology Control and QoS Routing in Multi-Channel Wireless Mesh Networks; ACM Mobi-Hoc'05, pp. 68-77, Urbana-champaign, IL, USA.

[13] W. Zhang, F. Kandah, J. Tang, K. Nygard, Interference-Aware Robust Topology Design in Multi-Channel Wireless Mesh Networks; IEEE CCNC'10, pp.6-10, LAS Vegas, NV, USA.

[14] X. Zhao, Y. Lv, T. H. Yeap, B. Hou, A Novel Authentication and Key Agreement Scheme for Wireless Mesh Networks; In Proceedings of NCM'09, pp.471-474,
Washington, DC, USA.