ISSN 2278 - 3091
International Journal on Advanced Trends in Computer Science and Engineering (IJATCSE), Vol. 4 No.1, Pages : 19 – 27 (2015)
Special Issue of ICACE 2015 - Held on March 02, 2015,Mysore, India
http://warse.org/pdfs/2015/icace2015sp05.pdf

Security and Privacy in Cloud Computing To Guarantee Simultaneous Localization of Data Errors

Nandini Prasad K S[1], Chetana Srinivas[2]
[1]Dr. Ambedkar Institute of Technology, INDIA, nandiniks1@gmail.com
[2]East West Institute of Technology, chetanasrinivas1@gmail.com

Abstract: *Cloud computing is the arising technology that delivers software, platform and infrastructure as a service over a network. Cloud minimizes the burden of users by allowing them to remotely store their data and eliminates the need of local storage and maintenance. In order to overcome the threat of integrity, user can entrust third party auditor to assess the risk of outsoured data when needed. In our project we have used cryptographic hash function to verify integrity of data along with key generation mechanism and mutual authentication with TPA (Third Party Auditor). However, when outsourcing the data and business application to a third party causes the security and privacy issues to become a critical concern. We have identified three most representative security and privacy attributes (i.e., Integrity, Accountability and privacy-preservability).*

In our paper, we propose an effective and flexible cryptographic scheme with explicit dynamic data support to ensure the correctness of user's data in the cloud. Our paper 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). These scheme further supports secure and efficient dynamic operations such as data upload, download and append.

Keywords: Cloud, Integrity, Security, Resources, Localization.

INTRODUCTION

Cloud computing has begun to emerge as a hotspot in both industry and academia. It represents a new business model and computing paradigm, which enables on-demand provisioning of computational and storage resources. Economic benefits consist of the main drive for cloud computing due to the fact that cloud computing offers an effective way to reduce capital expenditure (CapEx) and operational expenditure (OpEx). The definition of cloud computing has been given in many literatures, but nothing has gained wide recognition. Throughout this, defines cloud computing as: "A large-scale distributed computing paradigm that is driven by economies of scale, in which a pool of abstracted, virtualized, dynamically scalable, managed computing power, storage, platforms, and services are delivered on demand to external customers over the Internet."

The data are encrypted under a certain key, which is shared only with the authorized users. The unauthorized users, including the CSP (Customer Service Provider), are unable to access the data because they do not have the decryption key. It provides data storage security on untrusted remote servers. The remotely stored data can be not only accessed by authorized users, but also updated and scaled by the owner. After updating, authorized users should receive the latest version of the data i.e., a technique is required to detect whether the received data are stale. Mutual trust between the data owner and the CSP is another imperative issue, which is addressed in the proposed scheme. A mechanism is introduced to determine the dishonest party, i.e., misbehavior from any side is detected and the responsible party is identified. Last but not least, the access control is considered, which allows the owner to grant or revoke access rights to the outsourced data. The design and implementation of a cloud-based storage scheme that has the following features:

1. It allows a data owner to outsource the data to a CSP, and perform full dynamic operations at the block-level, i.e., it supports operations such as block modification, insertion, and append.
2. It ensures the newness property, i.e., the authorized users receive the most recent version of the outsourced data.
3. It establishes indirect mutual trust between the data owner and the CSP because each party resides in a different trust domain and it enforces the access control for the outsourced data.

Cloud computing is one of the fasted growing segments of the IT industry. However, as more information on individuals and companies is placed in the cloud, companies must address cloud computing security issues**.** As with other major business decisions, an enterprise must evaluate the benefits and be prepared to address any risks and challenges cloud adoption brings. Moving applications to the cloud and accessing the benefits means first evaluating specific data security issues and cloud security issues.

When enterprises move applications from on-premise to cloud-based, challenges arise from data residency, industry compliance requirements, privacy and third party obligations concerning the treatment of sensitive data. Corporate policies or the regulations of the governing jurisdictions impact the way sensitive data is managed including where it is located, what types of data can be collected and stored and who has access to it. Cloud security issues fall primarily into three areas:

- **Data Residency -** Many companies face legislation by their country of origin or the local country that the business entity is operating in, requiring certain types of data to be kept within defined geographic borders. There are specific regulations that must be followed, centered on data access, management and control.
- **Data privacy-** Business data often needs to be guarded and protected more stringently than non-sensitive data.

ISSN 2278 - 3091
International Journal on Advanced Trends in Computer Science and Engineering (IJATCSE), Vol. 4 No.1, Pages : 19 – 27 (2015)
Special Issue of ICACE 2015 - Held on March 02, 2015,Mysore, India
http://warse.org/pdfs/2015/icace2015sp05.pdf

The enterprise is responsible for any breaches to data and must be able ensure strict cloud security in order to protect sensitive information.

- **Industry & Regulation Compliance** - Organizations often have access to and are responsible for data that is highly regulated and restricted. It requires an enterprise to follow defined standards to safeguard private and business data and to comply with applicable laws.

The main objective of our paper is to provide data security and privacy for the cloud customer's data using cryptographic method. In terms of security, when storing data on the cloud one might want to make sure if the data is correctly stored and can be retrieved later. In terms of privacy, the cloud provider and the user have a mutual trust such that the cloud provider can be assured that the user is not some malicious hacker and the user can be assured of data consistency and data storage.

LITERATURE SURVEY

Cloud solutions are a relatively new concept. Wang et al. identified the difficulties and potential security problems of direct extensions with fully dynamic data updates from prior works and then showed how to construct an elegant verification scheme for the seamless integration salient features in their protocol design. In particular, to achieve efficient data dynamics, they improved the existing proof of storage models by manipulating the classic Merkle Hash Tree construction for block tag authentication. To support efficient handling of multiple auditing tasks, they further explored the technique of bilinear aggregate signature to extend our main result into a multiuser setting, where TPA can perform multiple auditing tasks simultaneously. Extensive security and performance analysis showed that the proposed schemes are highly efficient and provably secure. Juels et al. [3] defined a proper "proof of retrievability" (POR) model for guaranteeing the remote information integrity. Their theme combines spot-checking and error correcting code to make sure each possession and retrievability of files on archive service systems. Ateniese et al. [5] are the first to consider public auditability in their "provable data possession" PDP) model for ensuring possession of data files on untrusted storages. They have utilized the RSA-based homomorphic linear authenticators for auditing outsourced data and suggest randomly sampling a few blocks of the file. However, among their two proposed schemes, the one with public auditability exposes the linear combination of sampled blocks to external auditor. When used directly, their protocol is not provably privacy preserving, and thus may leak user data information to the external auditor. Besides it is often insufficient to detect the data corruption only when accessing the data, as it does not give users correctness assurance for those unaccessed data and might be too late to recover the data loss or damage.

CLOUD COMPUTING DATA STORAGE SYSTEM

The cloud computing storage model considered in this work consists of four main components as shown in the Fig.1.

1. A data owner that can be an organization generating sensitive data to be stored in the cloud and made available for controlled external use.

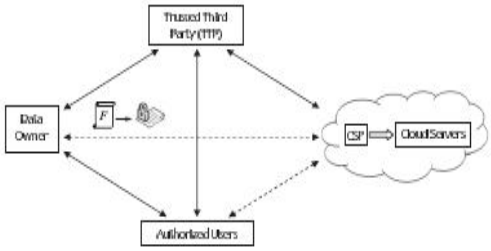


Fig. 1Cloud computing data storage system model

2. A CSP who manages cloud servers and provides paid storage space on its infrastructure to store the owner's files and make them available for authorized users.
3. Authorized users a set of owner's clients who have the right to access the remote data.
4. A Trusted Third Party (TTP), an entity who is trusted by all other system components, and has capabilities to detect/specify dishonest parties.

Fig. 1 shows the relations between different system components are represented by double sided arrows, where solid and dashed arrows represent trust and distrust relations, respectively. For example, the data owner, the authorized users, and the CSP trust the TTP. On the other hand, the data owner and the authorized users have mutual distrust relations with the CSP. Thus, the TTP is used to enable indirect mutual trust between these three components. There is a direct trust relation between the data owner and the authorized users. The auditing process of the data received from the CSP is done by authorized users, and we resort to the TTP only to resolve disputes that may arise regarding data integrity or newness. Reducing the storage overhead on the CSP side is economically a key feature to lower the fees paid by the customers. Moreover, decreasing the overall computation cost in the system is another crucial aspect.

SYSTEM ANALYSIS AND SAMPLE CODING

Cloud computing environments are multidomain environments in which each domain can use different security, privacy, and trust requirements and potentially employ various mechanisms, interfaces, and semantics. Such domains could represent individually enabled services or other infrastructural or application components. Service-oriented architectures are naturally relevant technology to facilitate such multidomain formation through service composition and orchestration. It is important to leverage existing research on multidomain policy integration and the secure-service composition to build a comprehensive policy-based management framework in cloud computing environments. As shown in Fig. 2, there are six specific areas of the cloud computing environment where equipment and software require substantial security attention. These six areas are:

(1) Security of data at rest, (2) Security of data in transit, (3)Authentication of users/applications/ processes, (4)Robust separation between data belonging to different

ISSN 2278 - 3091
International Journal on Advanced Trends in Computer Science and Engineering (IJATCSE), Vol. 4 No.1, Pages : 19 – 27 (2015)
Special Issue of ICACE 2015 - Held on March 02, 2015,Mysore, India
http://warse.org/pdfs/2015/icace2015sp05.pdf

customers, (5) Cloud legal and regulatory issues, and (6) Incident response.

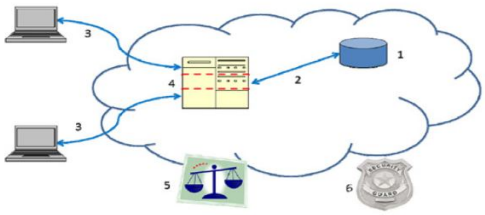


Fig. 2 Areas for security concerns in cloud computing

Limitations of existing system:

Different attack methods include phishing, fraud, and exploitation of software vulnerabilities. Credentials and passwords are often reused that amplifies the impact of various attacks.

Cloud Security Threats: Few security threats that are relevant in cloud computing and are being detected and researched by academia, security organization and both cloud service providers and the cloud customers and are as discussed below:

Side channel attacks: An emerging concern for cloud delivery models using virtualization platforms is the risk of side channel attacks causing data leakage across co-resident virtual machine instances. This risk is evolving, though currently is considered to be in its infancy, as the virtual machine technologies mature. However, it is possible that attackers who fail to compromise endpoints or penetrate cloud infrastructure from outside the cloud perimeter, may consider this technique - acting as a rogue customer within a shared cloud infrastructure to access other customers' data.

Social networking attacks: Business and personal social networking sites are at risk of advanced social engineering attack. Cloud computing systems are targeted due to their large customer data stores. The complex set of relationships between cloud providers, customers, suppliers and vendors means that many employees of these organizations will be listed on social networking sites and be connected to each other. Attackers can setup identities to gain trust, and use online information to determine relationships and roles of staff to prepare their attacks. A combination of technical attack and social engineering attacks can be deployed against a target user by taking advantage of the people they know and the online social network they use.

Increased authentication demands: Instead of purchasing a license and installing software on the client side, users will authenticate in order to be able to use a cloud application. There are some advantages in such a model, such as making software piracy more difficult and making centralized monitoring more convenient. It may also help prevent the spread of sensitive data on untrustworthy clients. This architecture also supports enhanced mobility of users, but demands more robust authentication protocols. Moreover, the movement towards increased hosting of data and applications in the cloud and lesser reliance on specific user machines is likely to increase the threat of phishing and theft of access credentials.

Cloud Computing is the arising technology that delivers software, platform and infrastructure as a service over a network. Recent advances have given rise to the popularity and success of cloud computing. However, when outsourcing the data and business application to a third party causes the security and privacy issues to become a critical concern.

In order to overcome the threat of integrity, user can entrust third party auditor to assess the risk of outsourced data when needed. In our paper we have used cryptographic hash function to verify integrity of data along with key generation mechanism and mutual authentication with TPA (Third Party Auditor). We have separated privacy from security due to its importance and speciality in cloud environments. We have considered the cloud environment as a new computing platform to which the classic methodology of security research can be applied. Thus we employ an attribute-driven methodology to conduct review.

Sample coding to generate key:

```
import java.security.*;
import java.security.spec.*;
import javax.crypto.spec.SecretKeySpec;
class GenKey
{
   GenKey()
   {  }
public static String genkey(String name)
   {
      String key="";
      int n=name.length();
      String result="";
      String nme=name;
      try
      {
           System.out.println("asdad in key "+name+"ssssssa
"+n);
           if(n==8)
          {
         System.out.println("equ8");
         SecretKeySpec sk= new
SecretKeySpec(name.getBytes(), name);
         result=sk.toString();
               String r=result;
               int n1=r.length();
               System.out.println("a1 "+n1+"res "+r);
               if(n1==40){
               r=(String)r.substring(32);
               System.out.println("a2 "+n1+"res "+r);
               key=r;
               }
      }
Sample coding to for server:
class connect extends Thread
{
   Socket soc;
   ObjectOutputStream oos;
   ObjectInputStream ois;
   String req;
   connect(Socket s)
   {
      super();
      soc=s;
      start();
```

ISSN 2278 - 3091
International Journal on Advanced Trends in Computer Science and Engineering (IJATCSE), Vol. 4 No.1, Pages : 19 – 27 (2015)
Special Issue of ICACE 2015 - Held on March 02, 2015,Mysore, India
http://warse.org/pdfs/2015/icace2015sp05.pdf

```
    }
    public void run()
    {
       try
       {
          ois=new ObjectInputStream(soc.getInputStream());
          oos=new
ObjectOutputStream(soc.getOutputStream());
          req=(String)ois.readObject();
          MnuTwo.dft.setValueAt(req,MnuTwo.Index,1);
          if(req.equals("REG"))
          {
             String info=(String)ois.readObject();
             String[] a=info.split("#");
             System.out.println("aaa
"+a[0]+"bbb"+a[1]+"xxx"+a[2]);
             MnuTwo.dft.setValueAt(a[0],MnuTwo.Index,0);
             Connection con=DBConnection.getConnection();
             PreparedStatement
ps=con.prepareStatement("select * from userinfo where
Name=?");
             ps.setString(1,a[0]);
             ResultSet rs=ps.executeQuery();
             if(rs.next())
{
   oos.writeObject("USER_ALREADY_EXISTS");
   MnuTwo.dft.setValueAt("USER_ALREADY_EXISTS",
MnuTwo.Index,2);
}
else
{
   String key=GenKey.genkey(a[0]);
   PreparedStatement ps1=con.prepareStatement("Insert
into userinfo values(?,?,?,?)");
   ps1.setString(1,a[0]);
   ps1.setString(2,a[1]);
   ps1.setString(3,a[2]);
   ps1.setString(4,key);
   int i=ps1.executeUpdate();
   if(i==1)
   {
      new File(Index.drop_box+"/"+a[0].trim()).mkdirs();
      oos.writeObject("SUCCESS "+key);
      MnuTwo.dft.setValueAt(key,MnuTwo.Index,2);
      MnuOne.getUsers();
   }
}
}
if(req.equals("LOGIN"))
{
   String info=(String)ois.readObject();
   String a[] = info.split("#");
   MnuTwo.dft.setValueAt(a[0],MnuTwo.Index,0);
   Connection con=DBConnection.getConnection();
         PreparedStatement
ps=con.prepareStatement("select * from userinfo where
Name=? and Pwd=?");
   ps.setString(1,a[0].trim());
   ps.setString(2,a[1].trim());
   ResultSet rs=ps.executeQuery();
   if(rs.next())
   {
      String file="";
      Connection con1=DBConnection.getConnection();
      PreparedStatement ps1=con1.prepareStatement("Select
Cloud_File from cloud_data where Cloud_User=?");
      ps1.setString(1,a[0].trim());
      ResultSet rs1=ps1.executeQuery();
   while(rs1.next())
      {
         file+=rs1.getString(1)+",";
      }
      if(file.equals(""))
      {
         file="NO_FILE";
      }
      oos.writeObject("SUCCESS
#"+rs.getString(4).trim()+"#"+file);

   MnuTwo.dft.setValueAt("SUCCESS",MnuTwo.Index,2);
   }
   else
   {
      oos.writeObject("FAILURE");

   MnuTwo.dft.setValueAt("FAILURE",MnuTwo.Index,2);
   }
}
if(req.equals("UPLOAD"))
{
   byte[] encData=(byte[])ois.readObject();
   String fName=(String)ois.readObject();
   int sig=(Integer)ois.readObject();
         String Uname=(String)ois.readObject();
   String s=new String(encData);
   String fil="";
   int ii=0;
   float filTotSiz=0;
   float dbSize=0;
   MnuTwo.dft.setValueAt(Uname,MnuTwo.Index,0);
   Connection con=DBConnection.getConnection();
         PreparedStatement ps0=
con.prepareStatement("select Cloud_File from cloud_data
where Cloud_user=?");
   ps0.setString(1,Uname);
   ResultSet rs0=ps0.executeQuery();
   while(rs0.next())
   {
      fil+=rs0.getString(1)+"#";
   }
   String a[]=fil.split("#");
   for(int i=0;i<a.length;i++)
   {
      File f=new File("C://DROP_BOX/"+Uname+"/"+a[i]);
      filTotSiz=filTotSiz+f.length();
      System.out.println("Siz "+filTotSiz);
   }
   float totMb=(filTotSiz/1024)/1024;
   System.out.println("TOTFilMB"+totMb);
         PreparedStatement ps1=
         con.prepareStatement("select Size from userinfo
         where Name=?");
```

ISSN 2278 - 3091
International Journal on Advanced Trends in Computer Science and Engineering (IJATCSE), Vol. 4 No.1, Pages : 19 – 27 (2015)
Special Issue of ICACE 2015 - Held on March 02, 2015,Mysore, India
http://warse.org/pdfs/2015/icace2015sp05.pdf

```
    ps1.setString(1,Uname);
    ResultSet rs1=ps1.executeQuery();
    while(rs1.next())
    {
      dbSize=(float)rs1.getInt(1);
      System.out.println("TOTDBSIZ"+dbSize+" CFLen
"+encData.length);
    }
    float totsiz=((encData.length/1024)/1024)+totMb;
    System.out.println("SIZ "+totsiz);
    if(dbSize>totsiz|dbSize==totsiz)
    {
      PreparedStatement ps=con.prepareStatement("select *
from cloud_data where Cloud_user=? and Cloud_file=?");
      ps.setString(1,Uname);
      ps.setString(2,fName);
      ResultSet rs=ps.executeQuery();
      if(rs.next())
      {
        oos.writeObject("FILE ALREADY EXIST");
        String resp=(String)ois.readObject();
        if(resp.equals("YES"))
        {
        FileOutputStream fos= new
FileOutputStream("C://DROP_BOX/"+Uname+"/"+fName)
;
        fos.write(encData);
        oos.writeObject("SUCCESS");
                        MnuTwo.dft.setValueAt(fName+
                "-SUCCESS",MnuTwo.Index,2);
                }
       else if(resp.equals("NO"))
              {
              String nufName=(String)ois.readObject();
             if(!nufName.equals(""))
       {

 PreparedStatement ps12=con.prepareStatement("Insert into
cloud_data values(?,?,?)");
 ps12.setString(1,Uname);
 ps12.setString(2,nufName);
 ps12.setInt(3,sig);
int i=ps12.executeUpdate();
 if(i==1)
 {
      FileOutputStream fos= new
FileOutputStream("C://DROP_BOX/"+Uname+"/"+nufNa
me);
      fos.write(encData);
      oos.writeObject("SUCCESS");

MnuTwo.dft.setValueAt(nufName+"-SUCCESS",MnuTwo.
Index,2);
 }
 else
 {
         oos.writeObject("UPLOAD FAILED!!! PLEASE
TRY LATER!");
   MnuTwo.dft.setValueAt(fName+"-FAILED",MnuTwo.In
dex,2);
 }
}
else
{
        oos.writeObject("UPLOAD FAILED!!! PLEASE
TRY LATER!");
        MnuTwo.dft.setValueAt(fName+"-FAILED",Mnu
Two.Index,2);
 }
}
else if(resp.equals("CANCEL"))
{
   oos.writeObject("UPLOAD FAILED!!! PLEASE TRY
LATER!");
   MnuTwo.dft.setValueAt(fName+"-FAILED",MnuTwo.In
dex,2);
}
}
else
{
   PreparedStatement ps11=con.prepareStatement("Insert
into cloud_data values(?,?,?)");
   ps11.setString(1,Uname);
   ps11.setString(2,fName);
   ps11.setInt(3,sig);
   int i=ps11.executeUpdate();
   if(i==1)
   {
   FileOutputStream fos= new
FileOutputStream("C://DROP_BOX/"+Uname+"/"+fName)
;
   fos.write(encData);
   oos.writeObject("SUCCESS");
   MnuTwo.dft.setValueAt(fName+"-SUCCESS",MnuTwo.
Index,2);
   }
    }
}
else
{
        oos.writeObject("CLOUD DATA USAGE
EXCEEDED!!!");
   MnuTwo.dft.setValueAt("CLOUD DATA USAGE
EXCEEDED!!!",MnuTwo.Index,2);
}
   MnuOne.getUsers();
}
   else if(req.equals("DOWNLOAD"))
   {
   String fNme=(String)ois.readObject();
   String uNme=(String)ois.readObject();
   byte[] encdata;
   MnuTwo.dft.setValueAt(uNme,MnuTwo.Index,0);
   Connection con=DBConnection.getConnection();
        PreparedStatement
        ps1=con.prepareStatement("Select * from
        cloud_data where Cloud_User=? and
        Cloud_File=?");
   ps1.setString(1,uNme.trim());
   ps1.setString(2,fNme.trim());
   ResultSet i=ps1.executeQuery();
   if(i.next())
```

ISSN 2278 - 3091
International Journal on Advanced Trends in Computer Science and Engineering (IJATCSE), Vol. 4 No.1, Pages : 19 – 27 (2015)
Special Issue of ICACE 2015 - Held on March 02, 2015,Mysore, India
http://warse.org/pdfs/2015/icace2015sp05.pdf

```
        {
          File f=new
File("C://DROP_BOX/"+uNme.trim()+"/"+fNme.trim());
          encdata=new byte[(int)f.length()];
          DataInputStream in= new DataInputStream(new
FileInputStream(f));
          in.read(encdata, 0, (int)f.length());
          in.close();
          int sig=i.getInt(3);
          oos.writeObject("FILE");
          oos.writeObject(encdata);
          oos.writeObject(sig);

   MnuTwo.dft.setValueAt(fNme+"-DOWNLOADED",Mn
uTwo.Index,2);
       }
       else
       {
          oos.writeObject("INVALID FILE NAME!!!");
          MnuTwo.dft.setValueAt(fNme+"-INVALID
FILE",MnuTwo.Index,2);
       }
        }
   }
   catch(Exception e)
      {
         System.out.println(e);
         e.printStackTrace();
      }
   }
}
```

Sample coding for client:

```
import java.io.*;
import javax.swing.*;
import javax.crypto.*;
import javax.crypto.spec.*;
import sun.misc.*;
import java.net.*;
class ClientUpDown
{
   static boolean flg=false;
   static File dFile;
   ClientUpDown()
   {  }
   public static boolean upLoad(File f)
   {
   boolean flg=false;
   int ch=0;
   byte[] data;
   try
   {
      int len=(int)f.length();
      data=new byte[len];
      DataInputStream in= new DataInputStream(new
FileInputStream(f));
      in.read(data);
      in.close();
      int sig=new String(data).hashCode();
      JOptionPane.showMessageDialog(null,"Generated
Signature :"+sig);
      System.out.println("DATA SIG"+sig);
      Cipher cipher= Cipher.getInstance("DES");
      SecretKeySpec spec= new
SecretKeySpec(ClientMnu.Ukey.getBytes(), "DES");
      cipher.init(Cipher.ENCRYPT_MODE, spec);
      data= cipher.doFinal(data);
      System.out.println("ENCDATA "+data);

   ClientUpDown.upLoadEncFile(data,sig,f.getName().trim
(), ClientMnu.Unm);
   }
catch(OutOfMemoryError ee)
{
         JOptionPane.showMessageDialog(null,"Selected
         File Exceded The Buffer Size!!!\n Select File With
         Lesser Size");
}
catch(Exception e)
{  }
return flg;
}
public static void downLoad(String file)
{
         byte[] encdata;
   int ch=0;
   String encStr="";
   String IP="";
   String fil=file;
   int sig=0;
   flg=false;
   try
   {
      FileInputStream fin=new
FileInputStream("ServerIP.txt");
      while((ch=fin.read())!=-1)
                IP+=(char)ch;
      IP.trim();
      Socket s=new Socket(IP,5000);
      ObjectOutputStream oos=new
ObjectOutputStream(s.getOutputStream());
      ObjectInputStream ois=new
ObjectInputStream(s.getInputStream());
      oos.writeObject("DOWNLOAD");
      oos.writeObject(fil);
      oos.writeObject(ClientMnu.Unm);
      String resp=(String)ois.readObject();
      if(resp.equals("FILE"))
      {
            encdata=(byte[])ois.readObject();
        sig=(Integer)ois.readObject();
Cipher cipher= Cipher.getInstance("DES");SecretKeySpec
spec= new SecretKeySpec(ClientMnu.Ukey.getBytes(),
"DES");
cipher.init(Cipher.DECRYPT_MODE, spec);
encdata= cipher.doFinal(encdata);
FileOutputStream fos= new
FileOutputStream(ClientMnu.pth+"/"+file);
fos.write(encdata);
dFile=new File(ClientMnu.pth+"/"+file);
String decdata=new String(encdata);
int recSig=decdata.hashCode();
```

ISSN 2278 - 3091
International Journal on Advanced Trends in Computer Science and Engineering (IJATCSE), Vol. 4 No.1, Pages : 19 – 27 (2015)
Special Issue of ICACE 2015 - Held on March 02, 2015,Mysore, India
http://warse.org/pdfs/2015/icace2015sp05.pdf

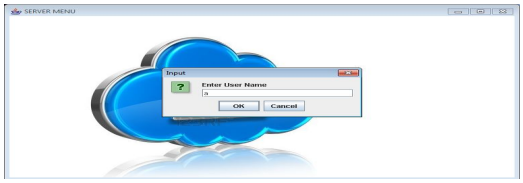
```
ClientMnu.jb3.setText("Verify Sig");
if(recSig==sig)
{
   flg=true;
}
else
{
   flg=true;
}
}
else if((resp.indexOf("INVALID"))!=-1)
{
   JOptionPane.showMessageDialog(null,resp);
}
}
catch(Exception e)
{
}
}
public static void upLoadEncFile(byte[] encData,int
sig,String fName,String Unme)
{
   int ch=0;
   String IP="";
   try
   {
FileInputStream fin=new FileInputStream("ServerIP.txt");
while((ch=fin.read())!=-1)
IP+=(char)ch;
IP.trim();
Socket s=new Socket(IP,5000);
ObjectOutputStream oos=new
ObjectOutputStream(s.getOutputStream());
ObjectInputStream ois=new
ObjectInputStream(s.getInputStream());
oos.writeObject("UPLOAD");
oos.writeObject(encData);
oos.writeObject(fName);
oos.writeObject(sig);
oos.writeObject(Unme);
String msg=(String)ois.readObject();
if(msg.equals("FILE ALREADY EXIST"))
{
   int res=JOptionPane.showConfirmDialog(null,"FILE
ALERDY EXIST          DO WANT TO OVERWRITE!!!");
   if(res==JOptionPane.YES_OPTION)
   {
      oos.writeObject("YES");
   }
   else if(res==JOptionPane.NO_OPTION)
        {
           String
nuFnme=JOptionPane.showInputDialog(null,"Provide New
Name      To File");
      oos.writeObject("NO");
      oos.writeObject(nuFnme);
   }
   else if(res==JOptionPane.CANCEL_OPTION)
    {
      oos.writeObject("CANCEL");
    }
    String msg1=(String)ois.readObject();
     if(!ClientMnu.s.contains(fName.trim()))
     {
   ClientMnu.s.add(fName);
   ClientMnu.jlst.setListData(ClientMnu.s);
          System.out.println("INNNNNNNNNNNNN
VECCCCCCC "+ClientMnu.s);
     }
   System.out.println("INNNNNNNNNNNNN
VECCCCCCC "+ClientMnu.s);
   JOptionPane.showMessageDialog(null,msg1);
}
 else
  {
   if(!ClientMnu.s.contains(fName.trim()))
   {
      ClientMnu.s.add(fName);
      ClientMnu.jlst.setListData(ClientMnu.s);
               System.out.println("INNNNNNNNNNNNN
VECCCCCCC          "+ClientMnu.s);
          }
      JOptionPane.showMessageDialog(null,msg);
      }
    }
  catch(Exception e)
}
```

RESULTS AND DISCUSSIONS

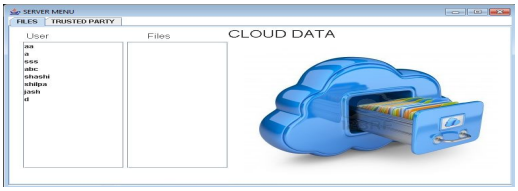
This section describes the security and privacy issues considered on both server and client side. The implementation is done using JAVA.

SERVER SIDE:

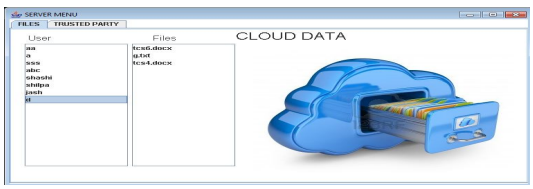
The cloud server has its own unique user name and password. Server must run before starting the client part. First enter the user name of the server.



Server menu has two tabs FILES and TRUSTED PARTY.

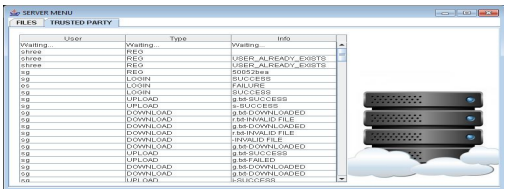


Files tab lists show the registered user's name and when we click on particular user in the file list it will display all the files that are already UPLODED on the cloud server.



ISSN 2278 - 3091
International Journal on Advanced Trends in Computer Science and Engineering (IJATCSE), Vol. 4 No.1, Pages : 19 – 27 (2015)
Special Issue of ICACE 2015 - Held on March 02, 2015,Mysore, India
http://warse.org/pdfs/2015/icace2015sp05.pdf

Trusted party tab maintains the clients' status. It displays which user is currently logged in, what type (Register ,Upload, Download and Editing) of action the client is performing and display the message "SUCCESS" when all the activities are taken place properly otherwise it shows "FAILURE" message.



CLINT SIDE:

In Registration part a new client can register in server by giving its user name, password, re-enter password and memory required in cloud. When the client press the register button, if the password and re-entered password is same then the user will be register successfully, otherwise it will show the error message. The cloud server generates a secret key for client and sent back to it. This secret key can be used by client to encrypt or decrypt files.

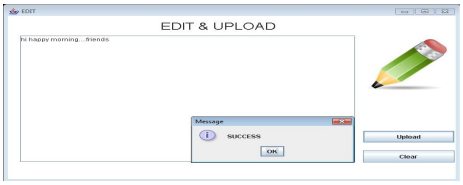
Once the client is successfully logged in then he can perform Download, Upload, and editing operation. The client can upload a file to server. Client can browse the file and upload to the server. Client can upload only one file at a time. Before uploading file, client encrypts the content and generates signature for encrypted content. Then the file is uploaded in to the server. The success message will be displayed after uploading the file successfully.

Clients can download a file from the server. When the file is downloaded, the server sends .the signature along with the file contents. When file downloaded from server, the client finds signature for the file and matches with the old signature.

If both signatures are matching, this means the file is in genuine condition and display the message 'content not modified'. If signatures don't match, then the file is modified at server.

If the file name is not there in the File Uploaded list, it display the error message "Invalid Filename", if the client try to download the file which is uploaded before. The client can download an existing file and edit that file. Again it can be saved in to server. If the file name is not there in the File Uploaded list, it display the error message "Invalid Filename", if the client try to download the file which is uploaded before. Editing the file successfully, signature is generated for editing file. Then the file is uploaded in to the server.

If the file name is same it will display the confirmation message "File Already Exist Do You Want to Overwrite the file name or not message" to client. The success message will be displayed after edit and re-uploading the file successfully.



CONCLUSION and FUTURE WORK

Data security problems exist in cloud data storage that is essentially a distributed storage system. In order to achieve the assurances of cloud data integrity and availability to enforce the quality of dependable cloud storage service for users, a privacy-preserving system for data storage security in cloud computing is proposed in our work. A cloud-based storage scheme which supports outsourcing of dynamic data. The proposed scheme enables the authorized users to ensure that they are receiving the secured data. Our work deals with PC to PC communication. Advertisement services are widely accepted among application developers. However it is still important to investigate the behavior of an application with regards to security and privacy. Many Android applications require permissions for sensitive information access and network features, and that among them are applications that connect to many outside servers without the user's acknowledgment. In future security and privacy issues can be dealt by using PC to Android phones.

ACKNOWLEDGEMENT

Dr. Nandini Prasad K S and Chetana Srinivas would like to thank Anitha, Deepa, Bindu and Shashikala for their work and support.

REFERENCES

[1] Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li, "Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing," IEEE Trans. Parallel and Distributed Systems, vol. 22, no. 5, pp. 847-859, 2011.

[2] C. Wang, K. Ren, W. Lou, and J. Li, "Towards Publicly Auditable Secure Cloud Data Storage Services," IEEE Network Magazine, vol. 24, no. 4, pp. 19-24, July/Aug. 2010.

[3] A. Juels and B.S. Kaliski Jr., "PORs: Proofs of Retrievability for Large Files," Proc. 14th ACM Conf. Computer and Comm. Security (CCS '07), pp. 584-597, Oct. 2007.

[4] K. Ren, C. Wang, and Q. Wang, "Security Challenges for the Public Cloud," IEEE Internet Computing, vol. 16, no. 1, pp. 69-73, 2012.

[5] G . Ateniese, R.D. Pietro, L.V. Mancini, and G. Tsudik, "Scalable and Efficient Provable Data Possession," Proc. Fourth Int'l Conf. Security Privacy Comm. Networks, pp. 1-10, 2008.

[6] K. D. Bowers, A. Juels, and A. Oprea, "HAIL: A High- Availability and Integrity Layer for Cloud Storage," Cryptology ePrint Archive, Report 2008/489, 2008, http://eprint.iacr.org/.

[7] K.D. Bowers, A. Juels, and A. Oprea, "Proofs of Retrievability: Theory and Implementation," Proc. ACM Workshop Cloud Computing Security (CCSW '09), pp. 43-54, 2009.

[8] Cloud Security Alliance, "Security Guidance for Critical Areas of Focus in Cloud Computing," http://www.cloudsecurityalliance. org, 2009.