GALGOTIAS UNIVERSITY

(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

Privacy-Preserving of Cloud Storage Security

A Report for the ETE Evaluation of Project 2

Submitted by

Adarsh Kumar Pandey

(1613105006 / 16SCSE105081)

in partial fulfilment for the award of the degree of

Bachelor of Technology

IN

Computer Science and Engineering with Specialization in

Cloud Computing and Virtualization

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

Under the Supervision of

Mr. A Daniel

Assistant Professor

APRIL / MAY- 2020

SCHOOL OF COMPUTING AND SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

Certified that this project report **"Privacy-Preserving of Cloud Storage Security"** is the bonafide work of **"Adarsh Kumar Pandey (1613105006)"** who carried out the project work under my supervision.

SIGNATURE OF HEAD
Dr. MUNISH SHABARWAL,
PhD (Management), PhD (CS)
Professor & Dean,
School of Computing Science & Engineering

SIGNATURE OF SUPERVISOR
Mr. A Daniel
Assistant Professor
School of Computing Science & Engineering

TABLE OF CONTENTS

1. ABSTRACT

Cloud computing has been envisioned as the next-generation architecture of IT enterprise. In contrast to traditional solutions, where the IT services are under proper physical, logical and personnel controls, cloud computing moves the application software and databases to the large data centres, where the management of the data and services may not be fully trustworthy. This unique attribute, however, poses many new security challenges which have not been well understood. In this article, we focus on cloud data storage security, which has always been an important aspect of quality of service. To ensure the correctness of users' data in the cloud, we propose an effective and flexible distributed scheme with two salient features, opposing to its predecessors. By utilizing the homomorphic token with distributed verification of erasure-coded data, our scheme achieves the integration of storage correctness insurance and data error localization, i.e., the identification of misbehaving server (s). Unlike most prior works, the new scheme further supports secure and efficient dynamic operations on data blocks, including: data update, delete and append. Extensive security and performance analysis show that the proposed scheme is highly efficient and resilient against Byzantine failure, malicious data modification attack, and even server colluding attacks.

2. INTRODUCTION

Several trends are opening up the era of Cloud Computing, which is an Internet-based development and use of computer technology. The ever cheaper and more powerful processors, together with the software as a service (SaaS) computing architecture, are transforming data centres into pools of computing service on a huge scale. The increasing network bandwidth and reliable yet flexible network connections make it even possible that users can now subscribe high quality services from data and software that reside solely on remote data centres.

Moving data into the cloud offers great convenience to users since they don't have to care about the complexities of direct hardware management. The pioneer of Cloud Computing vendors, Amazon Simple Storage Service (S3) and Amazon Elastic Compute Cloud (EC2) are both well-known examples. While these internet-based online services do provide huge amounts of storage space and customizable computing resources, this computing platform shift, however, is eliminating the responsibility of local machines for data maintenance at the same time. As a result, users are at the mercy of their cloud service providers for the availability and integrity of their data. Recent downtime of Amazon's S3 is such an example.

From the perspective of data security, which has always been an important aspect of quality of service, Cloud Computing inevitably poses new challenging security threats for number of reasons. Firstly, traditional cryptographic primitives for the purpose of data security protection cannot be directly adopted due to the users' loss control of data under Cloud Computing. Therefore, verification of correct data storage in the cloud must be conducted without explicit knowledge of the whole data. Considering various kinds of data for each user stored in the cloud and the demand of long-term continuous assurance of their data safety, the problem of verifying correctness of data storage in the cloud becomes even more challenging. Secondly, Cloud Computing is not just a third-party data warehouse. The data stored in the cloud may be frequently updated by the users, including insertion, deletion, modification, appending, reordering, etc. To ensure storage correctness under dynamic data update is hence of paramount importance. However, this dynamic feature also makes traditional integrity insurance techniques futile and entails new solutions.

3. EXISTING SYSTEM

From the perspective of data security, which has always been an important aspect of quality of service, Cloud Computing inevitably poses new challenging security threats for number of reasons.

1. Firstly, traditional cryptographic primitives for the purpose of data security protection cannot be directly adopted due to the users' loss control of data under Cloud Computing. Therefore, verification of correct data storage in the cloud must be conducted without explicit knowledge of the whole data. Considering various kinds of data for each user stored in the cloud and the demand of long-term continuous assurance of their data safety, the problem of verifying correctness of data storage in the cloud becomes even more challenging.
2. Secondly, Cloud Computing is not just a third-party data warehouse. The data stored in the cloud may be frequently updated by the users, including insertion, deletion, modification, appending, reordering, etc. To ensure storage correctness under dynamic data update is hence of paramount importance.

These techniques, while can be useful to ensure the storage correctness without having users possessing data, cannot address all the security threats in cloud data storage, since they are all focusing on single server scenario and most of them do not consider dynamic data operations. As a complementary approach, researchers have also proposed distributed protocols for ensuring storage correctness across multiple servers or peers. Again, none of these distributed schemes is aware of dynamic data operations. As a result, their applicability in cloud data storage can be drastically limited.

4. PROPOSED SYSTEM

In this project, we propose an effective and flexible distributed scheme with explicit dynamic data support to ensure the correctness of users' data in the cloud. We rely on erasure correcting code in the file distribution preparation to provide redundancies and guarantee the data dependability. This construction drastically reduces the communication and storage overhead as compared to the traditional replication-based file distribution techniques. By utilizing the homomorphic token with distributed verification of erasure-coded data, our scheme achieves the storage correctness insurance as well as data error localization: whenever data corruption has been detected during the storage correctness verification, our scheme can almost guarantee the simultaneous localization of data errors, i.e., the identification of the misbehaving server(s).

1. Compared to many of its predecessors, which only provide binary results about the storage state across the distributed servers, the challenge-response protocol in our work further provides the localization of data error.
2. Unlike most prior works for ensuring remote data integrity, the new scheme supports secure and efficient dynamic operations on data blocks, including: update, delete and append.
3. Extensive security and performance analysis shows that the proposed scheme is highly efficient and resilient against Byzantine failure, malicious data modification attack, and even server colluding attacks.

5. IMPLEMENTATION

Implementation is the stage of the project when the theoretical design is turned out into a working system. Thus, it can be considered to be the most critical stage in achieving a successful new system and in giving the user, confidence that the new system will work and be effective. The implementation stage involves careful planning, investigation of the existing system and its constraints on implementation, designing of methods to achieve changeover and evaluation of changeover methods.

Main Modules:

1. Client Module:

In this module, the client sends the query to the server. Based on the query the server sends the corresponding file to the client. Before this process, the client authorization step is involved. In the server side, it checks the client name and its password for security process. If it is satisfied and then received the queries form the client and search the corresponding files in the database. Finally, find that file and send to the client. If the server finds the intruder means, it set the alternative Path to those intruders.

2. System Module:

Representative network architecture for cloud data storage is illustrated in Figure 1. Three different network entities can be identified as follows:

• User:

Users, who have data to be stored in the cloud and rely on the cloud for data computation, consist of both individual consumers and organizations.

• Cloud Service Provider (CSP):

A CSP, who has significant resources and expertise in building and managing distributed cloud storage servers, owns and operates live Cloud Computing systems,

• Third Party Auditor (TPA):

An optional TPA, who has expertise and capabilities that users may not have, is

Trusted to assess and expose risk of cloud storage services on behalf of the users upon request.

3. Cloud data storage Module:

Cloud data storage, a user stores his data through a CSP into a set of cloud servers, which are running in a simultaneous, the user interacts with the cloud servers via CSP to access or retrieve his data. In some cases, the user may need to perform block level operations on his data. users should be equipped with security means so that they can make continuous correctness assurance of their stored data even without the existence of local copies. In case that users do not necessarily have the

time, feasibility or resources to monitor their data, they can delegate the tasks to an optional trusted TPA of their respective choices. In our model, we assume that the point-to-point communication channels between each cloud server and the user is authenticated and reliable, which can be achieved in practice with little overhead.

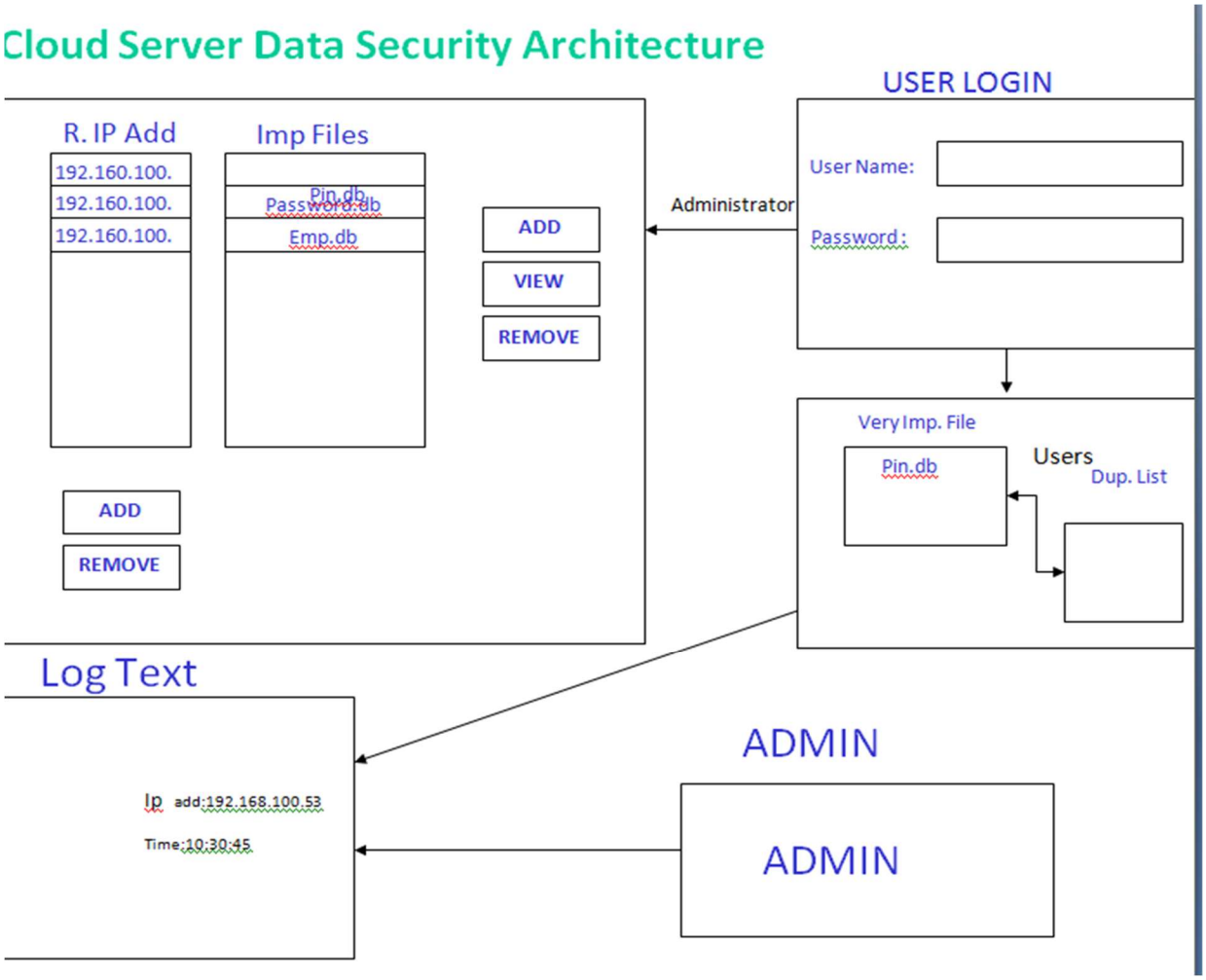
4. Cloud Authentication Server:

The Authentication Server (AS) functions as any AS would with a few additional behaviours added to the typical client-authentication protocol. The first addition is the sending of the client authentication information to the masquerading router. The AS in this model also functions as a ticketing authority, controlling permissions on the application network. The other optional function that should be supported by the AS is the updating of client lists, causing a reduction in authentication time or even the removal of the client as a valid client depending upon the request

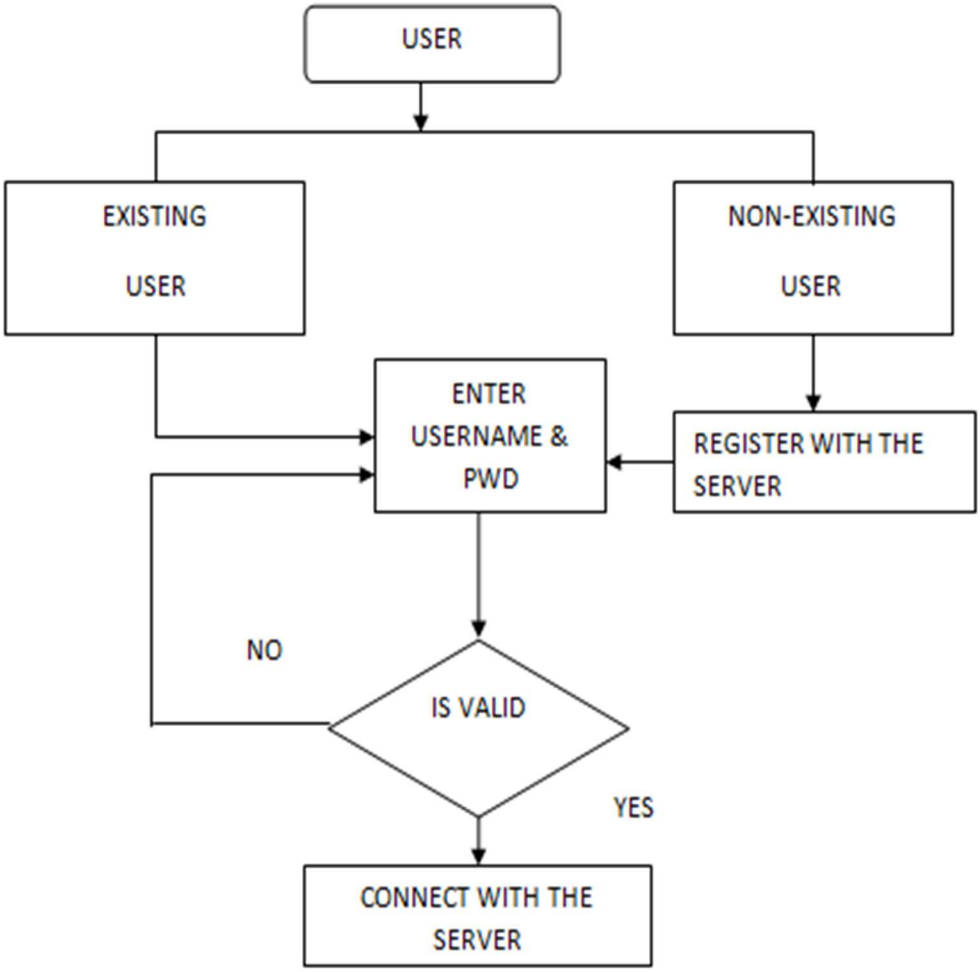
5. Unauthorized data modification and corruption module:

One of the key issues is to effectively detect any unauthorized data modification and corruption, possibly due to server compromise and/or random Byzantine failures. Besides, in the distributed case when such inconsistencies are successfully detected, to find which server the data error lies in is also of great significance.

6.LIST OF FIGURES



Client-


USER
EXISTING USER
NON-EXISTING USER
ENTER USERNAME & PWD
REGISTER WITH THE SERVER
NO
IS VALID
YES
CONNECT WITH THE SERVER

Use Case Diagram

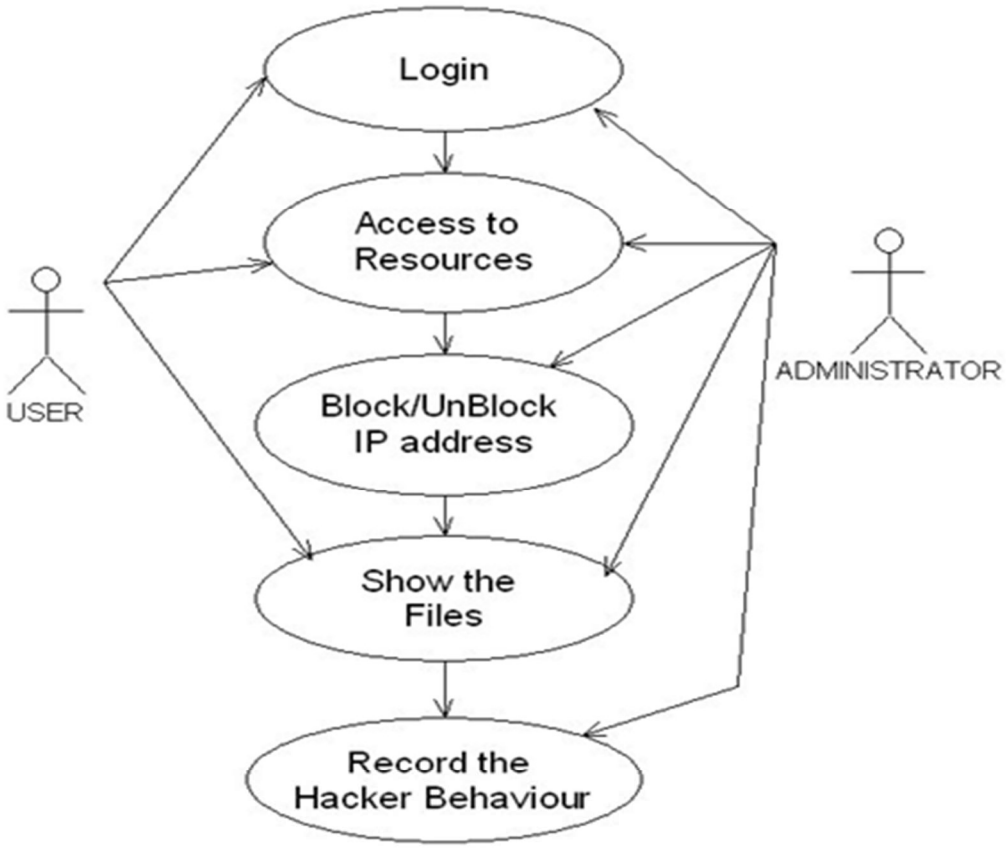
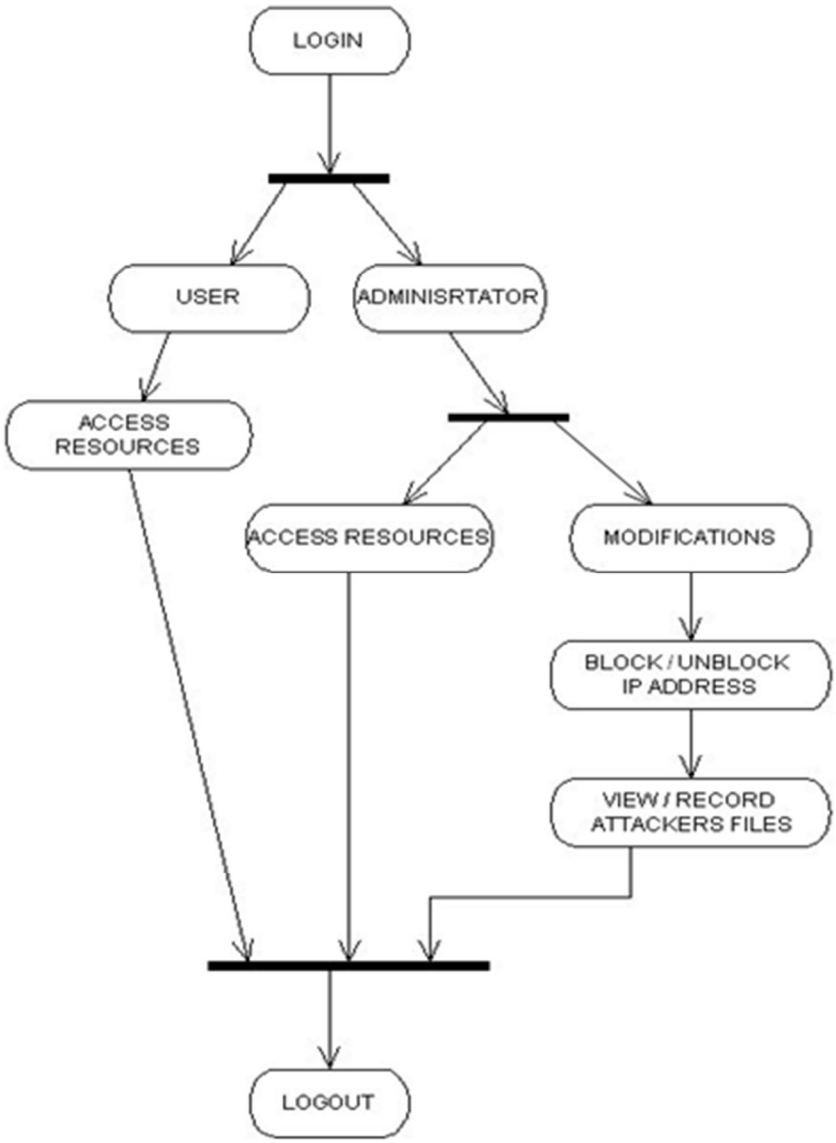
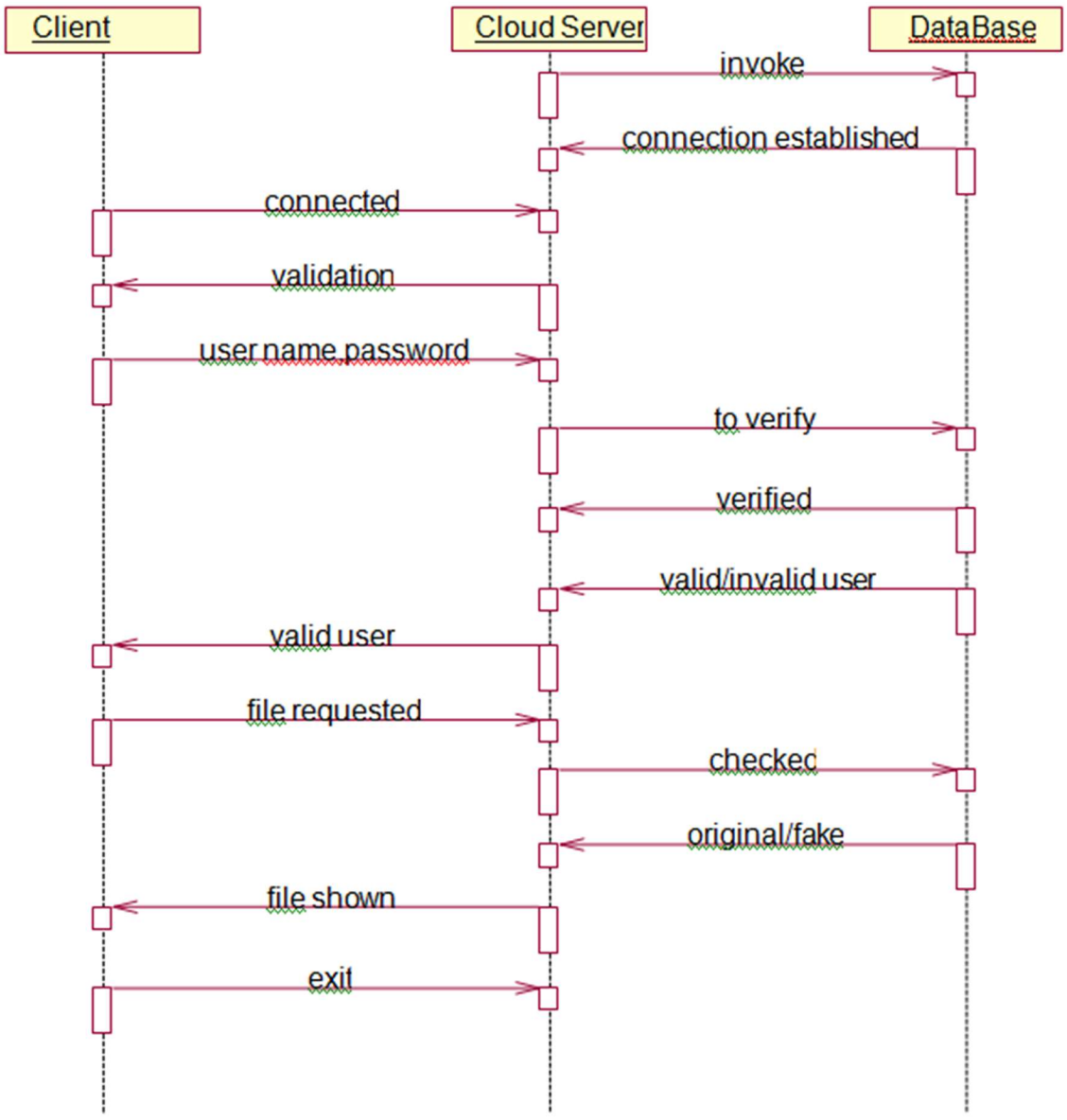


Fig Use case Diagram

Activity Diagram



Sequence Diagram


Client
Cloud Server
DataBase
invoke
connection established
connected
validation
user name,password
to verify
verified
valid/invalid user
valid user
file requested
checked
original/fake
file shown
exit

CLASS DIAGRAM

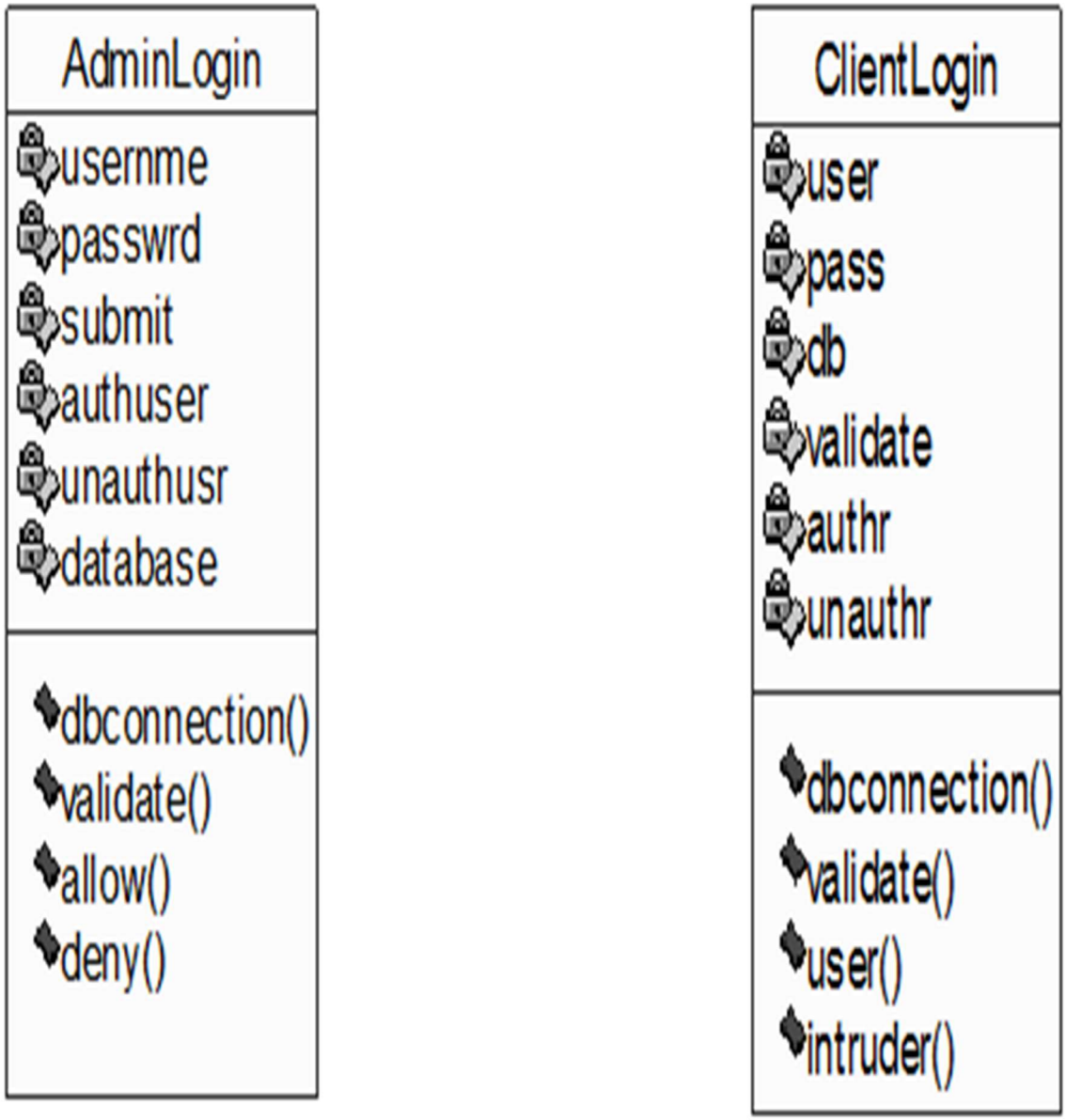
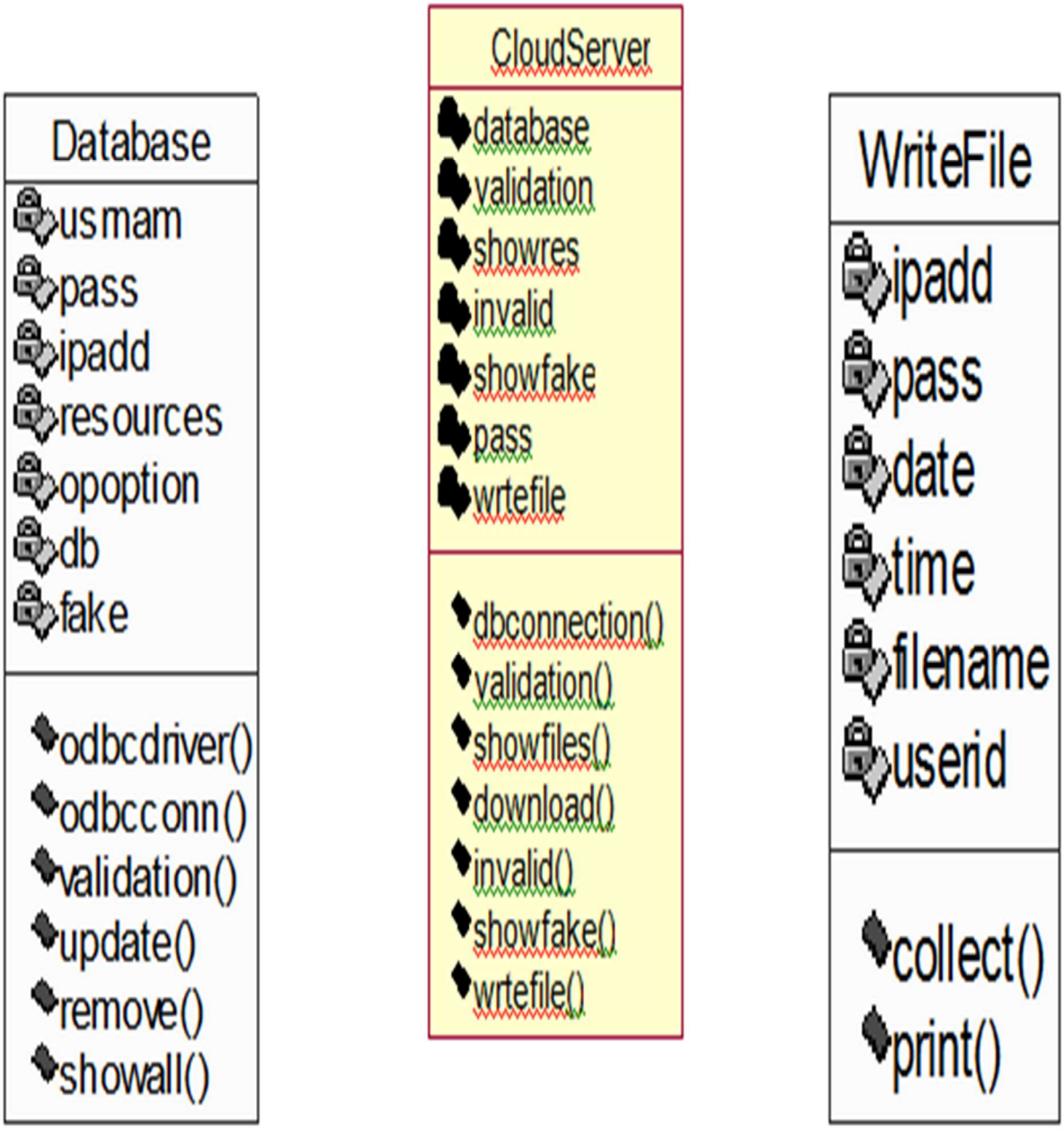


Fig Class Diagrams



7.SOURCE CODE

```
import java.awt.*;
import java.awt.event.*;
import java.io.PrintStream;
import java.rmi.Naming;
import java.util.StringTokenizer;
import java.util.Vector;
import javax.swing.*;
import javax.swing.border.*;

public class Server extends JFrame
{

   private JPanel contentpane;
   private JPanel RemoveIP;
   private JPanel jPanel2;
         private JButton AddIP;
   private JButton AddRes;
   private JButton RemoveRes;
   private JButton StartSer;
   private JButton Startview;
   private JButton Startmobilealert;
   private JButton Adduser;
   private JButton removeIP;
   private JButton addIP;
```

```
    private JButton addRes;
    private JLabel jLabel1;
    private JLabel jLabel2;
    private JLabel headlabl;
          private JLabel headlab2;
    private JList jList1;
    private JList jList2;

    private JOptionPane op;
    private static JFrame frame = null;

    public Server()
    {
   op=new JOptionPane();
        contentpane = (JPanel) this.getContentPane();
        StartSer = new JButton();
        Startview = new JButton();
       Startmobilealert=new JButton();
        Adduser=new JButton();
        RemoveIP = new JPanel();
        jLabel1 = new JLabel();
        jList1 = new JList();
        AddIP = new JButton();
        removeIP = new JButton();
///
        addIP = new JButton();
        jPanel2 = new JPanel();
```

```
        jLabel2 = new JLabel();
        headlabl=new JLabel();
                    headlab2=new JLabel();
         jList2 = new JList();
        AddRes = new JButton();
        RemoveRes = new JButton();
        addRes = new JButton();
//////////
        contentpane.setLayout(null);
        this.setSize(new Dimension(900, 620));
//        this.setResizable(false);
        this.setTitle("Ensuring Data Storage Security in Cloud Computing");
          headlabl.setText("Ensuring Data Storage Security in Cloud Computing");
          headlab2.setText("Cloud Authentication Server ");

          headlabl.setForeground(new Color(255, 0, 10 ));
          //headlabl.setBackground(new Color(255, 255, 255));
                    headlabl.setFont(new Font("Arial", 0, 20));
          headlabl.setBounds(200, 0, 800, 30);
          contentpane.add(headlabl,null);
        //setBackground(new Color(254, 253, 184));

                    headlab2.setForeground(new Color(255, 100, 0));
          //headlabl.setBackground(new Color(255, 255, 255));
                    headlab2.setFont(new Font("Arial", 0, 20));
          headlab2.setBounds(300, 15, 500, 70);
          contentpane.add(headlab2,null);
```

```
        addWindowListener(new WindowAdapter() {

            public void windowClosing(WindowEvent windowevent)
            {
                exitForm(windowevent);
            }

        });

        StartSer.setBackground(new Color(36, 72, 72));
        StartSer.setFont(new Font("Arial", 0, 14));
        StartSer.setForeground(new Color(255, 255, 255));
        StartSer.setText("Start Server");
        StartSer.setBorder(new BevelBorder(0));
        StartSer.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent actionevent)
            {
                StartSerActionPerformed(actionevent);
            }

        });
        contentpane.add(StartSer,null);
        StartSer.setBounds(30, 65, 140, 23);
///
        Startview.setBackground(new Color(36, 72, 72));
```

```
        Startview.setFont(new Font("Arial", 0, 14));
        Startview.setForeground(new Color(255, 255, 255));
        Startview.setText("Hackers Info");
        Startview.setBorder(new BevelBorder(0));
        Startview.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent actionevent)
            {
                StartviewActionPerformed(actionevent);
            }

        });
        contentpane.add(Startview,null);
        Startview.setBounds(210, 65, 140, 23);

//////////
        Startmobilealert.setBackground(new Color(36, 72, 72));
        Startmobilealert.setFont(new Font("Arial", 0, 14));
        Startmobilealert.setForeground(new Color(255, 255, 255));
        Startmobilealert.setText("Enable Mobile Alert");
        Startmobilealert.setBorder(new BevelBorder(0));
        Startmobilealert.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent actionevent)
            {
                StartmobilealertActionPerformed(actionevent);
            }
```

```
        });
        contentpane.add(Startmobilealert,null);
        Startmobilealert.setBounds(380, 65, 140, 23);

///
//////////
        Adduser.setBackground(new Color(36, 72, 72));
        Adduser.setFont(new Font("Arial", 0, 14));
        Adduser.setForeground(new Color(255, 255, 255));
        Adduser.setText("Add Newuser");
        Adduser.setBorder(new BevelBorder(0));
        Adduser.addActionListener(new ActionListener() {

           public void actionPerformed(ActionEvent actionevent)
           {
              AdduserActionPerformed(actionevent);
           }

        });
        contentpane.add(Adduser,null);
        Adduser.setBounds(560, 65, 140, 23);

///
        RemoveIP.setLayout(null);
        RemoveIP.setBackground(new Color(0, 116, 232));
        RemoveIP.setBorder(new EtchedBorder());
```

```
        jLabel1.setFont(new Font("Arial", 0, 17));
                jLabel1.setForeground(new Color(255, 255, 255));
        jLabel1.setText("Restricted IP Addresses:");
        RemoveIP.add(jLabel1);
        jLabel1.setBounds(20, 10, 200, 40);
        jList1.setBorder(new BevelBorder(2));
        RemoveIP.add(jList1);
        jList1.setBounds(10, 70, 220, 450);
        AddIP.setBackground(new Color(205, 205, 205));
        AddIP.setFont(new Font("Arial", 0, 14));
        AddIP.setForeground(new Color(0, 0, 0));
        AddIP.setText("View");
        AddIP.setBorder(new SoftBevelBorder(0));
        AddIP.addActionListener(new ActionListener() {

           public void actionPerformed(ActionEvent actionevent)
           {
              AddIPActionPerformed(actionevent);
           }

        });
        RemoveIP.add(AddIP);
        AddIP.setBounds(250, 70, 110, 25);
    //new ip
        removeIP.setBackground(new Color(205, 205, 205));
        removeIP.setFont(new Font("Arial", 0, 15));
        removeIP.setForeground(new Color(0, 0, 0));
```

```
        removeIP.setText("Remove IP");
        removeIP.setBorder(new SoftBevelBorder(0));
        removeIP.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent actionevent)
            {
                removeIPActionPerformed(actionevent);
            }

        });
        RemoveIP.add(removeIP);
        removeIP.setBounds(250, 110, 110, 25);

////////////new addip
        addIP.setBackground(new Color(205, 205, 205));
        addIP.setFont(new Font("Arial", 0, 15));
        addIP.setForeground(new Color(0, 0, 0));
        addIP.setText("Add New IP");
        addIP.setBorder(new SoftBevelBorder(0));
        addIP.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent actionevent)
            {
                addIPActionPerformed(actionevent);
            }

        });
```

```
        RemoveIP.add(addIP);
        addIP.setBounds(250, 150, 110, 25);

//////////////
        contentpane.add(RemoveIP,null);
        RemoveIP.setBounds(30, 90, 400, 550);
        jPanel2.setLayout(null);
        jPanel2.setBackground(new Color(0, 116, 232));
        jPanel2.setBorder(new EtchedBorder());
        jLabel2.setFont(new Font("Arial", 0, 17));
                jLabel2.setForeground(new Color(255, 255, 255));
        jLabel2.setText("Available Resources:");
        jPanel2.add(jLabel2);
        jLabel2.setBounds(10, 10, 200, 40);
        jList2.setBorder(new BevelBorder(2));
        jPanel2.add(jList2);
        jList2.setBounds(10, 70, 220, 450);
        AddRes.setBackground(new Color(205, 205, 205));
        AddRes.setFont(new Font("Arial", 0, 15));
        AddRes.setForeground(new Color(0, 0, 0));
        AddRes.setText("View");
        AddRes.setBorder(new SoftBevelBorder(0));
        AddRes.addActionListener(new ActionListener() {

           public void actionPerformed(ActionEvent actionevent)
           {
              AddResActionPerformed(actionevent);
```

```
        }

    });
    jPanel2.add(AddRes);
    AddRes.setBounds(250, 70, 110, 25);
    RemoveRes.setBackground(new Color(205, 205, 205));
    RemoveRes.setFont(new Font("Arial", 0, 15));
    RemoveRes.setForeground(new Color(0, 0, 0));
    RemoveRes.setText("Remove ");
    RemoveRes.setBorder(new SoftBevelBorder(0));
    RemoveRes.addActionListener(new ActionListener() {

        public void actionPerformed(ActionEvent actionevent)
        {
            RemoveResActionPerformed(actionevent);
        }

    });
    jPanel2.add(RemoveRes);
    RemoveRes.setBounds(250, 110, 110, 25);
//////////////new Res
      addRes.setBackground(new Color(205, 205, 205));
    addRes.setFont(new Font("Arial", 0, 15));
    addRes.setForeground(new Color(0, 0, 0));
    addRes.setText("Add New File");
    addRes.setBorder(new SoftBevelBorder(0));
    addRes.addActionListener(new ActionListener() {
```

```
            public void actionPerformed(ActionEvent actionevent)
            {
                addResActionPerformed(actionevent);
            }

        });
        jPanel2.add(addRes);
        addRes.setBounds(250, 150, 110, 25);
////////////////
        contentpane.add(jPanel2,null);
        jPanel2.setBounds(470, 90, 400, 550);
        pack();
    }

    private void StartSerActionPerformed(ActionEvent actionevent)
    {
        try
        {
            RMISImpl rmisimpl = new RMISImpl();
            Naming.rebind("RMIServer", rmisimpl);
            System.out.println("Server Is Runnig...");
        }
        catch(Exception exception)
        {
            System.out.println("Exception :" + exception);
        }
```

```
    }

//
    private void StartmobilealertActionPerformed(ActionEvent actionevent)
    {
        try
        {
emserver ob=new emserver();
ob.start();
       System.out.println("Mobile Alert Service Running..... ");
        }
        catch(Exception exception)
        {
            System.out.println("Exception :" + exception);
        }
    }
//
//
private void AdduserActionPerformed(ActionEvent actionevent)
    {
        try
        {

          new NewUser().show();
           }
        catch(Exception exception)
        {
```

```
            System.out.println("Exception :" + exception);
        }
    }

//

//
private void StartviewActionPerformed(ActionEvent actionevent)
    {
      new view();
    }
//
    private void RemoveResActionPerformed(ActionEvent actionevent)
    {    dbcon dbcon1 = new dbcon();
         String sss=(String) jList2.getSelectedValue();
         System.out.println("selected:"+sss);
         dbcon1.removede(sss,"RESOURCES","files");
         rss();
    }
    private void addResActionPerformed(ActionEvent actionevent)
    {
         String Fn=(String)JOptionPane.showInputDialog(this,"Enter The new
FileName:");
         String fFn=(String)JOptionPane.showInputDialog(this,"Enter The fake
FileName:");
         String op1=(String)JOptionPane.showInputDialog(this,"Enter The option of
FileSecurity:","false");
```

```
        String ip=(String)JOptionPane.showInputDialog(this,"Enter The IP
Address:");

if(Fn==null || fFn==null || op1==null ||Fn.equals("")||fFn.equals("")||op1.equals(""))
        {
        op.showConfirmDialog(this,"Enter proper Data
Only","Alert",JOptionPane.DEFAULT_OPTION,JOptionPane.ERROR_MESSAG
E);
        }
else
        {
        add_rss(Fn,fFn,op1,ip);
        }

   }

private void addIPActionPerformed(ActionEvent actionevent)
   {
String a=(String)JOptionPane.showInputDialog(this,"Enter The new IPAddress:");
new_addipr(a);
System.out.println("addip called"+a);
   }
private void rss()
{
dbcon dbcon1 = new dbcon();
       String s = dbcon1.listfile();
       Vector vector = new Vector(2);
```

```
        String s1;
        for(StringTokenizer stringtokenizer = new StringTokenizer(s, ";");
stringtokenizer.hasMoreTokens(); vector.addElement(s1))
        {
            s1 = stringtokenizer.nextToken();
        }

        jList2.setListData(vector);

}
private void add_rss(String f,String ff,String op,String allowedip)
{
        dbcon dbcon1 = new dbcon();
       dbcon1.add_res(f,ff,op,allowedip);
        String s = dbcon1.listfile();
        Vector vector = new Vector(2);
        String s1;
        for(StringTokenizer stringtokenizer = new StringTokenizer(s, ";");
stringtokenizer.hasMoreTokens(); vector.addElement(s1))
        {
            s1 = stringtokenizer.nextToken();
        }

        jList2.setListData(vector);

}
```

```
    private void AddResActionPerformed(ActionEvent actionevent)
    {

rss();
    }
private void ipr()
{
        dbcon dbcon1 = new dbcon();
         String s = dbcon1.listip();
          Vector vector = new Vector(2);
          String s1;
          for(StringTokenizer stringtokenizer = new StringTokenizer(s, ";");
stringtokenizer.hasMoreTokens(); vector.addElement(s1))
                {
                        s1 = stringtokenizer.nextToken();
                }

      jList1.setListData(vector);



}

private void new_addipr(String ip_n)
{
        dbcon dbcon1 = new dbcon();
        dbcon1.add_ip(ip_n);
```

```
        String s = dbcon1.listip();
        Vector vector = new Vector(2);
        String s1;
        for(StringTokenizer stringtokenizer = new StringTokenizer(s, ";");
stringtokenizer.hasMoreTokens(); vector.addElement(s1))
                {
                        s1 = stringtokenizer.nextToken();
                }

      jList1.setListData(vector);


}



   private void removeIPActionPerformed(ActionEvent actionevent)
   {
        String a=(String)JOptionPane.showInputDialog(this,"Enter The Key
Word");
        if(a.equals("admin"))
        {
        dbcon dbcon1 = new dbcon();
        String sss=(String) jList1.getSelectedValue();
        System.out.println("selected:"+sss);
        dbcon1.removede(sss,"ADDRESS","IPADDRESS");
        ipr();
```

```
        }
        else
        {
        op.showConfirmDialog(this,"Check The Key or Select
Ip","Alert",JOptionPane.DEFAULT_OPTION,JOptionPane.ERROR_MESSAGE);
        }

    }

    private void AddIPActionPerformed(ActionEvent actionevent)
    {
        ipr();
    }



    private void exitForm(WindowEvent windowevent)
    {
       System.exit(0);
    }
static void showMiddle(){
    Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();
    Dimension frameSize = frame.getSize();
    if (frameSize.height > screenSize.height) {
      frameSize.height = screenSize.height;
    }
    if (frameSize.width > screenSize.width) {
      frameSize.width = screenSize.width;
```

```
    }
    frame.setLocation((screenSize.width - frameSize.width) / 2, (screenSize.height -
frameSize.height) / 2);
    frame.setVisible(true);

}

    public static void main(String args[])
    {
        Server frame = new Server();
        ImageIcon image1 = new
ImageIcon(LoginDialog.class.getResource("images/network.JPG"));
        LoginDialog dia = new LoginDialog();
        JSplashScreen sp = new JSplashScreen(image1);
        sp.displayForm();
        dia.setSize(780,580);
        dia.showDialog();
        if(dia.getConfirm()){
                sp.closeSplashScreen();
                //frame.setVisible(true);
                frame.setSize(900,700);
                ////
                frame.show();//Middle();

        }
        else {
                sp.closeSplashScreen();
```

```
                System.exit(1);
        }

    }
}
```

DBCON

```
import java.sql.*;

public class dbcon
    {

                Connection con;
                    Statement st;
                    ResultSet rs;
dbcon()
{
        try{    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

                                con=DriverManager.getConnection("jdbc:odbc:db5");
}catch(ClassNotFoundException e){System.out.println(e);} catch (SQLException
e) {
   System.out.println(e);
                    }
```

```
}
        public String check(String u,String p)
        { String s=null;


                try{

                 st=con.createStatement();
                 rs=st.executeQuery("select * from login where user='"+u+"' and
pass='"+p+"' ");

                 if(rs.next())

                        {
                        s="ok";
                        }
                        else
                        {
                        s="notok";
                        }
                }catch(SQLException e){System.out.println(e);}

        return s;
        }

        public String check_key(String u,String key)
        { String s=null;
```

```
                try{

                 st=con.createStatement();
                 rs=st.executeQuery("select * from login where user='"+u+"' and
s_code='"+key+"' ");

                 if(rs.next())

                        {
                        s="ok";
                        }
                        else
                        {
                        s="notok";
                        }
                }catch(SQLException e){System.out.println(e);}

        return s;
        }

public String checkip(String ip)
        { String s=null;


                try{
```

```
                st=con.createStatement();
                rs=st.executeQuery("select * from ADDRESS where
ipaddress='"+ip+"' ");

                if(rs.next())

                        {
                        s="ok";
                        }
                        else
                        {
                        s="notok";
                        }
                }catch(SQLException e){System.out.println(e);}

        return s;
        }

public String checkfileoption(String option1)
        { String s=null;


                try{
```

```
                st=con.createStatement();
                rs=st.executeQuery("select * from RESOURCES where
files='"+option1+"' ");

                if(rs.next())

                        {
                        s=rs.getString(3);
                        }
                }catch(SQLException e){System.out.println(e);}

        return s;
        }

public String listfile(String clientIp)
        { String s="";


                try{

//select distinct username from userinfo
                st=con.createStatement();
                rs=st.executeQuery("select distinct files from RESOURCES where
allowedip='"+clientIp+"' ");

                while(rs.next())
```

```
                {
                s=s+rs.getString(1)+";";
                }

            }catch(SQLException e){System.out.println(e);}

        return s;
        }
public String listfile()
        { String s="";


            try{


             st=con.createStatement();
             rs=st.executeQuery("SELECT distinct files FROM  RESOURCES");

             while(rs.next())

                {
                s=s+rs.getString(1)+";";
                }

            }catch(SQLException e){System.out.println(e);}

        return s;
```

```
        }
public String listip()
        { String s="";


                try{


                 st=con.createStatement();
                 rs=st.executeQuery("select * from ADDRESS ");

                 while(rs.next())

                        {
                        s=s+rs.getString(1)+";";
                        }

                }catch(SQLException e){System.out.println(e);}

        return s;
        }

public String getfake(String fil)
        { String s=null;


                try{
```

```
                st=con.createStatement();
                rs=st.executeQuery("select * from RESOURCES where
files='"+fil+"'");

                if(rs.next())

                        {
                        s=rs.getString(2);
                        }

                }catch(SQLException e){System.out.println(e);}
                System.out.println(s);
        return s;
        }

public void removede(String data,String table1,String wh)
        {


                try{


                 st=con.createStatement();

String sql="delete  from "+table1+" where "+wh+"='"+data+"'";
```

```
System.out.println(sql);

                st.executeUpdate(sql);
                st.executeUpdate(sql);
                }catch(SQLException e){System.out.println(e);}
                System.out.println("deleted");

        }

public void add_ip(String s)
        {

                try{


                st=con.createStatement();

String sql="insert into ADDRESS values('"+s+"')";
System.out.println(sql);

                st.executeUpdate(sql);
                //st.executeUpdate(sql);
                }catch(SQLException e){System.out.println(e);}
                System.out.println("added");

        }
```

```
public void add_res(String f,String ff,String op,String allowedip)
        {

                try{
                 st=con.createStatement();

String sql="insert into RESOURCES
values('"+f+"','"+ff+"','"+op+"','"+allowedip+"')";
System.out.println(sql);

                 st.executeUpdate(sql);
                //st.executeUpdate(sql);
                }catch(SQLException e){System.out.println(e);}
                System.out.println("added");

        }

public void add_user(String user,String pass,String code,int age,String
gender,String phone,String address,String mobile)
        {

                try{


                 st=con.createStatement();
```

```
String sql="insert into LOGIN
values('"+user+"','"+pass+"','"+code+"',"+age+",'"+gender+"','"+phone+"','"+addre
ss+"','"+mobile+"')";
System.out.println(sql);

                st.executeUpdate(sql);
               //st.executeUpdate(sql);
               }catch(SQLException e){System.out.println(e);}
               System.out.println("added");

        }




public static void main(String arg[])
{
dbcon ob=new dbcon();
/*System.out.println(ob.check("a","a"));
System.out.println(ob.checkip("127.0.0.1"));
System.out.println(ob.listfile());
System.out.println(ob.getfake("Login.java"));
ob.removede("dsds","ADDRESS","IPADDRESS");
*/
//ob.add_ip("aasd");
//ob.add_res("aasd","sss","ddd");
//ob.add_user("a","a","a","w","s","d");
```

```
System.out.println(ob.checkfileoption("Logisn.java"));
}


 }
```

Login.java

```
import java.io.*;
import javax.swing.*;
import java.awt.*;
import java.util.StringTokenizer;
import java.net.InetAddress;
import java.rmi.Naming;
import java.util.Date;
import java.util.Vector;
public class Login extends javax.swing.JFrame
{
//String honey="127.0.0.1";
            RMISIntf ref;
           JOptionPane op;
            Vector v= new Vector(2);
            String sss=null;
                 static String username;

   /** Creates new form Login */

            public Login()
          {
```

```
        initComponents();
    }

    private void initComponents()
            {
            //GEN-BEGIN:initComponents
            jPan = new javax.swing.JPanel();
            jLabel1 = new javax.swing.JLabel();
            user = new javax.swing.JTextField();
            jLabel2 = new javax.swing.JLabel();
            submit = new javax.swing.JButton();
            reset = new javax.swing.JButton();
            exit = new javax.swing.JButton();
            pass = new javax.swing.JPasswordField();
            jLabel3 = new javax.swing.JLabel();
            op=new JOptionPane();
            getContentPane().setLayout(null);

            jPanel1 = new javax.swing.JPanel();
            listres = new javax.swing.JList();
            jLab = new javax.swing.JLabel();
            download = new javax.swing.JButton();
            resArea = new TextArea();
            save = new javax.swing.JButton();

            getContentPane().setLayout(null);
```

```
addWindowListener(new java.awt.event.WindowAdapter()
  {
        @Override
           public void windowClosing(java.awt.event.WindowEvent evt)
          {

          exitForm(evt);

                    }
    });

jPan.setLayout(null);

jPan.setBackground(new java.awt.Color(0, 116, 232));
jPan.setBorder(new javax.swing.border.EtchedBorder());
jLab.setFont(new java.awt.Font("Arial", 0, 14));
          jLab.setForeground(new java.awt.Color(255, 255, 255));
jLab.setText("User ID:");
jPan.add(jLab);
jLab.setBounds(40, 40, 90, 30);

jPan.add(user);
user.setBounds(140, 40, 100, 30);

jLabel2.setFont(new java.awt.Font("Arial", 0, 14));
          jLabel2.setForeground(new java.awt.Color(255, 255, 255));
jLabel2.setText("Password:");
```

```
jPan.add(jLabel2);
jLabel2.setBounds(40, 90, 90, 30);

submit.setBackground(new java.awt.Color(255, 255, 255));
submit.setFont(new java.awt.Font("Arial", 0, 14));
submit.setForeground(new java.awt.Color(0, 0, 0));
submit.setText("Submit");

submit.addActionListener(new java.awt.event.ActionListener()
  {
        public void actionPerformed(java.awt.event.ActionEvent evt)
           {
                    submitActionPerformed(evt);
                }
  });

jPan.add(submit);
submit.setBounds(20, 150, 80, 27);

reset.setBackground(new java.awt.Color(255, 255, 255));
reset.setFont(new java.awt.Font("Arial", 0, 14));
reset.setForeground(new java.awt.Color(0, 0, 0));
reset.setLabel("Reset");

reset.addActionListener(new java.awt.event.ActionListener()
   {
          public void actionPerformed(java.awt.event.ActionEvent evt)
```

```
            {
                resetActionPerformed(evt);
                    }
              });

            jPan.add(reset);
            reset.setBounds(120, 150, 80, 27);

            exit.setBackground(new java.awt.Color(255, 255, 255));
            exit.setFont(new java.awt.Font("Arial", 0, 14));
            exit.setForeground(new java.awt.Color(0, 0, 0));
            exit.setLabel("Exit");
            exit.addActionListener(new java.awt.event.ActionListener()
              {
                    public void actionPerformed(java.awt.event.ActionEvent
evt)
                    {
                        exitActionPerformed(evt);
                            }
                });

            jPan.add(exit);
            exit.setBounds(220, 150, 70, 27);

            jPan.add(pass);
            pass.setBounds(140, 91, 100, 30);
```

```
            getContentPane().add(jPan);
            jPan.setBounds(60, 100, 320, 200);

            jLabel3.setFont(new java.awt.Font("Arial", 0, 20));
            jLabel3.setForeground(new java.awt.Color(255, 0, 0));
            jLabel3.setText("Privacy-Preserving of Cloud Storage Security");
            getContentPane().add(jLabel3);
            jLabel3.setBounds(200, 10, 700, 30);
                    JLabel imageLabel = new JLabel();
        ImageIcon ii = new
ImageIcon(LoginDialog.class.getResource("images/network.JPG"));
        imageLabel.setIcon(ii);
        imageLabel.setBounds(400,50,481,241);
        imageLabel.setBackground(new Color(0, 116, 232));
        this.getContentPane().add(imageLabel);


            pack();

            jPanel1.setLayout(null);
        jPanel1.setBackground(new java.awt.Color(0, 116, 232));
            jPanel1.setBorder(new javax.swing.border.EtchedBorder());
            listres.setBorder(new
javax.swing.border.SoftBevelBorder(javax.swing.border.BevelBorder.LOWERED
));
            jPanel1.add(listres);
            listres.setBounds(20, 50, 150, 220);
```

```
jLabel1.setFont(new java.awt.Font("Arial", 0, 17));
        jLabel1.setForeground(new java.awt.Color(255, 255, 255));
jLabel1.setText("Resources Available:");
jPanel1.add(jLabel1);
jLabel1.setBounds(20, 5, 200, 40);

download.setBackground(new java.awt.Color(255, 255, 255));
download.setFont(new java.awt.Font("Arial", 0, 15));
        download.setForeground(new java.awt.Color(0, 0, 0));
download.setText("Download");

download.addActionListener(new java.awt.event.ActionListener()
  {
         public void actionPerformed(java.awt.event.ActionEvent evt)
        {
                downloadActionPerformed(evt);
              }
    });

jPanel1.add(download);
download.setBounds(180, 60, 110, 27);

getContentPane().add(jPanel1);
jPanel1.setBounds(20, 40, 340, 280);
```

```
        getContentPane().add(resArea);
        resArea.setBounds(20, 330, 450, 200);

        save.setBackground(new java.awt.Color(255, 0, 0));
        save.setFont(new java.awt.Font("Arial", 0, 15));
                save.setForeground(new java.awt.Color(255, 255, 255));
        save.setText("Save");
        save.addActionListener(new java.awt.event.ActionListener()
          {
                 public void actionPerformed(java.awt.event.ActionEvent evt)
                {

                    saveActionPerformed(evt);

                        }
          });

        getContentPane().add(save);
        save.setBounds(500, 340, 70, 27);
       resArea.setVisible(false);
       save.setVisible(false);

        jPanel1.setVisible(false);
        setBackground(new java.awt.Color(0, 204, 204));

}
//GEN-END:initComponents
```

```
        private void exitActionPerformed(java.awt.event.ActionEvent evt)
             {
            //GEN-FIRST:event_exitActionPerformed

                 System.exit(0);
              }


        private void resetActionPerformed(java.awt.event.ActionEvent evt)
            {
            //GEN-FIRST:event_resetActionPerformed

                user.setText("");
                pass.setText("");
              }

   private void submitActionPerformed(java.awt.event.ActionEvent evt)
       {
       try
       {

            ref= (RMISIntf)Naming.lookup("rmi://"+"127.0.0.1"+"/RMIServer");
            //System.out.println(ref.res("sdsd","dsa"));


if(user.getText().trim().equals(""))
```

```
                    {
                      JOptionPane.showConfirmDialog(this,"Enter The User
Name","Alert",JOptionPane.DEFAULT_OPTION,JOptionPane.ERROR_MESSA
GE);
                      user.grabFocus();
                    }
        else
        {
                      if(pass.getText().trim().equals(""))
                            {
                                  JOptionPane.showConfirmDialog(this,"Enter The
PassWord","Alert",JOptionPane.DEFAULT_OPTION,JOptionPane.ERROR_MES
SAGE);
                             pass.grabFocus();
                             }
                    else
                    {
                          InetAddress Address = InetAddress.getLocalHost();
                          String c =Address.getHostAddress();
                          ref=
(RMISIntf)Naming.lookup("rmi://"+"127.0.0.1"+"/RMIServer");
                                  String
ss=ref.CliDet(user.getText(),pass.getText(),c);
                                  if(ss.equals("notok"))
                                  {

        JOptionPane.showConfirmDialog(this,"UnAuthorised
```

```
User","Alert",JOptionPane.DEFAULT_OPTION,JOptionPane.ERROR_MESSAG
E);
                                        user.grabFocus();
                                        }

else
{

username=user.getText();
String key=(String)JOptionPane.showInputDialog(this,"Enter your Key:");
ref=(RMISIntf)Naming.lookup("rmi://"+"127.0.0.1"+"/RMIServer");
String status=ref.CliDet_key(user.getText(),key);
        if(status.equals("ok"))
        {

        username=user.getText();
        //getContentPane().remove(jPan);//.visible(false);
        jPan.setVisible(false);
        jPanel1.setVisible(true);
        resArea.setVisible(true);
        save.setVisible(true);
        StringTokenizer token=new StringTokenizer(ss,";");
        while(token.hasMoreTokens())
        {
        String nextToken = token.nextToken();
        v.addElement(nextToken);
        System.out.println(nextToken);
```

```
        }
        listres.setListData(v);
        }///inner if


        else
        {
        JOptionPane.showConfirmDialog(this,"UnAuthorised
User","Alert",JOptionPane.DEFAULT_OPTION,JOptionPane.ERROR_MESSAG
E);
                                        user.grabFocus();
        }
}

}

                }

        }catch(Exception e){System.out.println(e);}



   }

   private void downloadActionPerformed(java.awt.event.ActionEvent evt)
                {
                //System.out.println(username);
```

```
//GEN-FIRST:event_downloadActionPerformed

    sss=(String)listres.getSelectedValue();
    System.out.println(sss);
int ch=0;
try{
        InetAddress Address = InetAddress.getLocalHost();
        String c =Address.getHostAddress();
        ////////////////////////////////////////////////////

        String aa[]=new String[4];
        ref=
(RMISIntf)Naming.lookup("rmi://"+"127.0.0.1"+"/RMIServer");
//////////////////////////
String chh=ref.checkipp(c,sss);
StringTokenizer st=new StringTokenizer(chh,";");
String ipp=null;
String opp=null;
//while(st.hasMoreTokens()) {

         ipp=st.nextToken();
        System.out.println(ipp+"hai");

        opp=st.nextToken();
        System.out.println(opp);

        //}
```

```
            String optiontest="null";
                if(ipp.equals("ok")||opp.equals("true"))
                {
                optiontest="true";

                String a=(String)JOptionPane.showInputDialog(this,"Enter
your password:");
                        aa[0]=a;
                        a=(String)JOptionPane.showInputDialog(this,"Enter your
correct password:");
                        aa[1]=a;
                        a=(String)JOptionPane.showInputDialog(this,"Enter
correct password:");
                        aa[2]=a;
                        System.out.println(aa[0]+aa[1]);
                        aa[3]=new Date().toString();
                ref.store(aa,c,username,sss);
                }
/////////change
                String ssss=ref.res(sss,c,optiontest);
                /********File ss=ref.res(sss,c,);

               //File file = new File(ss);
                FileInputStream in = new FileInputStream(ss);
                int i = in.available();
                char st[]=new char[i];
```

```
                int j =0;
                while((ch=in.read())!=-1)
                        {
                                st[j] = (char) ch;
                                j++;
                        }
                     String str=new String(st);
                  resArea.setText(str);
                                **********/
                      System.out.println("new  "+ssss);
                  resArea.setText(ssss);

            } catch(Exception e)
                        {
                         System.out.println(e);
                        }

   }


   private void saveActionPerformed(java.awt.event.ActionEvent evt)
            {
                //GEN-FIRST:event_saveActionPerformed

            try {
                FileDialog fd=new FileDialog(this,"File Store",
FileDialog.SAVE);
```

```
                    fd.setVisible(true);
                    String f= fd.getFile();
                    fd.setFile(f); // Filename filter
                    fd.setDirectory("."); // Current directory
                    //fd.show();
                    FileOutputStream out=new FileOutputStream(f);
                    String s=resArea.getText();
                    System.out.println(s);
                    byte b[]=s.getBytes();
                    out.write(b);
               } catch(IOException e)
                    {
                    System.out.println(e);
                    }
}

/** Exit the Application */

private void exitForm(java.awt.event.WindowEvent evt)
      {
     //GEN-FIRST:event_exitForm
          System.exit(0);
     }

/**
 * @param args the command line arguments
 */
```

```
public static void main(String args[]) {
      JFrame jf1=new Login();
       jf1.setResizable(false);
       jf1.setSize(900,600);
       jf1.setTitle("Privacy-Preserving of Cloud Storage Security");
      jf1.show();



}

// Variables declaration - do not modify//GEN-BEGIN:variables
private javax.swing.JButton exit;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel3;
private javax.swing.JPanel jPan;
private javax.swing.JPasswordField pass;
private javax.swing.JButton reset;
private javax.swing.JButton submit;
private javax.swing.JTextField user;
private javax.swing.JButton download;
private javax.swing.JLabel jLab;
private javax.swing.JPanel jPanel1;
private javax.swing.JList listres;
private java.awt.TextArea resArea;
private javax.swing.JButton save;
```

```
    // End of variables declaration//GEN-END:variables

}
```

Passwordgenerator.java

```
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;
import java.util.Random;

/**

* PasswordGenerator- Class to generate password of 8 character length.
* Passwords will contain at least (1) upper case letter.
* Passwords will contain at least (1) lower case letter.
* Passwords will contain at least (1) number.
* Passwords will contain at least (1) special character.
* Passwords will contain at least (8) characters in length.
*/

public class PasswordGenerator {

/**
* Array to store number characters.
*/
char numberChars[] = "0123456789".toCharArray();
```

```
/**
* Array to store lower case characters.
*/
char lowerChars[] = "abcdefghijklmnopqrstuvwxyz".toCharArray();

/**
* Array to store upper case characters.
*/
char upperChars[] = "ABCDEFGHIJKLMNOPQRSTUVWXYZ".toCharArray();

/**
* Array to store special characters.
*/
char otherSpecialChars[] = "`~!@#$%^&*()-_=+[{]}\\|;:'\",<.>/?".toCharArray();

/**
* List contains set of number to get from each array, total of all is the length of password i.e. 8
*/
List <Integer>listForLengthOfPassword = new ArrayList<Integer>();

/**
* Random object to generate the random number to pick up random character from array.
*/
Random random = new Random();
```

```
/**
* Final password string object holder list.
*/
List <Character>finalpasswordList;

/**
* Constructor.
*/
public PasswordGenerator(){

listForLengthOfPassword.add(1);
listForLengthOfPassword.add(2);
listForLengthOfPassword.add(2);
listForLengthOfPassword.add(3);
}

/**
* getPassword() creates password by picking random character from each array.
Every time it is invoked,
* shuffle the number of character to take from each array dynamically.
* @return string as generated password.
*/
public String getPassword(){

Collections.shuffle(listForLengthOfPassword);
finalpasswordList = new ArrayList<Character>();
```

```
for(int t=0;t<listForLengthOfPassword.size();t++){

int numberOfCharPerArray = listForLengthOfPassword.get(t);
for(int z=0;z<numberOfCharPerArray;z++){

   switch (t) {
      case 0:
         finalpasswordList.add(numberChars[random.nextInt(10)]);
         break;
      case 1:
         finalpasswordList.add(lowerChars[random.nextInt(26)]);
         break;
      case 2:
         finalpasswordList.add(upperChars[random.nextInt(26)]);
         break;
      case 3:
         finalpasswordList.add(otherSpecialChars[random.nextInt(32)]);
         break;
      default:
         break;
   }

}//end for

}//end for

String password = new String();
```

```
Collections.shuffle(finalpasswordList);
for(int s=0;s<finalpasswordList.size();s++){

password += finalpasswordList.get(s);
}
return password;
}

/**
* Generates 100 password for testing.
* @param args
*/
public static void main(String args[]){

PasswordGenerator passwordGenerator = new PasswordGenerator();

//for(int u=0; u<1000; u++)
{

System.out.println(passwordGenerator.getPassword());
}
}

}
```

Server.java

```
import java.awt.*;
```

```
import java.awt.event.*;
import java.net.MalformedURLException;
import java.rmi.Naming;
import java.rmi.RemoteException;
import java.util.StringTokenizer;
import java.util.Vector;
import javax.swing.*;
import javax.swing.border.*;

public class Server extends JFrame
{

   private final JPanel contentpane;
   private final JPanel RemoveIP;
   private final JPanel jPanel2;
        private final JButton AddIP;
   private final JButton AddRes;
   private final JButton RemoveRes;
   private final JButton StartSer;
   private final JButton Startview;
   private final JButton Startmobilealert;
   private final JButton Adduser;
   private final JButton removeIP;
   private final JButton addIP;
   private final JButton addRes;
   private final JLabel jLabel1;
   private final JLabel jLabel2;
```

```
    private final JLabel headlabl;
          private final JLabel headlab2;
    private final JList jList1;
    private final JList jList2;

    private final JOptionPane op;
    private static final JFrame frame = null;

    public Server()
    {
   op=new JOptionPane();
        contentpane = (JPanel) this.getContentPane();
        StartSer = new JButton();
        Startview = new JButton();
       Startmobilealert=new JButton();
        Adduser=new JButton();
        RemoveIP = new JPanel();
        jLabel1 = new JLabel();
        jList1 = new JList();
        AddIP = new JButton();
        removeIP = new JButton();
///
        addIP = new JButton();
        jPanel2 = new JPanel();
        jLabel2 = new JLabel();
        headlabl=new JLabel();
                    headlab2=new JLabel();
```

```
        jList2 = new JList();
        AddRes = new JButton();
        RemoveRes = new JButton();
        addRes = new JButton();
//////////
        contentpane.setLayout(null);
        this.setSize(new Dimension(900, 620));
//        this.setResizable(false);
        this.setTitle("Privacy-Preserving of Cloud Storage Security");
          headlabl.setText("Privacy-Preserving of Cloud Storage Security");
          headlab2.setText("Cloud Authentication Server ");

          headlabl.setForeground(new Color(255, 0, 10 ));
          //headlabl.setBackground(new Color(255, 255, 255));
                  headlabl.setFont(new Font("Arial", 0, 20));
          headlabl.setBounds(200, 0, 800, 30);
          contentpane.add(headlabl,null);
        //setBackground(new Color(254, 253, 184));

                  headlab2.setForeground(new Color(255, 100, 0));
          //headlabl.setBackground(new Color(255, 255, 255));
                  headlab2.setFont(new Font("Arial", 0, 20));
          headlab2.setBounds(300, 15, 500, 70);
          contentpane.add(headlab2,null);

        addWindowListener(new WindowAdapter() {
```

```
        @Override
        public void windowClosing(WindowEvent windowevent)
        {
            exitForm(windowevent);
        }

    });

    StartSer.setBackground(new Color(36, 72, 72));
    StartSer.setFont(new Font("Arial", 0, 14));
    StartSer.setForeground(new Color(255, 255, 255));
    StartSer.setText("Start Server");
    StartSer.setBorder(new BevelBorder(0));
    StartSer.addActionListener(new ActionListener() {

        public void actionPerformed(ActionEvent actionevent)
        {
            StartSerActionPerformed(actionevent);
        }

    });
    contentpane.add(StartSer,null);
    StartSer.setBounds(30, 65, 140, 23);
///
    Startview.setBackground(new Color(36, 72, 72));
    Startview.setFont(new Font("Arial", 0, 14));
    Startview.setForeground(new Color(255, 255, 255));
```

```
        Startview.setText("Hackers Info");
        Startview.setBorder(new BevelBorder(0));
        Startview.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent actionevent)
            {
                StartviewActionPerformed(actionevent);
            }

        });
        contentpane.add(Startview,null);
        Startview.setBounds(210, 65, 140, 23);

//////////
        Startmobilealert.setBackground(new Color(36, 72, 72));
        Startmobilealert.setFont(new Font("Arial", 0, 14));
        Startmobilealert.setForeground(new Color(255, 255, 255));
        Startmobilealert.setText("Enable Mobile Alert");
        Startmobilealert.setBorder(new BevelBorder(0));
        Startmobilealert.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent actionevent)
            {
                StartmobilealertActionPerformed(actionevent);
            }

        });
```

```
        contentpane.add(Startmobilealert,null);
        Startmobilealert.setBounds(380, 65, 140, 23);

///
//////////
        Adduser.setBackground(new Color(36, 72, 72));
        Adduser.setFont(new Font("Arial", 0, 14));
        Adduser.setForeground(new Color(255, 255, 255));
        Adduser.setText("Add Newuser");
        Adduser.setBorder(new BevelBorder(0));
        Adduser.addActionListener(new ActionListener() {

           public void actionPerformed(ActionEvent actionevent)
           {
              AdduserActionPerformed(actionevent);
           }

        });
        contentpane.add(Adduser,null);
        Adduser.setBounds(560, 65, 140, 23);

///
        RemoveIP.setLayout(null);
        RemoveIP.setBackground(new Color(0, 116, 232));
        RemoveIP.setBorder(new EtchedBorder());
        jLabel1.setFont(new Font("Arial", 0, 17));
                jLabel1.setForeground(new Color(255, 255, 255));
```

```
        jLabel1.setText("Restricted IP Addresses:");
        RemoveIP.add(jLabel1);
        jLabel1.setBounds(20, 10, 200, 40);
        jList1.setBorder(new BevelBorder(2));
        RemoveIP.add(jList1);
        jList1.setBounds(10, 70, 220, 450);
        AddIP.setBackground(new Color(205, 205, 205));
        AddIP.setFont(new Font("Arial", 0, 14));
        AddIP.setForeground(new Color(0, 0, 0));
        AddIP.setText("View");
        AddIP.setBorder(new SoftBevelBorder(0));
        AddIP.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent actionevent)
            {
                AddIPActionPerformed(actionevent);
            }

        });
        RemoveIP.add(AddIP);
        AddIP.setBounds(250, 70, 110, 25);
    //new ip
        removeIP.setBackground(new Color(205, 205, 205));
        removeIP.setFont(new Font("Arial", 0, 15));
        removeIP.setForeground(new Color(0, 0, 0));
        removeIP.setText("Remove IP");
        removeIP.setBorder(new SoftBevelBorder(0));
```

```
        removeIP.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent actionevent)
            {
                removeIPActionPerformed(actionevent);
            }

        });
        RemoveIP.add(removeIP);
        removeIP.setBounds(250, 110, 110, 25);

//////////////new addip
        addIP.setBackground(new Color(205, 205, 205));
        addIP.setFont(new Font("Arial", 0, 15));
        addIP.setForeground(new Color(0, 0, 0));
        addIP.setText("Add New IP");
        addIP.setBorder(new SoftBevelBorder(0));
        addIP.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent actionevent)
            {
                addIPActionPerformed(actionevent);
            }

        });
        RemoveIP.add(addIP);
        addIP.setBounds(250, 150, 110, 25);
```

```
////////////
        contentpane.add(RemoveIP,null);
        RemoveIP.setBounds(30, 90, 400, 550);
        jPanel2.setLayout(null);
        jPanel2.setBackground(new Color(0, 116, 232));
        jPanel2.setBorder(new EtchedBorder());
        jLabel2.setFont(new Font("Arial", 0, 17));
                jLabel2.setForeground(new Color(255, 255, 255));
        jLabel2.setText("Available Resources:");
        jPanel2.add(jLabel2);
        jLabel2.setBounds(10, 10, 200, 40);
        jList2.setBorder(new BevelBorder(2));
        jPanel2.add(jList2);
        jList2.setBounds(10, 70, 220, 450);
        AddRes.setBackground(new Color(205, 205, 205));
        AddRes.setFont(new Font("Arial", 0, 15));
        AddRes.setForeground(new Color(0, 0, 0));
        AddRes.setText("View");
        AddRes.setBorder(new SoftBevelBorder(0));
        AddRes.addActionListener(new ActionListener() {

           public void actionPerformed(ActionEvent actionevent)
           {
              AddResActionPerformed(actionevent);
           }
```

```
        });
        jPanel2.add(AddRes);
        AddRes.setBounds(250, 70, 110, 25);
        RemoveRes.setBackground(new Color(205, 205, 205));
        RemoveRes.setFont(new Font("Arial", 0, 15));
        RemoveRes.setForeground(new Color(0, 0, 0));
        RemoveRes.setText("Remove ");
        RemoveRes.setBorder(new SoftBevelBorder(0));
        RemoveRes.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent actionevent)
            {
                RemoveResActionPerformed(actionevent);
            }

        });
        jPanel2.add(RemoveRes);
        RemoveRes.setBounds(250, 110, 110, 25);
//////////////new Res
          addRes.setBackground(new Color(205, 205, 205));
        addRes.setFont(new Font("Arial", 0, 15));
        addRes.setForeground(new Color(0, 0, 0));
        addRes.setText("Add New File");
        addRes.setBorder(new SoftBevelBorder(0));
        addRes.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent actionevent)
```

```
            {
                addResActionPerformed(actionevent);
            }

        });
        jPanel2.add(addRes);
        addRes.setBounds(250, 150, 110, 25);
//////////////////
        contentpane.add(jPanel2,null);
        jPanel2.setBounds(470, 90, 400, 550);
        pack();
    }

    private void StartSerActionPerformed(ActionEvent actionevent)
    {
        try
        {
            RMISImpl rmisimpl = new RMISImpl();
            Naming.rebind("RMIServer", rmisimpl);
            System.out.println("Server Is Runnig...");
        }
        catch(MalformedURLException exception)
        {
            System.out.println("Exception :" + exception);
        } catch (RemoteException exception) {
            System.out.println("Exception :" + exception);
        }
```

```
    }

//
    private void StartmobilealertActionPerformed(ActionEvent actionevent)
    {
        try
        {
emserver ob=new emserver();
ob.start();
       System.out.println("Mobile Alert Service Running..... ");
        }
        catch(Exception exception)
        {
            System.out.println("Exception :" + exception);
        }
    }
//
//
private void AdduserActionPerformed(ActionEvent actionevent)
    {
        try
        {

          new NewUser().show();
           }
        catch(Exception exception)
        {
```

```
            System.out.println("Exception :" + exception);
        }
    }

//

//
private void StartviewActionPerformed(ActionEvent actionevent)
    {
      new view();
    }
//
    private void RemoveResActionPerformed(ActionEvent actionevent)
    {    dbcon dbcon1 = new dbcon();
         String sss=(String) jList2.getSelectedValue();
         System.out.println("selected:"+sss);
         dbcon1.removede(sss,"RESOURCES","files");
         rss();
    }
    private void addResActionPerformed(ActionEvent actionevent)
    {
         String Fn=(String)JOptionPane.showInputDialog(this,"Enter The new
FileName:");
         String fFn=(String)JOptionPane.showInputDialog(this,"Enter The fake
FileName:");
         String op1=(String)JOptionPane.showInputDialog(this,"Enter The option of
FileSecurity:","false");
```

```
        String ip=(String)JOptionPane.showInputDialog(this,"Enter The IP
Address:");

if(Fn==null || fFn==null || op1==null ||Fn.equals("")||fFn.equals("")||op1.equals(""))
        {
        JOptionPane.showConfirmDialog(this,"Enter proper Data
Only","Alert",JOptionPane.DEFAULT_OPTION,JOptionPane.ERROR_MESSAG
E);
        }
else
        {
        add_rss(Fn,fFn,op1,ip);
        }

   }

private void addIPActionPerformed(ActionEvent actionevent)
   {
String a=(String)JOptionPane.showInputDialog(this,"Enter The new IPAddress:");
new_addipr(a);
System.out.println("addip called"+a);
   }
private void rss()
{
dbcon dbcon1 = new dbcon();
       String s = dbcon1.listfile();
       Vector vector = new Vector(2);
```

```
        String s1;
        for(StringTokenizer stringtokenizer = new StringTokenizer(s, ";");
stringtokenizer.hasMoreTokens(); vector.addElement(s1))
        {
            s1 = stringtokenizer.nextToken();
        }

        jList2.setListData(vector);

}
private void add_rss(String f,String ff,String op,String allowedip)
{
        dbcon dbcon1 = new dbcon();
       dbcon1.add_res(f,ff,op,allowedip);
        String s = dbcon1.listfile();
        Vector vector = new Vector(2);
        String s1;
        for(StringTokenizer stringtokenizer = new StringTokenizer(s, ";");
stringtokenizer.hasMoreTokens(); vector.addElement(s1))
        {
            s1 = stringtokenizer.nextToken();
        }

        jList2.setListData(vector);

}
```

```
    private void AddResActionPerformed(ActionEvent actionevent)
    {

rss();
    }
private void ipr()
{
        dbcon dbcon1 = new dbcon();
         String s = dbcon1.listip();
          Vector vector = new Vector(2);
          String s1;
          for(StringTokenizer stringtokenizer = new StringTokenizer(s, ";");
stringtokenizer.hasMoreTokens(); vector.addElement(s1))
                {
                        s1 = stringtokenizer.nextToken();
                }

      jList1.setListData(vector);



}

private void new_addipr(String ip_n)
{
        dbcon dbcon1 = new dbcon();
        dbcon1.add_ip(ip_n);
```

```
        String s = dbcon1.listip();
        Vector vector = new Vector(2);
        String s1;
        for(StringTokenizer stringtokenizer = new StringTokenizer(s, ";");
stringtokenizer.hasMoreTokens(); vector.addElement(s1))
                {
                        s1 = stringtokenizer.nextToken();
                }

      jList1.setListData(vector);


}



   private void removeIPActionPerformed(ActionEvent actionevent)
   {
        String a=(String)JOptionPane.showInputDialog(this,"Enter The Key
Word");
        if(a.equals("admin"))
        {
        dbcon dbcon1 = new dbcon();
        String sss=(String) jList1.getSelectedValue();
        System.out.println("selected:"+sss);
        dbcon1.removede(sss,"ADDRESS","IPADDRESS");
        ipr();
```

```
        }
        else
        {
        JOptionPane.showConfirmDialog(this,"Check The Key or Select
Ip","Alert",JOptionPane.DEFAULT_OPTION,JOptionPane.ERROR_MESSAGE);
        }

    }

    private void AddIPActionPerformed(ActionEvent actionevent)
    {
        ipr();
    }



    private void exitForm(WindowEvent windowevent)
    {
       System.exit(0);
    }
static void showMiddle(){
    Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();
    Dimension frameSize = frame.getSize();
    if (frameSize.height > screenSize.height) {
      frameSize.height = screenSize.height;
    }
    if (frameSize.width > screenSize.width) {
      frameSize.width = screenSize.width;
```

```
    }
    frame.setLocation((screenSize.width - frameSize.width) / 2, (screenSize.height -
frameSize.height) / 2);
    frame.setVisible(true);

}

    public static void main(String args[])
    {
        Server frame = new Server();//).show();
        ImageIcon image1 = new
ImageIcon(LoginDialog.class.getResource("images/network.JPG"));
        LoginDialog dia = new LoginDialog();
        JSplashScreen sp = new JSplashScreen(image1);
        sp.displayForm();
        dia.setSize(780,580);
        dia.showDialog();
        if(dia.getConfirm()){
                sp.closeSplashScreen();
                //frame.setVisible(true);
                frame.setSize(900,700);
                ////
                frame.show();//Middle();

        }
        else {
                sp.closeSplashScreen();
```

```
            System.exit(1);
    }
```

8. RESULT & OUTPUT

OBJECTIVES

1.Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

3.When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user

will not be in maize of instant. Thus, the objective of input design is to create an input layout that is easy to follow-

INPUT DESIGN

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

OUTPUT DESIGN

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source

information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.

2.Select methods for presenting information.

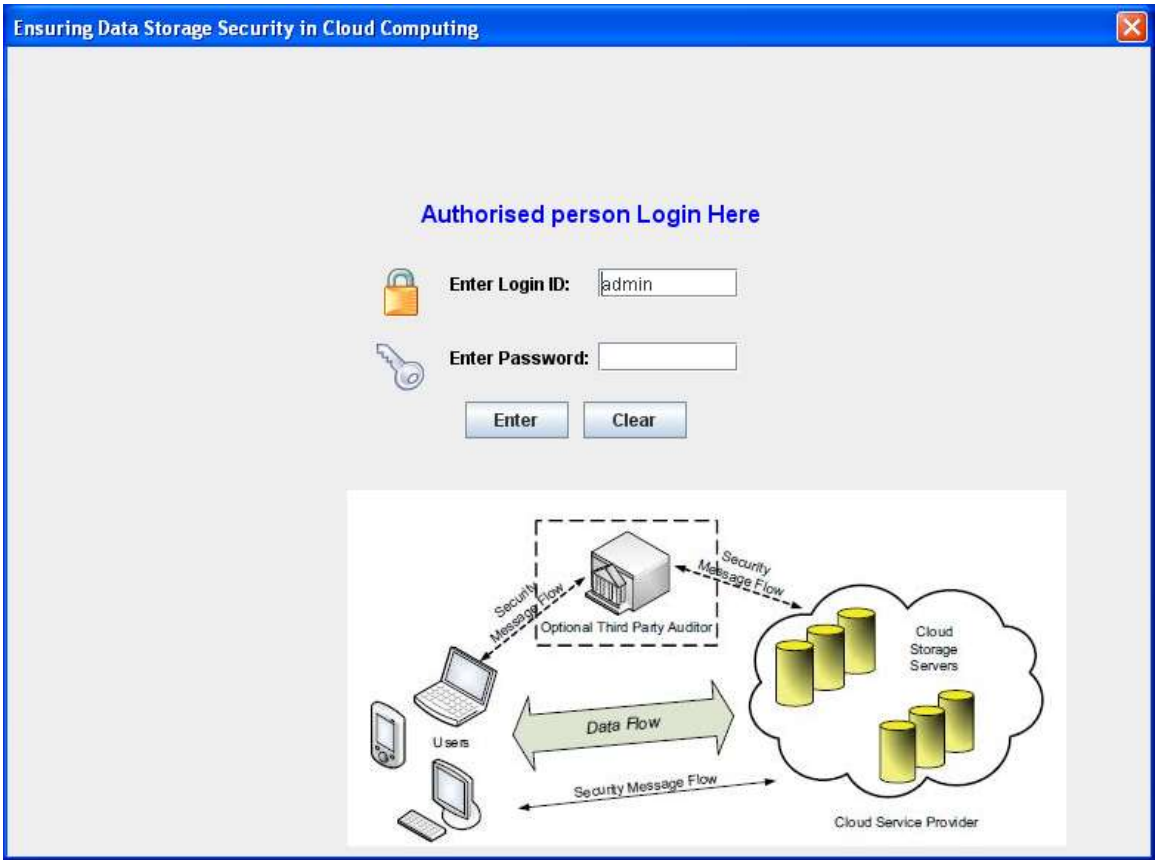
3.Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

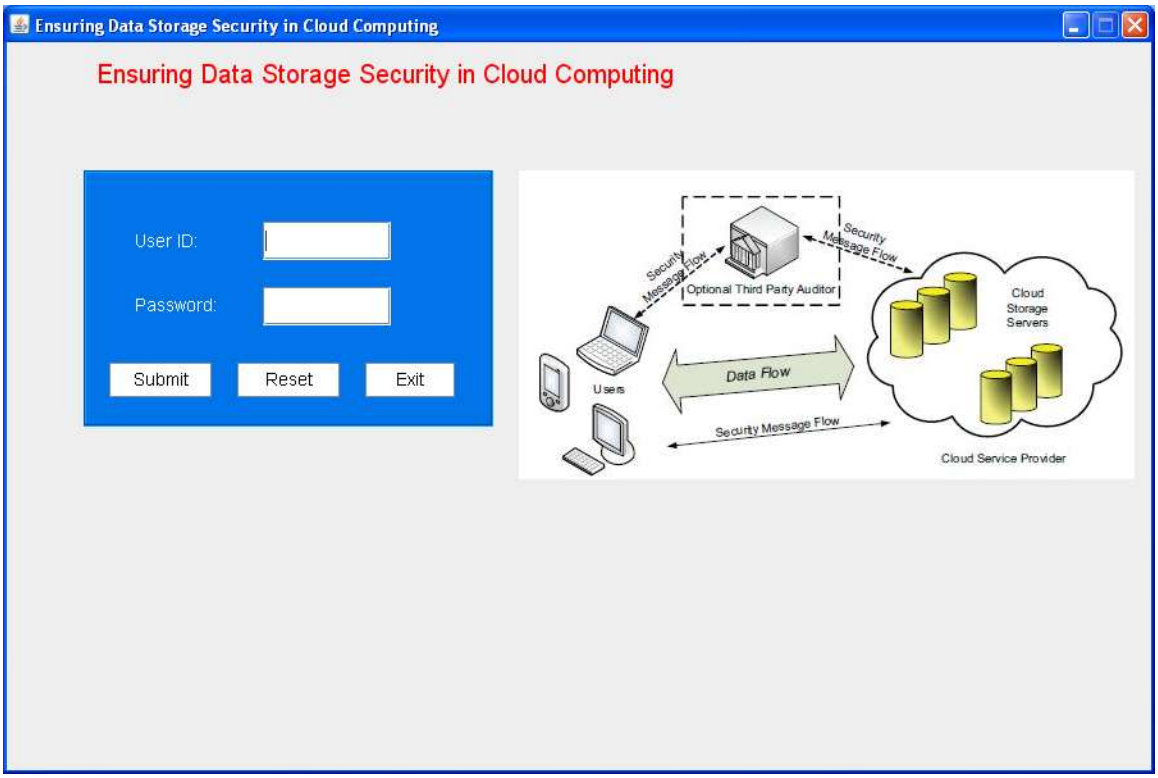
- Convey information about past activities, current status or projections of the
- Future.
- Signal important events, opportunities, problems, or warnings.
- Trigger an action.
- Confirm an act.

Screen Shots

Cloud Server Login



Client side Login



System info


Ensuring Data Storage Security in Cloud Computing
Cloud Authentication Server
Start Server
Hackers Info
Enable Mobile Alert
Add Newuser
Restricted IP Addresses:
skcomput-w3sdnt
192.168.100.10211
127.7.8.6
133.23.33.44
128.68.100.57
192.168.100.57
123.87.100.23
192.168.100.78
192.168.1.55
View
Remove IP
Add New IP
Available Resources:
asd.txt
demo.txt
hi.txt
output.txt
sample.txt
sds.txt
skdotcom.txt
skinfo.txt
sss.txt
x1.txt
xx.txt
xx2.txt
View
Remove
Add New File

Adding or removing users

9. CONCLUSION

In this project, we investigated the problem of data security in cloud data storage, which is essentially a distributed storage system. To ensure the correctness of users' data in cloud data storage, we proposed an effective and flexible distributed scheme with explicit dynamic data support, including block update, delete, and append. We rely on erasure-correcting code in the file distribution preparation to provide redundancy parity vectors and guarantee the data dependability. By utilizing the homomorphic token with distributed verification of erasure coded data, our scheme achieves the integration of storage correctness insurance and data error localization, i.e., whenever data corruption has been detected during the storage correctness verification across the distributed servers, we can almost guarantee the simultaneous identification of the misbehaving server(s). Through detailed security and performance analysis, we show that our scheme is highly efficient and resilient to Byzantine failure, malicious data modification attack, and even server colluding attacks. We believe that data storage security in Cloud Computing, an area full of challenges and of paramount importance, is still in its infancy now, and many research problems are yet to be identified. We envision several possible directions for future research on this area. The most promising one we believe is a model in which public verifiability is enforced. Public verifiability, supported in allows TPA to audit the cloud data storage without demanding users' time, feasibility or resources. An interesting question in this model is if we can construct a scheme to achieve both public verifiability and storage correctness assurance of dynamic data. Besides, along with our research on dynamic cloud data storage, we also plan to investigate the problem of fine-grained data error localization.

10. REFERENCES

https://www.sciencedirect.com/science/article/pi/S187705095812

https://www.researchgate.net/publication/3060422_A_Study_on_Data_Storage_Security_Issues_in_Cloud_Computing

https://www.researchgate.net/publication/300058400_Security_Techniques_for_Data_Protection_in_Cloud_Computing

Amazon.com, “Amazon Web Services (AWS),” Online at http://aws. amazon.com, 2008. [2] N. Gohring, “Amazon’s S3 down for several hours,” Online at http://www.pcworld.com/businesscenter/article/142549/amazons s3 down for several hours.html