www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 2 Issue 4 April, 2013 Page No. 1394-1425

Active Source Routing Protocol for Mobile Network:

Anil Kumar Prasad , Anamika Bhusan, Divya Gupta
B.tech(final year) computer science & engineering

Abstract

An ad-hoc mobile network is a collection of mobile nodes that are dynamically and arbitrarily located in such a manner that the interconnections between nodes are capable of changing on a continual basis. The primary goal of such an ad-hoc network routing protocol is correct and efficient route establishment between a pair of nodes so that messages may be delivered in a timely manner. LAR is an on-demand protocol who is based on the DSR(Dynamic Source Routing). The Location Aided Routing protocol uses location information to reduce routing overhead of the ad-hoc network! Normally the LAR protocol uses the GPS(Global Positioning System) to get these location information's. With the availability of GPS, the mobile hosts knows there physical location.Ad hoc networks are a new wireless networking paradigm for mobile hosts. Unlike traditional mobile wireless networks, ad hoc networks do not rely on any fixed infrastructure. Instead, hosts rely on each other to keep the network connected. The military tactical and other security-sensitive operations are still the mainapplications of ad hoc networks, although there is a trend to adopt ad hoc networks for commercial uses due to their unique properties. One main challenge in design of these networks is their vulnerability to security attacks. In this paper, we study the threats an ad hoc network faces and the security goals to be achieved. We identify the new challenges and opportunities posed by this new networking environment and explore new approaches to secure its communication. In particular, we take advantage of the inherent redundancy in ad hoc networks — multiple routes between nodes — to defend routing against denial of service attacks.

Key word:

Interconnection, dynamic source routing,vulnerability,security-sensitive, global positioning system, denial.

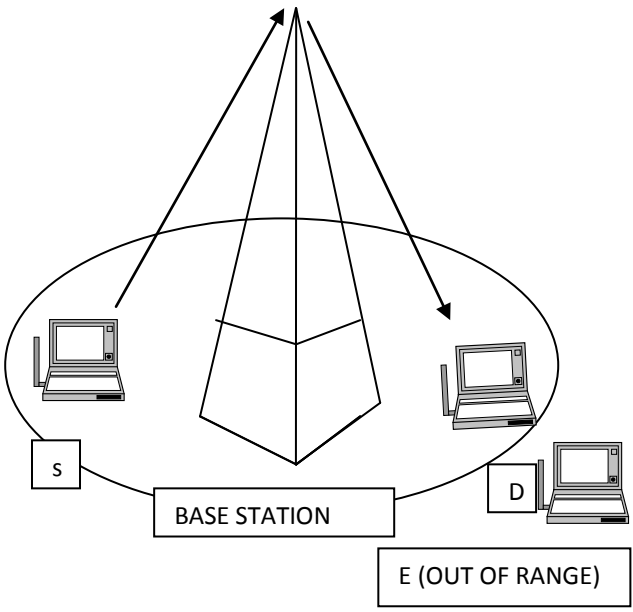
INTRODUCTION

An ad-hoc mobile network is a collection of mobile nodes that are dynamically and arbitrarily located in such a manner that the interconnections between nodes are capable of changing on a continual basis. The primary goal of such an ad-hoc network routing protocol is correct and efficient route establishment between a pair of nodes so that messages may be delivered in a timely manner. LAR is an on-demand protocol who is based on the DSR(Dynamic Source Routing). The Location Aided Routing protocol uses location information to reduce routing overhead of the ad-hoc network! Normally the LAR protocol uses the GPS(Global Positioning System) to get these location information's. With the availability of GPS, the mobile hosts knows there physical location.Ad hoc networks are a new wireless networking paradigm for mobile hosts. Unlike traditional mobile wireless networks, ad hoc networks do not rely on any fixed

infrastructure. Instead, hosts rely on each other to keep the network connected. The military tactical and other security-sensitive operations are still the main applications of ad hoc networks, although there is a trend to adopt ad hoc networks for commercial uses due to their unique properties. One main challenge in design of these networks is their vulnerability to security attacks. In this paper, we study the threats an ad hoc network faces and the security goals to be achieved. We identify the new challenges and opportunities posed by this new networking environment and explore new approaches to secure its communication. In particular, we take advantage of the inherent redundancy in ad hoc networks — multiple routes between nodes — to defend routing against denial of service attacks. We also use replication and new cryptographic schemes, such as threshold cryptography, to build a highly secure and highly available key management service, which forms the core of our security framework.Ad hoc networks are a new paradigm of wireless communication for mobile hosts (which we call nodes). In an ad hoc network, there is no fixed infrastructure such as base stations or mobile switching centers. Mobile nodes that are within each other's radio range communicate directly via wireless links, while those that are far apart rely on other nodes to relay messages as routers. Node mobility in an ad hoc network causes frequent changes of the network topology. Figure 1 shows such an example: initially, nodes A and D have a direct link between them. When D moves out of A's radio range, the link is broken. However, the network is still connected, because A can reach D through C, E, and F. Military tactical operations are still the main application of ad hoc networks today. For example, military units (e.g., soldiers, tanks, or planes), equipped with wireless communication devices, could form an ad hoc network when they roam in a battlefield. Ad hoc networks can also be used for emergency, law enforcement, and rescue missions. Since an ad hoc network can be deployed rapidly with relatively low cost, it becomes an attractive option for commercial uses such as sensor networks or virtual classrooms.

EXISTING SYSTEM

The first one is to introduce a third fixed party (a base station) that will hand over the offered traffic from a station to another, as illustrated in Figure 1. The same entity will regulate the attribution of radio resources, for instance. When a node S wishes to communicate to a node D, the former notifies the base station, which eventually establishes a communication with the destination node. At this point, the communicating nodes do not need to know of a route for one to each other. All that matters is that both nodes source and destination are within the transmission range of the base station. If one of them fails to fulfill this condition, the communication will abort.



Infra structured Network

Fig.1. infra structured network.

Here the base station's range is illustrated by the oval. The two nodes S and D which want to

communicate are in the range of the base station. S send the message to the base station which in turn forwards it to destination node D. Thus communication is carried out with help of a base station. All messages have to pass through the base station. Node E is out of the range of the base station this prevents it from communicating to other nodes in the network. When node E wants to communicate to any node in the network it has to contact the base station. Since it is out of range communication is not possible. What happens if the base station is unavailable? Or what happens if we are in a situation where such an infrastructure does not exist at the first place? The answer is that we simply do not communicate! This is where the second approach is useful. However that this form of centralized administration is very popular among wide cellular networks such as GSM etc.

PROPOSED SYSTEM

The second approach, called the Ad-Hoc, does not rely on any stationary infra structure. The concept behind these infra-structureless networks is the collaboration between its participating members, i.e, instead of making data transit through a fixed base station, nodes consequentially forward data packets from one to another until a destination node is finally reached. Typically, a packet may travel through a number of network points before arriving at its destination.Ad-hoc networking introduces a completely new flavor of network formation. The term Ad-Hoc means, in this instance, a type instantaneous network connecting various mobile devices without the intervention of fixed infrastructure. The routers and hosts are free to move randomly and organize themselves in an arbitrary fashion, thus the network topology changes rapidly and unpredictably. Absence of a supporting structure in mobile ad-hoc networks, to a certain extent, invalidates almost all of the existing techniques developed for routine network controls in the existing wireless networks.

A MANET consists of mobile platforms (e.g., a router with multiple hosts and wireless communications devices)--herein simply referred to as "nodes"--which are free to move about arbitrarily. The nodes may be located in or on airplanes, ships, trucks, cars, perhaps even on people or very small devices, and there may be multiple hosts per router. A MANET is an autonomous system of mobile nodes. The system may operate in isolation, or may have gateways to and interface with a fixed network. MANET nodes are equipped with wireless transmitters and receivers using antennas which may be unidirectional (broadcast), highly directional (point-to-point), possibly steerable, or some combination thereof. At a given point in time, depending on the nodes' positions and their transmitter and receiver coverage patterns, transmission power levels and co-channel interference levels, a wireless connectivity in the form of a random, multi hop graph or "Ad Hoc" network exists between the nodes. This ad hoc topology may change with time as the nodes move or adjust their transmission and reception parameters.

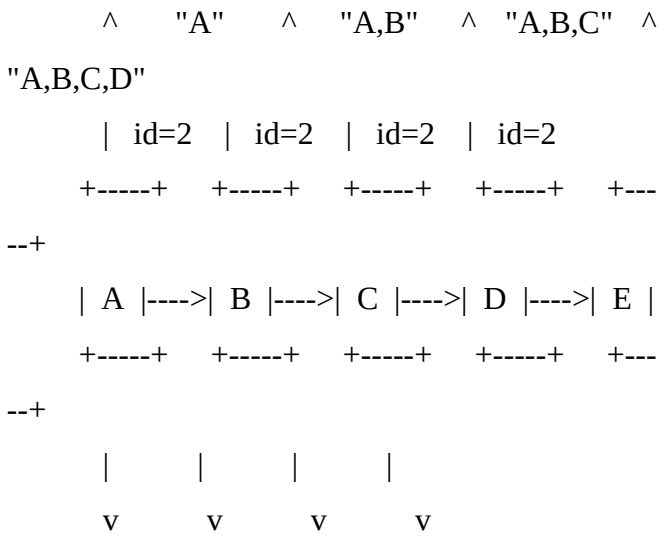
LITERATURE SURVEY

In literature survey, we have found some problem in network security and how can provides a solution for securing routing in the managed-open environment and also from a variety of attacks. To solve this problem we use a protocol ie, ARAN (Authenticated Routing for Ad hoc Network) provides a solution for securing routing in the managed-open environment.ARANprovidesauthenticationandnonrepudiationservicesusingpredeterminedcryptographiccertifcates that guarantees end-to-end authentication. In doing so, ARAN limits or prevents attacks that can afflict other insecure protocols. ARAN is a simple protocol that does not require significant additional work from nodes within the group. Our simulations show that ARAN is as efficient as AODV in discovering and maintaining routes, at the cost of using larger routing packets which result in a higher overall routing load, and at the cost of higher latency in route discovery because of the cryptographic computation that must occur.

PROBLEM FORMULATION

When some source node originates a new packet addressed to some destination node, the source node places in the header of the packet a "source route" giving the sequence of hops that the packet is to follow on its way to the destination. Normally, the sender will obtain a suitable source route by searching its "Route Cache" of routes previously learned; if no route is found in its cache, it will initiate the Route Discovery protocol to dynamically find a new route to this destination node. In this case, we call the source node the "initiator" and the destination node the "target" of the Route Discovery.

For example, suppose a node A is attempting to discover a route to node E. The Route Discovery initiated by node A in this example would proceed as follows:



To initiate the Route Discovery, node A transmits a "Route Request" as a single local broadcast packet, which is received by (approximately) all nodes currently within wireless transmission range of A, including node B. Each Route Request also contains a record listing the address of each intermediate node through which this particular copy of the Route Request has been forwarded. This route record is initialized to an empty list by the initiator of the Route Discovery. In this example, the route record initially lists only node A.

When another node receives this Route Request (such as node B in this example), if it is the target of the Route Discovery, it returns a "Route Reply" to the initiator of the Route Discovery, giving a copy of the accumulated route record from the Route Request; when the initiator receives this Route Reply, it caches this route in its Route Cache for use in sending subsequent packets to this destination.Otherwise, if this node receiving the Route Request has recently seen another Route Request message from this initiator bearing this same request identification and target address, or if this node's own address is already listed in

the route record in the Route Request, this node discards the Request. Otherwise, this node appends its own address to the route record in the Route Request and propagates it by transmitting it as a local broadcast packet (with the same request identification). In this example, node B broadcast the Route Request, which is received by node C; nodes C and D each also, in turn, broadcast the Request, resulting in a copy of the Request being received by node E.

EXTRA SOLUTION

DATA SECURITY USING RSA

- Source sends the encrypted data packet to the destination through the route discovered.
- Destination decrypts the data packet received from the source and sends the acknowledgement.

KEY GENERATION:

i. Select two prime numbers p and q such that p not equal to q ii. Calculate n=p x q

iii. Calculate ø (n)=(p-1)(q-1)

iv. Select integer e such that gcd (ø(n),e)=1;1<e< ø (n)

v. Calculate d such that d=e¹mod ø(n)

vi. Public key KU={e,n}

vii. Private key KR={d,n}

ENCRYPTION:

The plain text M (M<n) is encrypted to cipher text using public key e. C=M pow e (mod n)

DECRYPTION

The cipher text C is decrypted to plain text using private key d.

M=C pow d (mod n)

DATA SECURITY

Message Encryption

Message Decryption

ALGORITHM STEP BY STEP DESCRIPTION:

1. **If the user wants to send data:**
    - Get the Destination identifier and the encrypted data to be transferred.
    - Initialize the buffer with the encrypted data to be transferred.
    - Setup a Request Zone.
    - Build a Route Request packet having the information about the source and the Destination identifiers, and the Request Zone information.
    - Broadcast the Route Request to its neighbors.
    - Setup a timer for receiving Route Reply.
2. **If the node receives a packet**
    - Find the type of the packet received.
    - Depending on the type of packet received do one of the following processes.
    - Process Route Request.
    - Process Route Reply.
    - Process Data Packet.
    - Process Decryption.
    - Process Acknowledgement.
    - Process Route Disconnect.
    - Process Route Disconnect reply.
    - Process Timer Run Out.

DATA SECURITY USING RSA

- Source sends the encrypted data packet to the destination through the route discovered.
- Destination decrypts the data packet received from the source and sends the acknowledgement.

INPUT AND OUTPUT PARAMETER

Input parameters

- At source, encrypted data packets are sent with destination address and route request.

Output parameters

- Receiving positive acknowledgement with efficient and reliable packet transmission.

MODULE SPECIFICATION

This project is divided into 2 modules.

1. password module

2. Request module

1.password module:

This module authenticates the user to enter. The user can type the text they want to send or browse the file they want to attach .this message is transferred to the destination after the data is encrypted.RSA algorithm is used for encryption.

2.Request module :

this module describes the wireless network communication protocol mechanism .

SOFTWARE REQUIREMENTS

Language : Java1.3

Front End Tool: Swing

Operating System: Windows 98.

HARDWARE REQUIREMENTS

Processor : Intel Pentium III Processor

Random Memory: 128M

Hard Disk :20GB

Processor Speed: 300 min

SOURCE CODE

Password.java

```
import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.WindowAdapter;

import java.awt.event.WindowEvent;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JOptionPane;

import javax.swing.JPasswordField;

import javax.swing.JTextField;

import java.sql.*;

public class Password  extends JFrame implements ActionListener

{

                                                JPasswordField

password;
```

```
JTextField Text1;
JLabel
login,pass,hea1;
JButton
ok,cancel;
final String log =
"routing";
final String
passW = "location";
JFrame fpass;

Password()
{

fpass = new
JFrame("Login");

fpass.getContentPane().setLayout(null);

hea1 = new
JLabel("LOG IN SCREEN");
login = new
JLabel("USER-ID");
Text1 = new
JTextField(10);
pass = new
JLabel("PASSWORD");
password =
new JPasswordField(10);
ok = new
JButton("OK");
cancel = new
JButton("CANCEL");

hea1.setBounds(
130,10,100,50);

login.setBounds(
70,40,70,50);

Text1.setBounds
(157,55,125,20);

pass.setBounds(
70,70,70,50);

password.setBo
unds(158,83,125,20);

ok.setBounds(80
,130,90,20);
```

```
                cancel.setBound                                 ok.addActionList
s(180,130,110,20);                                              ener(this);

                                                                cancel.addActio
                                                                nListener(this);

                fpass.getContent
Pane().add(hea1);
                                                                fpass.setSize(40
                                                                0, 250);
                fpass.getContent
Pane().add(login);
                                                                fpass.setVisible(true);

                fpass.getContent
Pane().add(Text1);
                                                                }

                fpass.getContent                                public void
                                                                actionPerformed(ActionEvent e)
Pane().add(pass);
                                                                {

                fpass.getContent                                String s =
Pane().add(password);                                           e.getActionCommand();

                fpass.getContent                                if(s.equals("OK"))
Pane().add(ok);
                                                                {

                fpass.getContent
Pane().add(cancel);                                             ////



                                                                try

                                                                {
```

```
int count = 1;

Class.forName("
sun.jdbc.odbc.JdbcOdbcDriver");

Connection con
=
DriverManager.getConnection("Jdbc:Odbc:connection
");

Statement
st=con.createStatement();

String str =
"select * from login";

ResultSet rs =
st.executeQuery(str);

while(rs.next())

{

if(((Text1.getText()).equals(rs.getString(1)))
&&

((password.getT
ext()).equals (rs.getString(2))))

{

count = 0;

break;

}

else

count = 1;

}

if(count ==1)

{

JOptionPane.sho
wMessageDialog(null, "Invaild UserName and
PassWord","Alert", JOptionPane.ERROR_MESSAGE);

Text1.setText("")
;

password.setTex
t("");

fpass.setVisible(t
rue);

}
```

```
}

public static
void main(String ar[])
{
Password ps =
new Password();

ps.addWindowLi
stener( new WindowAdapter()
{

public void windowClosing(WindowEvent e)
{

System.exit(0);
}
}
);
//addWindowListener

}//main method
}

else
{

fpass.setVisible(false);
Sender
sen=new Sender();

}

}catch(Exception
e1){System.out.println(e1);}
////

}
else
if(s.equals("CANCEL"))
{

System.exit(0);
}
```

Sender.java

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.io.*;
import java.net.*;
import java.util.*;

class Sender implements ActionListener
{
        JFrame jf;
        static JTextArea jta=new JTextArea();
        JButton jb,browse;
        JTextField jt1,jt2,jt3;
        JLabel jl1,jl2,jl3,jl4;
        JScrollPane jsp;
        Container c;
        static String destination="";
        static String ttt;
        static String tte;
        static Vector vvv=new Vector();
        static int i=0;

        Sender()
        {

            jf=new JFrame("Sender");
            jb=new JButton("Send");
            jt1=new JTextField();
            jt2=new JTextField();
            jt3=new JTextField();
            jl1=new JLabel("Destination");
            browse=new JButton("Browse");
            jl2=new JLabel("Request Zone Limit");
            jsp=new JScrollPane(jta,ScrollPaneConstants.VERTICAL_SCROLLBAR_ALWAYS,ScrollPaneConstants.HORIZONTAL_SCROLLBAR_AS_NEEDED);

            jf.setSize(800,800);
            c=jf.getContentPane();
            c.setLayout(null);
            c.add(jsp);
            c.add(jb);
            c.add(browse);
            c.add(jt1);
            c.add(jl1);
            c.add(jl2);
```

```
jsp.setBounds(100,100,400,400);

jb.setBounds(550,350,100,25);

jt1.setBounds(550,150,100,25);

browse.setBounds(550,250,100,25);

jl1.setBounds(550,100,100,25);

jb.addActionListener(this);

browse.addActionListener(this);

jf.setVisible(true);

}

public void actionPerformed(ActionEvent ae)

{

destination=jt1.getText();

ttt=jta.getText();

try

{

FileOutputStream
fop=new FileOutputStream("msg.txt");

fop.write(ttt.getBytes());

ObjectOutputStream
oop=new ObjectOutputStream(fop);

}

catch(Exception e){}

// vvv.add(ttt);

System.out.println("The vect
size is "+vvv.size());

if(jb==ae.getSource())

{

try

{

System.out.println("The String
sent is :"+ttt);

RSAKey rsa=new
RSAKey();

}

catch(Exception e)

{

System.out.println("Error : "+e);

}

}

if((JButton)browse==ae.getSource())

{

JFileChooser fc = new
JFileChooser();

int option =
fc.showOpenDialog(jf);

if(option ==
JFileChooser.APPROVE_OPTION)
```

```
                {
                        try
                        {
                                String sf=fc.getSelectedFile().getAbsolutePath();

                        FileInputStream in = new FileInputStream(sf);
                                byte str[] = new byte[in.available()];

                        in.read(str,0,str.length);

                        jta.setText(new String (str));
                        }

                        catch(Exception e){}
                        }
                   }
}

        public static void Socket2(int y,String tt)throws Exception
                {
                        FileInputStream fis=new FileInputStream("path"+y+".txt");
                        ObjectInputStream ois=new ObjectInputStream(fis);
                        Vector d=(Vector)ois.readObject();
                        int siz=d.size();
                        System.out.println("The size of the vector is ::"+siz);
                        for(int j=0;j<siz;j++)
                        {
                        System.out.print("++__++--
                        >>"+d.get(j));

                        }
                        if(siz==2)
                        {

                        Socket soc=new Socket((String)d.get(1),8888);
                        ObjectOutputStream oos=new ObjectOutputStream(soc.getOutputStream());
                        oos.writeObject("file");
                        oos.writeObject(tt);
                        }
                        else
                        {
                        d.removeElementAt(0);
                        int m=d.size();
                        System.out.println("The size after removing the element..."+m);
                        Socket soc=new Socket((String)d.get(1),8888);
```

```
                    ObjectOutputStream
                                oos=new
        ObjectOutputStream(soc.getOutputStream());
                    oos.writeObject("file++");
                    oos.writeObject(tt);
                    oos.writeObject(d);


                                }

  }
}
```

RSA KEY.java

```
import javax.swing.*;
import java.awt.event.*;

class RSAKey implements ActionListener
{
        JButton jb,jb1,jb2;
        JFrame jf;

        RSAKey()
        {
                jf=new JFrame("RSA KEY");
                JInternalFrame jf1=new
JInternalFrame("KEY Value");
                //JLabel jl=new JLabel("Exponent
Value:");
                //JLabel jl1=new JLabel("N Value:");
                //JPasswordField jt=new
        JPasswordField();
                //JPasswordField jt1=new
        JPasswordField();
                jb=new JButton("RSAKeyGen");
                jb1=new JButton("Send");
                jb2=new JButton("Exit");
                JPanel jp=new JPanel();
                JPanel jp1=new JPanel();
                //jp.add(jl);
                //jp.add(jl1);
                //jp.add(jt);
                //jp.add(jt1);
                jp.add(jb);
                jp.add(jb1);
                jp.add(jb2);
                jb.addActionListener(this);
                jb1.addActionListener(this);
                jb2.addActionListener(this);
                jp.setLayout(null);
                //jl.setBounds(40,60,100,25);
                        //jt.setBounds(150,60,100,25);
                        //jl1.setBounds(40,90,100,25);
                       //jt1.setBounds(150,90,100,25);
                          jb.setBounds(90,100,120,25);
                          jb1.setBounds(50,180,75,25);
```

```
                                                    catch(Exception ex){}
                                                    }
                                                if(ae.getSource()==jb2)
                                                    {
                                                    //System.exit(0);
                                                        jf.setVisible(false);
                                                    }
                                            }
                                            public static void main(String arg[])
                                            {
                                                    RSAKey rsa=new RSAKey();
                                            }
                                    }
```

RSAKeyDesgn.java

```
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;
import java.io.FileInputStream;
import java.io.*;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JTextField;
import javax.swing.*;
```

```
                jb2.setBounds(140,180,75,25);
                jf1.setContentPane(jp);
                jp1.setLayout(null);
                jf1.setBounds(30,30,180,180);
                jf1.setVisible(true);
                jf.setContentPane(jf1);
                jf.setSize(300,300);
                jf.setVisible(true);
                }

public void actionPerformed(ActionEvent ae)
                {
                if(ae.getSource()==jb)
                {
                RSAKeyDsgn rsad=new
                RSAKeyDsgn();
                }
                if(ae.getSource()==jb1)
                {
                try
                {
                new Timers();
                jf.setVisible(false);
                EnRSA en=new
                EnRSA();

en.WriteRSA(keyrsa1.d,keyrsa1.n);
                }
```

```
public class RSAKeyDsgn extends JFrame implements ActionListener
{
        private JLabel pval,qval,rsa;
        private JTextField pfield,qfield;
        private JButton ok,cancel;
        String pstr,qstr;
        JFrame frsa;

         RSAKeyDsgn()
         {

                frsa = new JFrame("RSA KeyGeneration");
                JPanel jp=new JPanel();
                jp.setLayout(null);
                        rsa = new JLabel("RSA KEYGENERATION");
                      pval = new JLabel("First Prime Number (P)");
                     qval = new JLabel("Second Prime Number (Q)");
                        pfield = new JTextField(20);
                        qfield = new JTextField(20);
                        ok = new JButton("OK");
                    cancel = new JButton("CANCEL");
                                jp.add(pval);
                                jp.add(qval);
                               jp.add(pfield);
                               jp.add(qfield);
                                jp.add(ok);
                               jp.add(cancel);

                            rsa.setBounds(140,20,140,20);
                           pval.setBounds(45,55,150,20);
                         pfield.setBounds(210,55,150,20);
                           qval.setBounds(45,90,150,20);
                         qfield.setBounds(210,90,150,20);
                           ok.setBounds(110,150,90,20);
                       cancel.setBounds(210,150,90,20);
                            frsa.setContentPane(jp);

                          ok.addActionListener(this);
                       cancel.addActionListener(this);

                           frsa.setSize(450,250);
                          frsa.setVisible(true);
                      frsa.addWindowListener( new WindowAdapter()
                      {
                      public void windowClosing(WindowEvent e)
                      {
                      frsa.setVisible(false);
                      }
                      }
                      );
                      }
```

```
        public void actionPerformed(ActionEvent
                                                arg0)
    {
      if(arg0.getSource()==ok)
      {
            pstr=pfield.getText();
            qstr=qfield.getText();
            try
            {
                  if(Integer.parseInt(pstr) >=100
|| Integer.parseInt(qstr) >=100)
                  {

      JOptionPane.showMessageDialog(null,"You
Have Entered into HIGH LEVEL TRUST ","LEVEL OF
TRUST",JOptionPane.INFORMATION_MESSAGE);
                        try{
                        FileOutputStream
fos=new FileOutputStream("trust");

ObjectOutputStream                    oos=new
ObjectOutputStream(fos);

      oos.writeObject("high");}

      catch(Exception e){}
                  }
                  else    if(Integer.parseInt(pstr)
>=25 || Integer.parseInt(qstr) >=25)
                  {

      JOptionPane.showMessageDialog(null,"You
Have Entered into MEDIUM LEVEL TRUST ","LEVEL OF
TRUST",JOptionPane.INFORMATION_MESSAGE);
                        try{
                        FileOutputStream
fos=new FileOutputStream("trust.txt");

ObjectOutputStream                    oos=new
ObjectOutputStream(fos);

      oos.writeObject("medium");}
                        catch(Exception e){}
                  }
                  else
                  {

      JOptionPane.showMessageDialog(null,"You
Have Entered into LOW LEVEL TRUST ","LEVEL OF
TRUST",JOptionPane.INFORMATION_MESSAGE);
                        try{
                        FileOutputStream
fos=new FileOutputStream("trust");

                  ObjectOutputStream oos=new
                  ObjectOutputStream(fos);

                  oos.writeObject("low");}

                  catch(Exception e){}
                  }
                  FileInputStream fis=new
                  FileInputStream("msg.txt");
                  byte str[]=new
                  byte[fis.available()];
                  fis.read(str,0,str.length);
```

```
String str1=new String(str);
System.out.println(str1);
keyrsa1 k1=new keyrsa1(str1);
k1.Key(pstr,qstr);
frsa.setVisible(false);
}
catch(Exception ew){}
}
}
}
```

EnRSA.java

```
import java.io.BufferedInputStream;
import java.io.FileInputStream;
import java.io.FileWriter;
import java.io.IOException;
import java.io.*;
import java.net.*;
import javax.swing.JOptionPane;

public class EnRSA
{
static String s,str="",ch="";
long e,n,c,f,data;
int k,i;
String estr,nstr;
public void GetRSA(long es, long ns,String s) throws IOException
{
System.out.println(s);
byte b[]=s.getBytes();
FileOutputStream fos=new FileOutputStream("data1.txt");
fos.write(b);
e=es;
n=ns;
FileInputStream file=new FileInputStream("data1.txt");
System.out.println("The value for e :"+e);
System.out.println("The value of n :"+n);

while((data=file.read())!=-1)
{
if(Math.max(data,n) == data)
{

JOptionPane.showMessageDialog(null,"To Encrypt Message is too Big, must (Message < n)","Error",1);
System.exit(0);
}

if ( e % 2 == 0)
{
```

```
 c = 1;
for ( i = 1; i <= e/2; i++)
{
  f = (data*data) % n;
  c = (f*c) % n;
 }
}
else
{

  c = data;
  for (  i = 1; i <= e/2; i++)
  {
          f = (data*data) % n;
                  c = (f*c) % n;
                          }

                          }
                  k=(int)c;
          str=Long.toString(c);
                  ch=ch+str+" ";

                          }
          System.out.println(ch);
                  file.close();
                          }

public void WriteRSA(long d,long n)
                  throws IOException
                                  {
                  try
                                  {
                  String s=d+"";
                  String s1=n+"";
                  FileWriter put = new
                  FileWriter("rsain.txt");
                  put.write(ch);
                  put.close();
                  System.out.println("Destination
                  Name:"+Sender.destination);
                  Send.send(Sender.destination);




                                  }
                  catch(Exception es){}
                  }

                                  }
```

DERSA.java

```
import java.io.BufferedReader;
import java.io.FileOutputStream;
```

```
import java.io.FileReader;
import java.io.IOException;
import java.io.*;


public class DERSA
{


        static String s,sn1="";
        int info[] = {48,49,50,51,52,53,54,55,56,57};
        long a[]  = {0,1,2,3,4,5,6,7,8,9};
        long d,n,g,f,value,temp=0;
        int k,l=0,data,i;
        char c[] = new char[1000];
        String dstr,nstr;


        public void ProcessDecrp(long ds,long ns,String sn) throws IOException
        {
         d=ds;
         n=ns;
         byte b[]=sn.getBytes();
         FileOutputStream fos=new FileOutputStream("rsain.txt");
         fos.write(b);
         FileInputStream f0=new FileInputStream("rsain.txt");
         FileOutputStream put = new FileOutputStream("Output.txt");
         while((data=f0.read()) != -1)
 {

System.out.println("Data:"+data);
        value=0;
        for(i=0;i<info.length;i++)
                {
                 if(data==info[i])
                temp=a[i];
                }

value=(value*10)+temp;
                        data=f0.read();

                        while(data!=32)
                                {

                for(i=0;i<info.length;i++)
                                {
                        if(data==info[i])
                                temp=a[i];
                                }

                value=(value*10)+temp;
                        data=f0.read();
                                }

                 if ( d % 2 == 0)
                                {
                                g = 1;
```

```
for ( i = 1; i <= d/2;
i++)
{
f = (value*value) % n;
g = (f*g) % n;

System.out.println("if:"+g);
}
}
else
{
g = value;
for (  i = 1; i <= d/2;
i++)
{
f = (value*value) % n;
g = (f*g) % n;

System.out.println("else:"+g);
}
}
char k1=(char)g;
sn1=sn1+k1;

put.write(sn1.getBytes());
}
Recv.ja.setText(sn1);
put.close();
}
```

```
}
```

Recv.java

```
import java.io.*;
import java.net.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

class Recv implements ActionListener
{
        JFrame jf;
        static JTextArea ja;
        JButton jb;
        JLabel jl;
        JScrollPane jsp;
        static String sr,sr1,sr2;
        static long d,n;
        Recv()
        {
                jf=new JFrame("Receiver");
                jl=new JLabel("Receive the
Message");
                ja=new JTextArea();
                jsp=new JScrollPane(ja);
                jb=new JButton("Receive");
                JPanel jp=new JPanel();
                jp.add(jb);
                jp.add(jsp);
```

```
        jp.add(jl);
        jb.addActionListener(this);
        jp.setLayout(null);
        jl.setBounds(40,60,150,25);
        jsp.setBounds(40,90,250,300);
        jb.setBounds(100,420,100,25);
        jf.setContentPane(jp);
        jf.setSize(350,500);
        jf.setVisible(true);
}

  public void actionPerformed(ActionEvent ae)
                                {
                    if((JButton)jb==ae.getSource())
                                {
                    RSAKeyDecry dn=new
                                RSAKeyDecry();
                                }
                                }
```

Req.java

```
import java.net.*;
import java.util.*;
import java.io.*;
import javax.swing.*;

public class Req
{

static Socket s1,s2,d;
static Vector localvect=new Vector(10);
static Vector localvecttemp=new Vector(10);
static Vector recvect=new Vector(10);
static int i=0;
static String file="";
static Vector vv=new Vector();
static String mm;
static String tte;
static Sender sen;
static String jdd;
static String dd;

public static void main(String a[]) throws
                                Exception
{

        String path="";
        Enumeration it;
        String sender="";
        String
localaddr=(InetAddress.getLocalHost()).getHostName(
);
        String destination="";
```

```
                Socket s1,s2,d;
                String sockaddr="";
                String next="";
                int vectsize;

   try
            {

                ServerSocket ss=new ServerSocket(8888);
                new tab();
       while(true)
                {

System.out.println("Connected ");
                        tab.setTable("Connected ");

                    System.out.println(localaddr+" is Waiting..... ");
                    tab.setTable(localaddr+" is Waiting..... ");
                        Socket s=ss.accept();
                        InputStream ins=s.getInputStream();
                        OutputStream ous=s.getOutputStream();
                        ObjectOutputStream oos=new ObjectOutputStream(ous);
                        ObjectInputStream ois=new ObjectInputStream(ins);
                        FileReader fr=new FileReader("nextnodes.txt");
                        BufferedReader br=new BufferedReader(fr);
                        String fd=(String)ois.readObject();
                        if(fd.equals("reqroute"))
                    {

            System.out.println("Route Request...");

            tab.setTable("Route Request...");

            localvect=(Vector)ois.readObject();

                sender=(String)localvect.firstElement();

             System.out.println("The Sender address is : "+sender);

             tab.setTable("The Sender address is : "+sender);

                destination=(String)localvect.get(1);

            System.out.println("The Destined address is : "+destination);

             tab.setTable("The Destined address is : "+destination);

            System.out.println("The Local Address is : "+localaddr);

             tab.setTable("The Local Address is : "+localaddr);

            if(destination.equalsIgnoreCase(localaddr))
                                                                {
```

```
req(localvect);
}
else
{
while((next=br.readLine())!=null)
{
try
{
s1=new Socket(next,8888);
OutputStream ous1=s1.getOutputStream();
ObjectOutputStream oos3=new ObjectOutputStream(ous1);
localvect.addElement(localaddr);
oos3.writeObject("reqroute");
oos3.writeObject(localvect);
}
catch(Exception e)
{
System.out.println("Error in file "+e);
tab.setTable("Error in file "+e);
}
}
}
}
else if(fd.equals("ack"))
{
Timers.rece();
System.out.println("Acknowledgement from destination ...");
tab.setTable1("Acknowledgement from destination ...");
localvect=(Vector)ois.readObject();
it=localvect.elements();
while(it.hasMoreElements())
{
path=(String)it.nextElement();
System.out.print(" "+path+" --> ");
```

```
System.out.println("The path from source to destination is as follows :");

tab.setTable1("The path from source to destination is as follows :");

while(it.hasMoreElements())
{
path=(String)it.nextElement();
System.out.println(" "+path+" --> ");
tab.setTable1(" "+path+" --> ");
}
new Store_Path(i,localvect);
Store_Path.file();
System.out.println("   Destination ");
tab.setTable1("    Destination   ");
Socket1();
}
else if(((String)localvect.get(2)).equalsIgnoreCase(localaddr))
{
        tab.setTable1(" "+path+" --> ");


        System.out.println();

if(((String)localvect.firstElement()).equalsIgnoreCase(localaddr))
                {
        i++;

        System.out.println("Acknowledgement sent from the destination ....");

        tab.setTable1("Acknowledgement sent from the destination ....");

        System.out.println("The sender can send the data now....");

        tab.setTable1("The sender can send the data now....");

        String destin=(String)localvect.remove(1);

        localvect.add(destin);

        it=localvect.elements();

        System.out.println("Source  ");

        tab.setTable1("Source ");
```

```
localvect.remove(localvect.size()-1);

s2=new Socket((String)localvect.firstElement(),8888);

OutputStream os2=s2.getOutputStream();

ObjectOutputStream ous2=new ObjectOutputStream(os2);

ous2.writeObject("ack");

FileOutputStream foo=new FileOutputStream("paths.txt");

ObjectOutputStream oopp=new ObjectOutputStream(foo);

oopp.writeObject(localvect);

ous2.writeObject(localvect);

}

else

{

localvect.remove(localvect.size()-1);

sockaddr=(String)localvect.get(localvect.size()-1);

Enumeration en=localvect.elements();

System.out.println("The next system address is "+sockaddr);

tab.setTable1("The next system address is "+sockaddr);

s2=new Socket(sockaddr,8888);

System.out.println("Intermediate node "+localaddr+" forwarding the ack to "+sockaddr);

tab.setTable1("Intermediate node "+localaddr+" forwarding the ack to "+sockaddr);

OutputStream os2=s2.getOutputStream();

ObjectOutputStream ous2=new ObjectOutputStream(os2);

ous2.writeObject("ack");

ous2.writeObject(localvect);

}

}

else if(fd.equals("file++"))

{
```

```
                                                                    Recv re=new
                                                                    Recv();


                                                                    FileInputStream pa=new
                                                                    FileInputStream("paths.txt");


                                                                    ObjectInputStream ooo=new
                                                                    ObjectInputStream(pa);

                                                                    Vector
                                                                    n=(Vector)ooo.readObject();

                                                                    for(int
                                                                    h=0;h<n.size();h++)

                                                                    {

                                                                    System.out.println("The value inside vector is
                                                                    "+n.get(h));


                                                                    tab.setTable("The value inside vector is
                                                                    "+n.get(h));

                                                                    }

                                                                    if(n.size()==2)

                                                                    {


                                                                    Socket ssp=new Socket((String)n.get(0),8888);


                                                                    ObjectOutputStream oop=new
                                                                    ObjectOutputStream(ssp.getOutputStream());


                                                                    oop.writeObject("ackfile");


                                                                    oop.writeObject("The message \""+dd+"\" has
                                                                    received by the destination "+(String)n.get(1));

                                                                    }

                                                                    else

                                                                    {
                                                i++;


                                                String
                                                sub=(String)ois.readObject();


                                                vv=(Vector)ois.readObject();


System.out.println("The String sent is "+sub);

                                                new
                                                Store_Path(i,vv);


                                                Store_Path.file();


                                                Sender.Socket2(i,sub);

                                                }

                                                else
                                                if(fd.equals("file"))

                                                {


JOptionPane.showMessageDialog(null,"File
Received","File",JOptionPane.INFORMATION_MESSA
GE);


                                                dd=(String)ois.readObject();


                                                System.out.println("The message sent from
                                                source is ::"+dd);


                                                tab.setTable1("The message sent from source
                                                is ::"+dd);
```

```
                                                    {
                                                    String ds=(String)ois.readObject();
                                                    Vector vl=(Vector)ois.readObject();
        Socket sss=new Socket((String)n.lastElement(),8888);                    if(vl.size()==2)
                                                    {
        n.removeElementAt(n.size()-1);
                                                    Socket sspd=new Socket((String)vl.get(0),8888);
        ObjectOutputStream oop1=new ObjectOutputStream(sss.getOutputStream());
                                                    ObjectOutputStream oop=new ObjectOutputStream(sspd.getOutputStream());
        oop1.writeObject("ackfile1");
                                                    oop.writeObject("ackfile");
        oop1.writeObject("The message \""+dd+"\" has received by the destinaton "+(String)n.get(1));
                                                    oop.writeObject(ds);
        oop1.writeObject(n);
                                                    }
                                                    else
                                                    {
                                    }
                                                    Socket ssj=new Socket((String)vl.lastElement(),8888);
                                    }
                                    else if(fd.equals("ackfile"))
                                                    vl.removeElementAt(vl.size()-1);
                                    {
                                                    ObjectOutputStream oop1=new ObjectOutputStream(ssj.getOutputStream());
                                    jdd=(String)ois.readObject();
                                                    oop1.writeObject("ackfile1");
                                    System.out.println(jdd);
                                                    oop1.writeObject(ds);
                                    tab.setTable1(jdd);
                                                    oop1.writeObject(vl);
                                    }
                                    else if(fd.equals("ackfile1"))
```

```
                }
            }
        }
    }
    catch(Exception e)
    {
        System.out.println("Some Error has occurred : "+e);
        tab.setTable("Some Error has occurred : "+e);
        }
    }

    public static void req(Vector v)
    {
        String b="";
        try
        {
            int o=v.size();
            if(o==2)
            {
b=(String)v.firstElement();
            }
            else
            {
b=(String)(v.lastElement());
            }
            Enumeration en=v.elements();
while(en.hasMoreElements())
            {
                String f=(String)en.nextElement();
System.out.println("The path from source to destination  : "+f);
                tab.setTable1("The path from source to destination  : "+f);
            }
            d=new Socket(b,8888);
            OutputStream os=d.getOutputStream();
            ObjectOutputStream oos1=new ObjectOutputStream(os);
            oos1.writeObject("ack");
            String ss="";
            localvect=v;

            FileOutputStream foo=new FileOutputStream("paths.txt");
            ObjectOutputStream oopp=new ObjectOutputStream(foo);
            oopp.writeObject(localvect);
            boolean boo=true;
            if(o==2)
```

```
{}
else
{
for(int i=2;i<o;i++)
{
v.add((String)localvect.get(i));
}
}
oos1.writeObject(v);
}
catch(Exception e)
{
System.out.println("Error at ack sending in destination "+e);
tab.setTable1("Error at ack sending in destination "+e);
}
}

public static void Socket1()throws Exception
{
System.out.println("Inside Socket1");
FileInputStream fip=new FileInputStream("rsain.txt");
byte str[]=new byte[fip.available()];
fip.read(str,0,str.length);
String tte=new String(str);
System.out.println("The mmm is :::"+tte);
FileInputStream fis=new FileInputStream("path1.txt");
ObjectInputStream ois=new ObjectInputStream(fis);
Vector d=(Vector)ois.readObject();
int siz=d.size();
System.out.println("The size of the vector is ::"+siz);
for(int j=0;j<siz;j++)
{
System.out.print("++__++-->>"+d.get(j));
}
if(siz==2)
{
Socket soc=new Socket((String)d.get(1),8888);
ObjectOutputStream oos=new ObjectOutputStream(soc.getOutputStream());
oos.writeObject("file");
System.out.println("The string in the textbox ::"+tte);
oos.writeObject(tte);
```

```
                }
                else
                {
                        d.removeElementAt(0);
                        int m=d.size();
                        System.out.println("The size after removing the element..."+m);
                        Socket soc=new Socket((String)d.get(1),8888);
                        ObjectOutputStream oos=new ObjectOutputStream(soc.getOutputStream());
                        oos.writeObject("file++");
                        System.out.println("The string in the textbox ::"+tte);
                        oos.writeObject(tte);
                        oos.writeObject(d);
                }
  }
}
```

CONCLUSION

The recent major trends of system modernization and transition to net enabled technologies present a growing need for more security in Data transfer. IT organizations are required to provide greater security for the transformation of messages. Hence we provide security using key Generation and Verification.The proposed system has fully satisfied the following task:Low time consumption, High ReliabilityFull control of source and target data definitions, Efficient utilization of effective utilizations, High operational speed Less manual effortFuture networks will offer seamless and ubiquitous services, so that the user may even not be aware of the existence of different radio access networks. A main issue we address is hand-over between different wireless access technologies. To achieve this goal, Active

RoutingProtocol(ASR) for ad-hoc network is proposed. It utilizes active network technology as a adaptive control path controller.

REFERENCES

1. W. Arbaugh, N. Shankar, and Y.C. Wan. Your 802.11 wireless network has no clothes. Technical report, Dept. of Computer Science, University of Maryland, March 2001.

2. E.M. Belding-Royer and C.-K. Toh. A review of current routing protocols for ad-hoc mobile wireless networks. *IEEE Personal Communicatio Magazine*, pages 46–55, April 1999.

3. Herbert Schildt, Edition (2003) '**The Complete Reference JAVA 2'** Tata McGraw Hill Publications .

4. Michael Foley and Mark McCulley, Edition(2002) **'JFC Unleashed'**Prentice-Hall India.

5. Andrew S Tanenbaum, Edition**(2003) ' Computer Networks** '

6. [6] A. Tsirigos and Z.J. Haas, ―Multipath Routing in the Presence of Frequent Topological Changes,‖ *IEEE Comm. Magazine*, pp. 132-138, Nov. 2001.

7. [7] J. Broch, D.A. Maltz, D.B. Johnson, Y-C. Hu, J. Jetcheva, ―A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols,‖ in proceedings of the *4th International Conference on Mobile Computing (Mobicom'98),* 1998.
8. [8] ―Secure on-demand distance-vector routing in ad hoc networks,‖ in *Proc. 2005 IEEE Sarnoff Symp.*, Princeton, NJ, Apr. 2005, pp. 168–171.