BAB IV

IMPLEMENTASI DAN PENGUJIAN SISTEM

Tahap implementasi dilaksanakan setelah tahap perancangan sistem. Tahap implementasi merupakan tahap meletakkan sistem upaya siap untuk dioperasikan dan dapat dipandang sebagai usaha untuk mewujudkan sistem yang telah dirancang.

4. 1. Implementasi Aplikasi

Subbab ini membahas mengenai lingkungan implementasi dan implementasi kelas dalam pengerjaan tugas akhir ini.

4. 1. 1. Lingkungan Implementasi

Perangkat lunak *digital signature* ini dilakukan dalam lingkup perangkat keras dan perangkat lunak. Spesifikasi perangkat lunak yang digunakan adalah :

Tabel 4. 1. Perangkat Lunak Yang Digunakan

No	Perangkat Lunak	Keterangan
1.	Sistem Operasi	Windows 7
2.	Bahasa Pemrograman	Java
3.	Perangkat pendukung	Java Development Kit

4.	Aplikasi Tool	NetBeans IDE 6.8
5.	UML Modeler	Rasional Rose 2000

4. 1. 2. Perangkat Keras Yang Digunakan

Perangkat Keras yang digunakan dalam membangun aplikasi *digital signature* ini adalah :

Tabel 4. 2. Perangkat Keras Yang Digunakan

Perangkat	Spesifikasi Komputer
Processor	420 Ghz
RAM	512 Mb
Hardisk	40 Gb
Kabel	UTP
Konektor	RJ45
penghubung	Switch

4. 1. 3. Implementasi kelas

4.1.3.1. Kelas Form

Nama method : Form() Input : - Output : -

initComponents();
 	
switchPanelUtama(panelHome);
 	
key=Key.loadFromFile("key.conf");
messageHandler=new MessageHandler(this);
messageHandler.start();
loadData();
refreshInbox();
refreshSent();
refreshContact();

Nama method : loadData() Input : - Output : -

inboxList.clear();
sentList.clear();
contactList.clear();

```
      BufferedReader in=null;
      String line;
      try
      {
        in= new BufferedReader(new FileReader("inbox.txt"));
        while((line=in.readLine())!=null)
        {
          String data[]=line.split("#");
          inboxList.add(newMessage(data[0],
data[1],data[2],Boolean.parseBoolean(data[3]),Long.parseLong(data[4])));
        }
      }catch(Exception e){}


      try
      {
        in= new BufferedReader(new FileReader("sent.txt"));

        while((line=in.readLine())!=null)
        {
          String data[]=line.split("#");
          sentList.add(newMessage(data[0],
data[1],data[2],Boolean.parseBoolean(data[3]),Long.parseLong(data[4])));
        }
      }catch(Exception e){}

      try
```

```
    {
      in= new BufferedReader(new FileReader("contact.txt"));

      while((line=in.readLine())!=null)
      {
        String data[]=line.split("#");
        contactList.add(new Contact(data[0], data[1]));
      }
    }
    catch(Exception e){}
```

Nama method : saveInbox()

Input : -

Output : -

```
      try
      {
        FileWriter fw=new FileWriter("inbox.txt");

        for(int i=0;i<inboxList.size();i++)
        {
          Message msg=inboxList.get(i);

fw.write(msg.ip+"#"+msg.fileName+"#"+msg.signature+"#"+msg.read+
"#"+msg.time+"\n");
        }
```

```
        fw.flush();
      }
      catch(Exception e){}
```

Nama method : saveSent()

Input : -

Output : -

```
      try
      {
        FileWriter fw=new FileWriter("sent.txt");

        for(int i=0;i<sentList.size();i++)
        {
          Message msg=sentList.get(i);

fw.write(msg.ip+"#"+msg.fileName+"#"+msg.signature+"#"+msg.read+
"#"+msg.time+"\n");
        }

        fw.flush();
      }
      catch(Exception e){}
```

Nama method : saveContact()

Input : -

Output : -

```
      try
              {
                  FileWriter fw=new FileWriter("contact.txt");

                  for(int i=0;i<contactList.size();i++)
                  {
                      Contact c=contactList.get(i);
                      fw.write(c.name+"#"+c.ip+"\n");
                  }

                  fw.flush();
              }
              catch(Exception e){}
```

Nama method : refreshInbox()

Input : -

Output : -

```
      DefaultTableModel tableModel= (DefaultTableModel)
tabelInbox.getModel();

              //hapus semua table
              int size=tableModel.getRowCount();
              for(int i=0;i<size;i++)
```

```
                {
                    tableModel.removeRow(0);
                }

                //bikin data
                Object[][] data=new Object[inboxList.size()][4];
                for(int i=0;i<data.length;i++)
                {
                    Message msg=inboxList.get(i);
                    try{
                        data[i][0]=msg.read?null:              new
ImageIcon("img/seruMerah.png");
                    }catch(Exception e){e.printStackTrace();}
                    data[i][1]=getNameContact(msg.ip);
                    data[i][2]=msg.fileName;
                    data[i][3]=String.format("%1$td-%1$tm-%1$tY
%1$tH:%1$tM",msg.time);
                }

                //masukkan data
                for(int i=0;i<data.length;i++)
                {
                    tableModel.addRow(data[i]);
                }
                tabelInbox.setModel(tableModel);
```

Nama method : refreshSent()

Input : -

Output : -

```
      DefaultTableModel      tableModel=      (DefaultTableModel)
tabelSent.getModel();


               //hapus semua table
               int size=tableModel.getRowCount();
               for(int i=0;i<size;i++)
               {
                   tableModel.removeRow(0);
               }


               //bikin data
               String[][] data=new String[sentList.size()][3];
               for(int i=0;i<data.length;i++)
               {
                   Message msg=sentList.get(i);
                   data[i][0]=getNameContact(msg.ip);
                   data[i][1]=msg.fileName;
                   data[i][2]=String.format("%1$td-%1$tm-%1$tY
%1$tH:%1$tM",msg.time);
               }


               //masukkan data
               for(int i=0;i<data.length;i++)
               {
```

```
                tableModel.addRow(data[i]);
            }
            tabelSent.setModel(tableModel);
```

Nama method : refreshContact()

Input : -

Output : -

```
    DefaultTableModel tableModel= (DefaultTableModel)
tabelContact.getModel();

            //hapus semua table
            int size=tableModel.getRowCount();
            for(int i=0;i<size;i++)
            {
                tableModel.removeRow(0);
            }

            //bikin data
            String[][] data=new String[contactList.size()][2];
            for(int i=0;i<data.length;i++)
            {
                Contact c=contactList.get(i);
                data[i][0]=c.name;
                data[i][1]=c.ip;
            }
```

```
                //masukkan data
                for(int i=0;i<data.length;i++)
                {
                    tableModel.addRow(data[i]);
                }
                tabelContact.setModel(tableModel);
```

```
Nama method : getNameContact()
Input       : -
Output       : -
```

```
      for(int i=0;i<contactList.size();i++)
                {
                    if(contactList.get(i).ip.equalsIgnoreCase(ip))
                        return contactList.get(i).name;
                }
                return ip;
```

```
Nama method : getIPContact()
Input       : -
Output       : -
```

```
      for(int i=0;i<contactList.size();i++)
                {

if(contactList.get(i).name.equalsIgnoreCase(name))
```

```
                    return contactList.get(i).ip;
            }
            return name;
```

Nama method : contactComposeCBActionPerformed()

Input : -

Output : -

```
      int index=contactComposeCB.getSelectedIndex();
            if(index>-1)
            {
                if(index<contactList.size())
                {

ipComposeTF.setText(contactList.get(index).ip);
                    ipComposeTF.setEnabled(false);
                }
                else {
                    ipComposeTF.setText("");
                    ipComposeTF.setEnabled(true);
                }
            }
```

Nama method : browseComposeBActionPerformed()

Input : -

Output : -

```
      JFileChooser jfc=new JFileChooser();
              int result=jfc.showOpenDialog(this);
              if(result==JFileChooser.APPROVE_OPTION)
              {
                  File fileTemp=jfc.getSelectedFile();
                  String
fileName=fileTemp.getName().toLowerCase();
                  if(fileName.endsWith(".doc")                  ||
fileName.endsWith(".txt"))
                  {
                      long size=fileTemp.length();
                      if(size<26214400)//lebih kecil dari 25 MB

fileComposeTF.setText(jfc.getSelectedFile().toString());
                      else
JOptionPane.showMessageDialog(this,"File  tidak  boleh  melebihi
25MB");
                  }
                  else
JOptionPane.showMessageDialog(this,"Aplikasi   hanya   mendukung
.txt dan .doc");
              }
```

```
Nama method : generateComposeBActionPerformed()
Input       : -
Output       : -
```

```
      if(fileComposeTF.getText().equals(""))
              {
                  JOptionPane.showMessageDialog(this,"Pilih file
pesan");
                  return;
              }

              Key keyTemp=null;

if((keyTemp=Key.parsePrivateKey(privateKeyComposeTF.getText()))=
=null)
              {
                  JOptionPane.showMessageDialog(this,"Private
key tidak valid");
                  return;
              }

              try
              {
                  //read file
                  File file=new File(fileComposeTF.getText());

                  byte fileArr[]=new byte[(int)file.length()];
                  new FileInputStream(file).read(fileArr);

                  String
signature=Signature.createSignature(fileArr, keyTemp);
```

```
                signatureComposeTF.setText(signature);
            }
            catch(Exception e)
            {
                JOptionPane.showMessageDialog(this,"Gagal
membuat signature, "+e);
                e.printStackTrace();
                return;
            }
```

Nama method : resetComposeBActionPerformed()

Input : -

Output : -

```
      ipComposeTF.setText("");
            fileComposeTF.setText("");

            contactComposeCB.removeAllItems();
            for(int i=0;i<contactList.size();i++)
            {

contactComposeCB.addItem(contactList.get(i).name);
            }
            contactComposeCB.addItem("-Masukkan IP-");

            signatureComposeTF.setText("");
```

```
            privateKeyComposeTF.setText(key.getPrivateKey());
```

Nama method : sendComposeBActionPerformed()

Input : -

Output : -

```
      if((contactList.size()==0                                    &&
ipComposeTF.getText().equals("")) ||

(contactComposeCB.getSelectedIndex()==contactList.size()      &&
ipComposeTF.getText().equals("")))
            {
                JOptionPane.showMessageDialog(this,"Masukkan
ip tujuan atau pilih contact");
                return;
            }

            if(signatureComposeTF.getText().equals(""))
            {
                JOptionPane.showMessageDialog(this,"Generate
dahulu signature file");
                return;
            }


            if(fileComposeTF.getText().equals(""))
            {
```

```
                    JOptionPane.showMessageDialog(this,"Pilih file
pesan");
                    return;
                }

                try
                {
                    //read file
                    File file=new File(fileComposeTF.getText());
                    byte fileArr[]=new byte[(int)file.length()];
                    new FileInputStream(file).read(fileArr);

                    String destination=ipComposeTF.getText();

if(contactComposeCB.getSelectedIndex()<contactList.size())destin
ation=getIPContact(""+contactComposeCB.getSelectedItem());



messageHandler.sendMessage(destination,file.getName(),fileArr,si
gnatureComposeTF.getText());

                    JOptionPane.showMessageDialog(this,"Sukses
kirim pesan");

                    sentList.add(new
Message(destination,fileComposeTF.getText(),signatureComposeTF.g
etText(),true,System.currentTimeMillis()));
                    saveSent();
```

```
                refreshSent();
            }
            catch(Exception e)
            {
                JOptionPane.showMessageDialog(this,"Gagal
mengirim pesan, "+e);
                e.printStackTrace();
                return;
            }
```

Nama method : deleteInboxBActionPerformed()

Input : -

Output : -

```
      if(tabelInbox.getSelectedRowCount()==0) {
                JOptionPane.showMessageDialog(this,"Pilih
dahulu message yang akan dihapus");
                return;
            }
            if(tabelInbox.getSelectedRowCount()>1) {
                JOptionPane.showMessageDialog(this,"Pilih
hanya satu message");
                return;
            }

            int index=tabelInbox.getSelectedRow();
            if(JOptionPane.showConfirmDialog(this,"Apakah anda
yakin?","",JOptionPane.YES_NO_OPTION)==JOptionPane.OK_OPTION)
```

```
                {
                    inboxList.remove(index);
                    saveInbox();
                    refreshInbox();
                }
```

```
Nama method : tabelInboxMouseClicked()
Input       : -
Output       : -
```

```
      if(evt.getClickCount()==2) {

                    int index=tabelInbox.getSelectedRow();
                    if(index!=-1) {
                        Message msg=inboxList.get(index);
                        tempMessage=msg;
                        msg.read=true;
                        replyViewB.setEnabled(true);
                        showViewMessage(msg.ip,
msg.fileName,msg.signature);
                        refreshInbox();
                        saveInbox();
                    }
                }
```

```
Nama method : deleteSentBActionPerformed()
Input       : -
```

Output : -

```
      if(tabelSent.getSelectedRowCount()==0) {
                  JOptionPane.showMessageDialog(this,"Pilih
dahulu message yang akan dihapus");
                  return;
              }
              if(tabelSent.getSelectedRowCount()>1) {
                  JOptionPane.showMessageDialog(this,"Pilih
hanya satu message");
                  return;
              }

              int index=tabelSent.getSelectedRow();
              if(JOptionPane.showConfirmDialog(this,"Apakah anda
yakin?","",JOptionPane.YES_NO_OPTION)==JOptionPane.OK_OPTION)
              {
                  sentList.remove(index);
                  saveSent();
                  refreshSent();
              }
```

Nama method : tabelSentMouseClicked()

Input : -

Output : -

```
      if(evt.getClickCount()==2) {
                  int index=tabelSent.getSelectedRow();
```

```
                    if(index!=-1) {
                        tempMessage=null;
                        Message msg=sentList.get(index);
                        replyViewB.setEnabled(false);
                        showViewMessage(msg.ip,
msg.fileName,msg.signature);
                    }
                }
```

Nama method : saveContactViewBActionPerformed() Input : - Output : -

```
      String
nama=JOptionPane.showInputDialog(viewMessageDialog,"Masukkan
nama contact");
               if(nama!=null && !nama.equals(""))
               {
                   Contact c=new Contact();
                   c.ip=fromViewTF.getText();
                   c.name=nama;
                   contactList.add(c);
                   saveContactViewB.setEnabled(false);
                   saveContact();
                   refreshContact();

JOptionPane.showMessageDialog(viewMessageDialog,"Sukses   tambah
contact");
```

```
            }
```

Nama method : openViewBActionPerformed()

Input : -

Output : -

```
      if(tempMessage==null)//buka file sent
              {
                  try{
                      openFile(new File(fileViewTF.getText()));
                  }
                  catch(Exception e){

JOptionPane.showMessageDialog(viewMessageDialog,"Gagal   membuka
file "+e);
                  }
                  return;
              }

              byte[] fileArr=null;
              File file=null;
              String fileName=""+tempMessage.time;

              try
              {
                  //read file
                  file=new File(fileName);
                  fileArr=new byte[(int)file.length()];
```

```
                    new FileInputStream(file).read(fileArr);
                }
                catch(Exception e)
                {

JOptionPane.showMessageDialog(viewMessageDialog,"Gagal   membaca
file, "+e);
                    e.printStackTrace();
                    return;
                }

                Key
keyTemp=Key.parsePublicKey(publicKeyViewTF.getText());
                if(keyTemp==null)
                {

JOptionPane.showMessageDialog(viewMessageDialog,"Public      key
tidak valid");
                    return;
                }

                try
                {
                    boolean
sukses=Signature.checkSignature(fileArr,                keyTemp,
signatureViewTF.getText());
                    if(sukses)
                    {
```

```
                        File to=new File(tempMessage.fileName);

                        if(!to.exists())//copy file
                        {

copyFile(file.toString(),to.toString());
                        }

                        openFile(to);
                    }
                    else
JOptionPane.showMessageDialog(viewMessageDialog,"Signature tidak
valid");
                }
                catch(Exception e)
                {

JOptionPane.showMessageDialog(viewMessageDialog,"Gagal  mengecek
signature "+e);
                    return;
                }
```

Nama method : replyViewBActionPerformed() Input : - Output : -
viewMessageDialog.setVisible(false); switchPanelUtama(panelCompose);

```
                String to=fromViewTF.getText();


if(to.contains("<"))to=to.substring(0,to.indexOf("<"));
                ipComposeTF.setText(to);


contactComposeCB.setSelectedIndex(contactList.size());
```

Nama method : addContactBActionPerformed() Input : - Output : -

namaContactTF.setText("");
ipContactTF.setText("");
addContactDialogB.setText("Add");
addContactDialog.pack();
addContactDialog.setVisible(true);

Nama method : deleteContactBActionPerformed() Input : - Output : -

if(tabelContact.getSelectedRowCount()==0) {
JOptionPane.showMessageDialog(this,"Pilih dahulu contact yang akan dihapus");
return;
}
if(tabelContact.getSelectedRowCount()>1) {
JOptionPane.showMessageDialog(this,"Pilih

```
hanya satu contact");
                    return;
                }

                int index=tabelContact.getSelectedRow();
                if(JOptionPane.showConfirmDialog(this,"Apakah anda
yakin?","",JOptionPane.YES_NO_OPTION)==JOptionPane.OK_OPTION)
                {
                    contactList.remove(index);
                    saveContact();
                    refreshContact();
                }
```

```
Nama method : editContactBActionPerformed()
Input       : -
Output      : -
```

```
      if(tabelContact.getSelectedRowCount()==0) {
                    JOptionPane.showMessageDialog(this,"Pilih
dahulu contact yang akan diedit");
                    return;
                }
                if(tabelContact.getSelectedRowCount()>1) {
                    JOptionPane.showMessageDialog(this,"Pilih
hanya satu contact");
                    return;
                }
```

```
                int index=tabelContact.getSelectedRow();
                contactTemp=contactList.get(index);
                namaContactTF.setText(contactTemp.name);
                ipContactTF.setText(contactTemp.ip);
                addContactDialogB.setText("Edit");
                addContactDialog.pack();
                addContactDialog.setVisible(true);
```

```
Nama method : addContactDialogBActionPerformed()
Input       : -
Output      : -
```

```
      String ip=ipContactTF.getText();
                if(ip==null || ip.equals("")) {

JOptionPane.showMessageDialog(addContactDialog,addContactDialogB
.getText()+" contact gagal\nIP harus dimasukkan");
                    return;
                }
                String name=namaContactTF.getText();
                if(name.equals("")                                  ||
getIPContact(name).equalsIgnoreCase(ip)) {

JOptionPane.showMessageDialog(addContactDialog,addContactDialogB
.getText()+"  contact  gagal\nNama  harus  dimasukkan/Nama  Telah
ada");
                    return;
                }
```

```
                Contact c=new Contact();


                c.ip=ip;
                c.name=name;


if(addContactDialogB.getText().equals("Add"))contactList.add(0,c
);
                else //edit
                {
                    int index=contactList.indexOf(contactTemp);
                    contactList.set(index, c);
                }
                saveContact();
                refreshContact();


                addContactDialog.setVisible(false);
```

```
Nama method : addContactDialogBActionPerformed()
Input       : -
Output       : -
```

```
      String ip=ipContactTF.getText();
                if(ip==null || ip.equals("")) {


JOptionPane.showMessageDialog(addContactDialog,addContactDialogB
.getText()+" contact gagal\nIP harus dimasukkan");
                    return;
                }
```

```
                String name=namaContactTF.getText();
                if(name.equals("")                                  ||
getIPContact(name).equalsIgnoreCase(ip)) {

JOptionPane.showMessageDialog(addContactDialog,addContactDialogB
.getText()+"  contact  gagal\nNama  harus  dimasukkan/Nama  Telah
ada");
                    return;
                }
                Contact c=new Contact();

                c.ip=ip;
                c.name=name;

if(addContactDialogB.getText().equals("Add"))contactList.add(0,c
);
                else //edit
                {
                    int index=contactList.indexOf(contactTemp);
                    contactList.set(index, c);
                }
                saveContact();
                refreshContact();

                addContactDialog.setVisible(false);
```

Nama method : generateSettingBActionPerformed() Input : - Output : -

Key k=Signature.generateKey();
privateSettingTF.setText(k.s+"-"+k.p+"-"+k.q+"-"+k.a);
publicSettingTF.setText(k.lamdha+"-"+k.p+"-"+k.q+"-"+k.a);

Nama method : saveSettingBActionPerformed() Input : - Output : -

Key privateK=Key.parsePrivateKey(privateSettingTF.getText());
if(privateK==null) {
JOptionPane.showMessageDialog(settingDialog,"Private key tidak valid");
return;
}
Key publicK=Key.parsePublicKey(publicSettingTF.getText());
if(publicK==null) {
JOptionPane.showMessageDialog(settingDialog,"Public key tidak valid");

```
                    return;
                }

                privateK.lamdha=publicK.lamdha;
                key=privateK;

                try {
                    Key.saveToFile(KEY_FILE, key);
                } catch(Exception e){}

                settingDialog.setVisible(false);
```

Nama method : broadcastSettingBActionPerformed() Input : - Output : -

```
      try {
                    messageHandler.sendKey( key);

JOptionPane.showMessageDialog(settingDialog,"Sukses    broadcast
key");
                } catch (Exception ex) {

JOptionPane.showMessageDialog(settingDialog,"Gagal broadcast key
: "+ex.getMessage());
                    ex.printStackTrace();
                }
```

Nama method : settingButtonActionPerformed() Input : - Output : -

try
{
privateSettingTF.setText(key.getPrivateKey());
publicSettingTF.setText(key.getPublicKey());
}
catch(Exception e){}
 	
settingDialog.pack();
settingDialog.setVisible(true);

Nama method : showViewMessage() Input : - Output : -

publicKeyViewTF.setText(key.getPublicKey());
String from=getNameContact(ip);
String alamat=ip;
if(!from.equalsIgnoreCase(ip)){
saveContactViewB.setEnabled(false);
alamat+="<"+from+">";
}
else saveContactViewB.setEnabled(true);
fromViewTF.setText(alamat);
 	
signatureViewTF.setText(signature);

```
                fileViewTF.setText(file.toString());


                viewMessageDialog.pack();
                viewMessageDialog.setVisible(true);
```

```
Nama method : receiveMessage()
Input       : -
Output       : -
```

```
      try
                {
                    long now=System.currentTimeMillis();
                    //tulis file
                    new FileOutputStream(""+now).write(filearr);


                    inboxList.add(0,new
Message(ip,namaFile,signature,false,now));
                    saveInbox();
                    refreshInbox();
                }catch(Exception e)
                {}
```

```
Nama method : openFile()
Input       : -
Output       : -
```

```
      try {
                    Runtime.getRuntime().exec("rundll32
```

```
SHELL32.DLL,ShellExec_RunDLL " + file.getAbsolutePath());
              } catch (Exception ex) {

JOptionPane.showMessageDialog(viewMessageDialog,"Gagal      buka
file, "+ex);
                  ex.printStackTrace();
              }
```

```
Nama method : copyFile()
Input       : -
Output      : -
```

```
      FileWriter writer=new FileWriter(to);
              FileReader reader=new FileReader(file);

              int kar;

              while((kar=reader.read())!=-1)
              {
                  writer.write(kar);
              }

              writer.flush();
              writer.close();
              reader.close();
```

4.1.3.2. Kelas key

Nama method : key() Input : - Output : -

this.p=p;
this.q=q;
this.a=a;
this.s=s;
this.lamdha=lamdha;

Nama method : loadFromFile() Input : String file Output : -

try
{
Properties props=new Properties();
props.load(new FileReader(file));
 	
Key privateK=parsePrivateKey(props.getProperty("private_key"));
Key publicK=parsePublicKey(props.getProperty("public_key"));
 	
privateK.lamdha=publicK.lamdha;

```
                return privateK;
            }
            catch(Exception e){}
            return null;
```

Nama method : saveToFile() Input : String file,Key key Output : -

```
      try
            {
                FileWriter fw=new FileWriter(file);
                fw.write("private_key  =  "+key.s+"-"+key.p+"-
"+key.q+"-"+key.a+"\n");
                fw.write("public_key     =     "+key.lamdha+"-
"+key.p+"-"+key.q+"-"+key.a);

                fw.flush();
                return true;
            }
            catch(Exception e){}
            return false;
```

Nama method : getPrivateKey() Input : - Output : -

```
return s+"-"+p+"-"+q+"-"+a;
```

```
Nama method : getPublicKey()
Input       : -
Output       : -
```

```
return lamdha+"-"+p+"-"+q+"-"+a;
```

```
Nama method : parsePrivateKey()
Input       : -
Output       : -
```

```
  try
        {
            String[] strTemp=str.split("-");
            if(strTemp.length!=4)return null;

            Key key=new Key();

            //cari s
            key.s=new BigInteger(strTemp[0]);

            //cari p
            key.p=new BigInteger(strTemp[1]);
```

```
                //cari q
                key.q=new BigInteger(strTemp[2]);

                //cari a
                key.a=new BigInteger(strTemp[3]);

                return key;
            }
            catch(Exception e){}
            return null;
```

Nama method : parsePublicKey()

Input : -

Output : -

```
try
        {
            String[] strTemp=str.split("-");
            if(strTemp.length!=4)return null;

            Key key=new Key();

            //cari lamdha
            key.lamdha=new BigInteger(strTemp[0]);

            //cari p
            key.p=new BigInteger(strTemp[1]);
```

```
                //cari q
                key.q=new BigInteger(strTemp[2]);

                //cari a
                key.a=new BigInteger(strTemp[3]);

                return key;
            }
            catch(Exception e){}
            return null;
```

4.1.3.3. Kelas Signature

```
Nama method : generateKey()
Input       : -
Output       : -
```

```
Random rand=new Random();

            BigInteger p=new BigInteger(BITNUM,0, rand);
            int i=0;
            while(true)
            {
                if(p.isProbablePrime(BITNUM))break;
                else p=p.nextProbablePrime();
                i++;
```

```
                if(i>1000)return generateKey();
            }

            //find the prime factor of p-1
            BigInteger pTemp=p.subtract(new BigInteger("1"));
            BigInteger q=new BigInteger("9");
            i=0;
            while(true)
            {
                if(pTemp.mod(q).intValue()==0)break;
                else q=q.nextProbablePrime();
                i++;
                if(i>1000)return generateKey();
            }

            BigInteger a=new BigInteger(BITNUM, rand);
            //cari a^q mod p == 1
            i=0;
            while(true)
            {

if(a.pow(q.intValue()).mod(p).intValue()==1)break;
                else
                {
                    a=a.subtract(new BigInteger("1"));
                    if(a.intValue()<=1                    ||
a.intValue()>=p.intValue())a=new BigInteger(BITNUM, rand);
                }
```

```
                i++;
                if(i>10000)return generateKey();
            }

            BigInteger s=new BigInteger(BITNUM, rand);
            //cari s<q
            i=0;
            while(true)
            {
                if(s.intValue()!=0 && s.compareTo(q)<0)break;
                else s=new BigInteger(BITNUM, rand);
                i++;
                if(i>10000)return generateKey();
            }

            BigInteger
lamdha=a.modInverse(p).pow(s.intValue()).mod(p);

            return new Key(p,q,a,s,lamdha);
```

Nama method : createSignature() Input : - Output : -
Random rand=new Random(); //buat r<q

```
                BigInteger r=new BigInteger(BITNUM, rand);


                int i=0;
                while(true)
                {
                    if(i>100000)throw  new  Exception("Private  key
tidak valid");
                    if(r.intValue()!=0                              &&
r.compareTo(key.q)<0)break;
                    else r=new BigInteger(BITNUM, rand);
                    i++;
                }


                 //buat x
                BigInteger x=key.a.pow(r.intValue()).mod(key.p);
                byte[] bytex=x.toByteArray();



                //bacaFile
                byte[] fileArr=new byte[file.length+bytex.length];


                //M||X
                System.arraycopy(file,0,fileArr,0,file.length);


System.arraycopy(bytex,0,fileArr,file.length,bytex.length);


                //hitung digest dari M||X
                MessageDigest   md=MessageDigest.getInstance("SHA-
```

```
256");
              byte[] resultdigest=md.digest(fileArr);


              BigInteger h=new BigInteger(resultdigest);
              String kembali=h.toString();
              h=h.mod(key.q);


              //cari y
              BigInteger sh=key.s.multiply(h);
              BigInteger y=r.add(sh).mod(key.q);
              kembali+=";"+y;


              return kembali;
```

```
      Nama method : checkSignature()
      Input       : -
      Output       : -
```

```
      String tempstr[]=signature.split(";");
              BigInteger                                    h=new
BigInteger(tempstr[0]).mod(key.q);


              BigInteger y=new BigInteger(tempstr[1]);
              BigInteger z=key.a.pow(y.intValue());
              BigInteger ztemp=key.lamdha.pow(h.intValue());
              z=z.multiply(ztemp).mod(key.p);
```

```
                byte bytez[]=z.toByteArray();


                //bacaFile
                System.out.println(file.length);
                byte[] fileArr=new byte[file.length+bytez.length];


                System.arraycopy(file,0,fileArr,0,file.length);


System.arraycopy(bytez,0,fileArr,file.length,bytez.length);


                //hitung digest
                MessageDigest   md=MessageDigest.getInstance("SHA-
256");
                byte[] resultdigest=md.digest(fileArr);


                BigInteger h1=new BigInteger(resultdigest);
                h1=h1.mod(key.q);


                System.out.println(h1+" "+h);


                return h.intValue()==h1.intValue();
```

4.1.3.4. Kelas Contact

```
Nama method : Contact()
Input       : -
Output       : -
```

```
name=n;
ip=i;
```

4.1.3.5. Kelas Message

Nama method : Message() Input : - Output : -

this.ip=ip;
this.fileName=file;
this.signature=signature;
this.read=read;
this.time=time;

4.1.3.6. Kelas MessageHandler

Nama method : MessageHandler() Input : - Output : -
handler=form;

```
serverSocket=new ServerSocket(PORT);
```

```
Nama method : Run()
Input       : -
Output      : -
```

```
while(true)
    {
      try
      {
        Socket s=serverSocket.accept();
        DataInputStream in=new DataInputStream(s.getInputStream());

        byte kode=in.readByte();
        if(kode==0)
        {
          String namaFile=in.readLine();
          int length=Integer.parseInt(in.readLine());
          byte file[]=new byte[length];
```

```
            int readedTotal=0;
            int read=0;

            while(true)
            {
               read=in.read(file,readedTotal,length-readedTotal);
               readedTotal+=read;
               if(readedTotal>=length)break;
            }

            String signature=in.readLine();

            //panggil handler

        handler.receiveMessage(s.getInetAddress().getHostAddress().toString(),
        namaFile,file,signature);
        }
        else if(kode==1)//update key
        {
```

```
            String temp=in.readLine();
            Key privateKey=Key.parsePrivateKey(temp);


            temp=in.readLine();
            Key publicKey=Key.parsePublicKey(temp);


            privateKey.lamdha=publicKey.lamdha;


            handler.setKey(privateKey);
        }
        //s.close();
    }
    catch(Exception e)
    {
        e.printStackTrace();
    }
            }
```

```
Nama method : sendMessage()
```

Input : - Output : -

Socket s=new Socket(ip, PORT);
OutputStream out=s.getOutputStream();
out.write(0);
out.write((fileName+"\n"+file.length+"\n").getBytes());
out.write(file);
out.write((signature+"\n").getBytes());
out.flush();
s.close();

Nama method : sendKey() Input : - Output : -

Enumeration<NetworkInterface> interfaces =NetworkInterface.getNetworkInterfaces();
while (interfaces.hasMoreElements()) {
NetworkInterface networkInterface = interfaces.nextElement();
if (networkInterface.isLoopback())continue;

```
// Don't want to broadcast to the loopback interface


                  for   (InterfaceAddress   interfaceAddress   :
networkInterface.getInterfaceAddresses())
                  {


System.out.println(interfaceAddress.getAddress());
                      InetAddress          broadcast          =
interfaceAddress.getBroadcast();
                      if (broadcast == null)continue;
                      // Use the address
                      try
                      {
                          System.out.println(broadcast);
                          Socket s=new Socket(broadcast, PORT);
                          OutputStream out=s.getOutputStream();
                          out.write(1);


out.write((key.getPrivateKey()+"\n").getBytes());


out.write((key.getPublicKey()+"\n").getBytes());
                          out.flush();
                          s.close();
                      }
                      catch(Exception e){}
                  }
              }
```

4. 2. Pengujian Alpha

Pengujian perangkat lunak dilakukan untuk memeriksa kekompakam antar komponen sistem dan sub sistemnya dengan tujuan utamanya adalah untuk memastikan elemen-elemen sistem berfungsi sesuai dengan yang diharapkan. Pengujian sistem juga termasuk pengujian program secara menyeluruh. Kumpulan program yang telah terintegerasikan perlu diuji coba atau tes untuk melihat apakah sebuah program dapat menerima dengan baik, memproses dan memberikan keluaran program yang baik pula.

4. 2. 1. Rencana Pengujian

Rencana pengujian akan dilakukan dengan menguji sistem secara alpha dan beta. Pengujian perangkat lunak aplikasi pembuatan *digital signature* ini menggunakan uji berdasarkan data yang terdapat di PT. Citra Bumi Madani.

4. 2. 2. Pengujian Sistem

Pengujian sistem konfigurasi kunci ini dilakukan dengan pengujian objektif. Tabel hasil pengujian sistem dari berabgai kemungkinan proses input yang dilakukan oleh pengguna selangkapnya dapat dilihat sebagai berikut :

Tabel 4. 3. *Skenario Pengujian Sistem*

Skenario Uji	Metode
Pengujian Jaringan Lan	*Black Box*
Pengujian Konfigurasi Kunci	*Black Box*
Pengujian Form Utama	*Black Box*

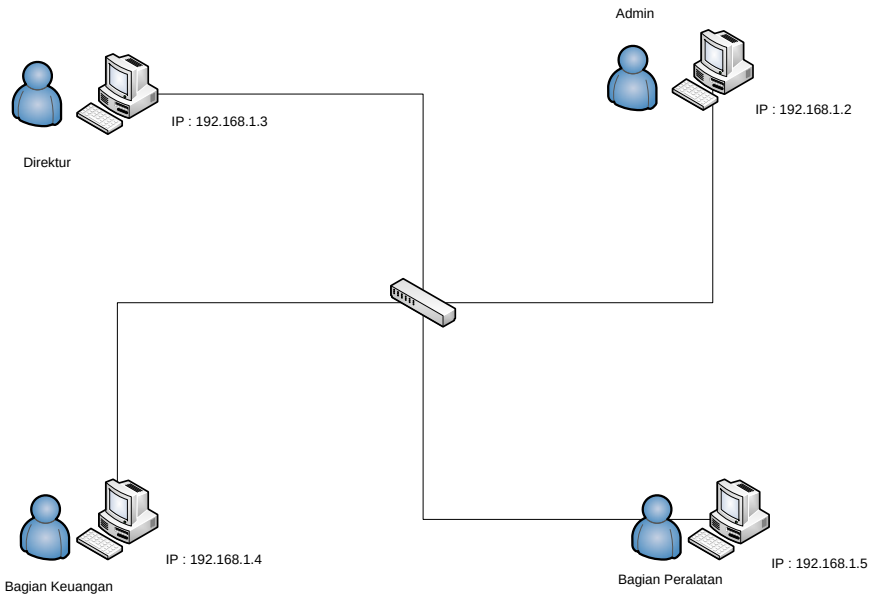
Pengujian Form Pesan Baru	*Black Box*
Pengujian Buka Pesan	*Black Box*
Pengujian add contact	*Black Box*

4. 2. 2. 1. Pengujian Jaringan LAN

Pada subbab ini dijelaskan mengenai pengujian jaringan di PT. Citra Bumi Madani.

4. 2. 2. 2. Skema jaringan

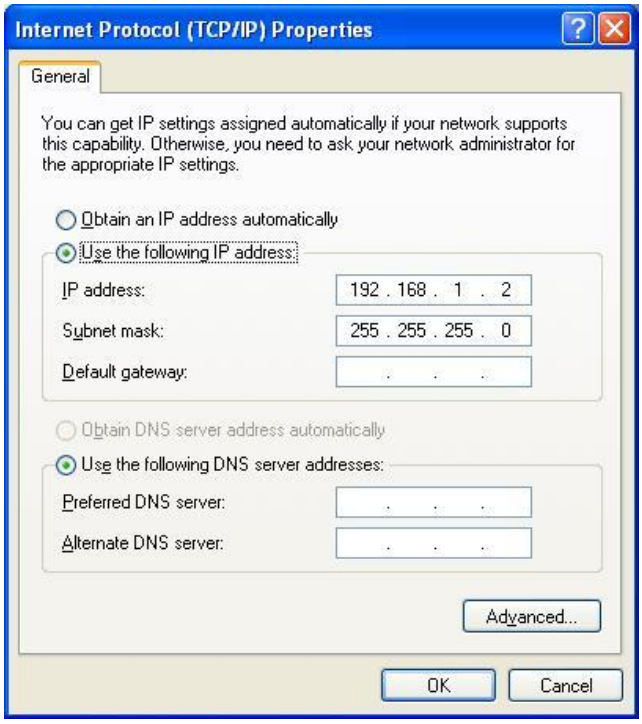
Topologi jaringan yang digunakan pada perusahaan adalah topologi star. Berikut skema jaringan yang digunakan pada perusahaan tersebut



Gambar 4. 1. Skema Jaringan

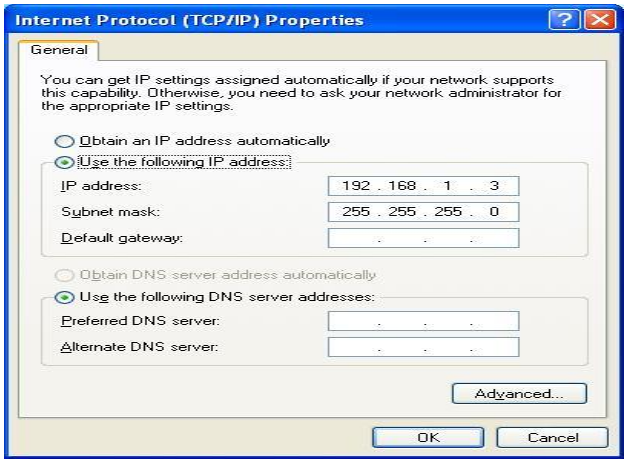
4. 2. 2. 3. Konfigurasi Jaringan

1. Admin



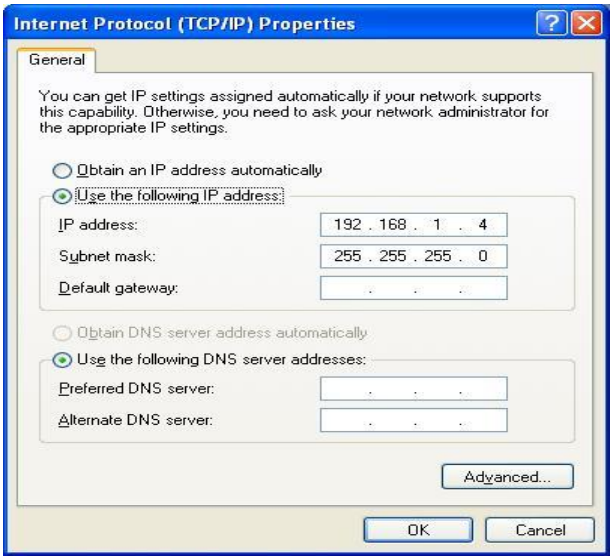
Gambar 4. 2. Konfigurasi Jaringan Admin

2. Direktur



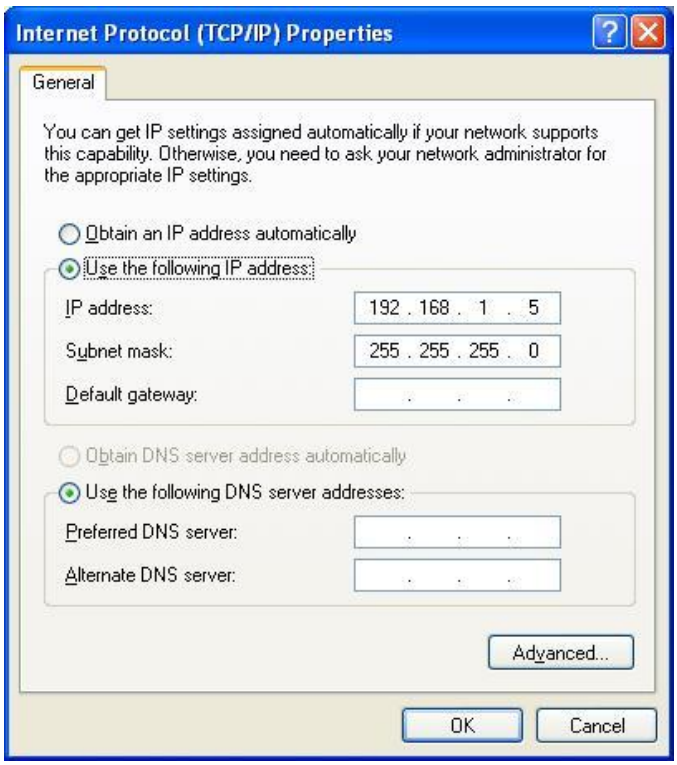
Gambar 4. 3. Konfigurasi Jaringan Direktur

3. Bagian peralatan



Gambar 4. 4. Konfigurasi Jaringan Bangian Peralatan

4. Bagian keuangan



Gambar 4. 5. Konfigurasi Jaringan Bagian Keuangan

4. 2. 3. Pengujian Fungsional Aplikasi

Pada subbab ini dibahas mengenai pengujian fungsional aplikasi *digital signature* ini dengan menggunakan metode *blackbox*.

4. 2. 3. 1 Pengujian Setting Key

Aksi	Reaksi yang diharapkan	Reaksi PL
Mengenerate kunci privat dan kunci publik	Generate key berhasil dilakukan dengan menekan tombol generate	Menampilkan kunci publik dan kunci privat
Menyimpan kunci privat dan kunci publik	Kunci berhasil tersimpan	Menyimpan kunci privat dan kunci publik
Reset kunci	Mengulang kunci yang akan di generate	Menghapus kunci

4. 2. 3. 2 Pengujian Compose Message

Aksi	Reaksi yang diharapkan	Reaksi PL
Tidak menginputkan alamat tujuan	Aplikasi menampilkan pesan peringatan untuk menginput alamat tujuan	Menampilkan pesan peringatan untuk penginputan alamat tujuan

Tidak membuat signature	Aplikasi akan menampilkan pesan peringatan untuk buat signature terlebih dahulu	Menampilkan pesan peringatan untuk membuat signature terlebih dahulu
Kirim file	Aplikasi berhasil Mengirimkan file dokumen	Mengirimkan file dokumen
Reset	Mengulang kembali proses compose message	Menghapus semua inputan

4. 2. 3. 3 Pengujian *Inbox*

Aksi	Reaksi yang diharapkan	Reaksi PL
Menghapus pesan	Aplikasi akan menghapus pesan yang ada di tabel inbox	Pesan terhapus
Membuka pesan	Akan muncul form view message bila di klik dua kali pada list inbox	Menampilkan form view message
Pesan baru	Pesan baru akan ditandai dengan suatu gambar	Pesan baru yang telah masuk berhasil diberi tanda
Pesan yang sudah dibaca	Pesan yang sudah dibaca akan dihilangkan tandanya	Tanda pada pesan baru telah dihilangkan

4. 2. 3. 4 Pengujian View Message

Aksi	Reaksi yang diharapkan	Reaksi PL
Buka file	Aplikasi akan menampilkan pesan sesuai dengan formatnya	File dokumen akan berhasil dibuka
Merubah kunci publik secara sembarang	Akan muncul pesan peringatan bahwa publik key tidak valid	Akan menampilkan pesan peringatan yang diharapkan
Merubah isi file	Muncul peringatan bahwa file tidak valid	Akan menampilkan pesan yang diharapkan dan file tidak berhasil di dekripsi
Save kontak	Akan muncul form save kontak	Menampilkan form yang diharapkan

4. 2. 3. 5 Pengujian Sent

Aksi	Reaksi yang diharapkan	Reaksi PL
Buka file	Aplikasi akan menampilkan pesan sesuai dengan formatnya	File dokumen akan berhasil dibuka
Merubah kunci publik	Akan muncul pesan peringatan	Akan menampilkan pesan peringatan yang

secara sembarang	bahwa publik key tidak valid	diharapkan
Merubah isi file	Muncul peringatan bahwa file tidak valid	Akan menampilkan pesan yang diharapkan dan file tidak berhasil di dekripsi
Save kontak	Akan muncul form save kontak	Menampilkan form yang diharapkan

4. 2. 3. 6 Pengujian Contact

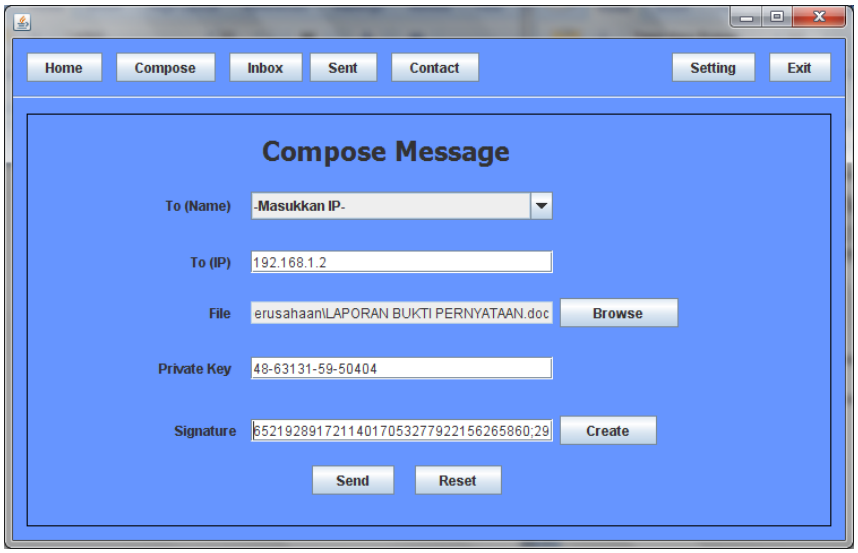
Aksi	Reaksi yang diharapkan	Reaksi PL
Menambah kontak	Menampilkan form tambah kontak alamat IP	Akan menampilkan form tambah IP
Menyimpan kontak IP	Nama dan alamat IP tersimpan pada kontak	Penyimpanan kontak berhasil dilakukan
Mengedit kontak	Menampilkan form edit kontak alamat IP	Akan menampilkan form edit kontak
Menghapus kontak	Data alamat kontak akan terhapus dari daftar jika telah dikonfirmasi	Data kontak terhapus

4. 2. 4. Pengujian Kasus Uji

Pada subbab ini akan dilakukan pengujian berdasarkan 3 kasus uji.

4. 2. 4. 1. Kasus uji pertama

Kasus uji pertama adalah pengguna mengirim file dokumen. Hasil dari kasus uji ini dapat dilihat pada gambar 4.6.



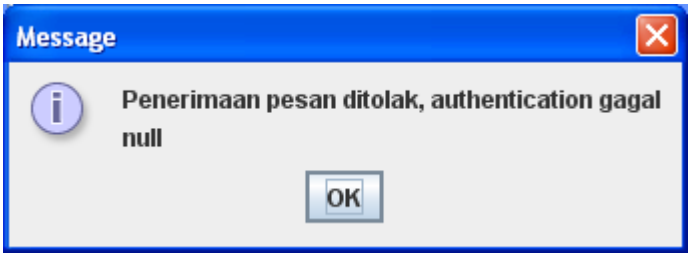
Gambar 4. 6. Kasus Uji 1

Jika si pegirim tidak menginformasikan kunci public pada si penerima maka file tidak bias dikirimkan dan akan muncul pesan berikut



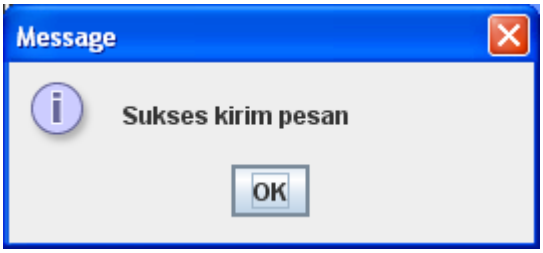
Gambar 4. 7. Pesan pada Pengirim

Kemudian akan muncul pesan berikut pada si penerima



Gambar 4. 8. Pesan Pada Penerima

Jika penerima sudah menginformasikan kunci public pada si penerima maka akan muncul pesan sebagai berikut

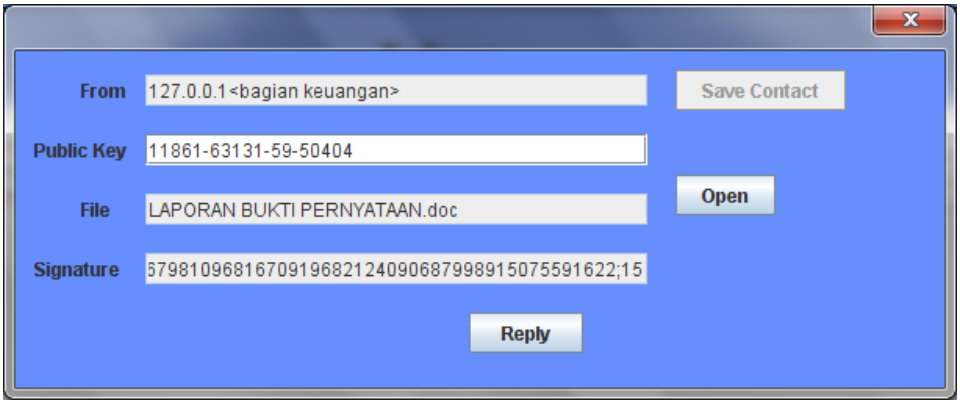


Gambar 4. 9. Berhasil kirim

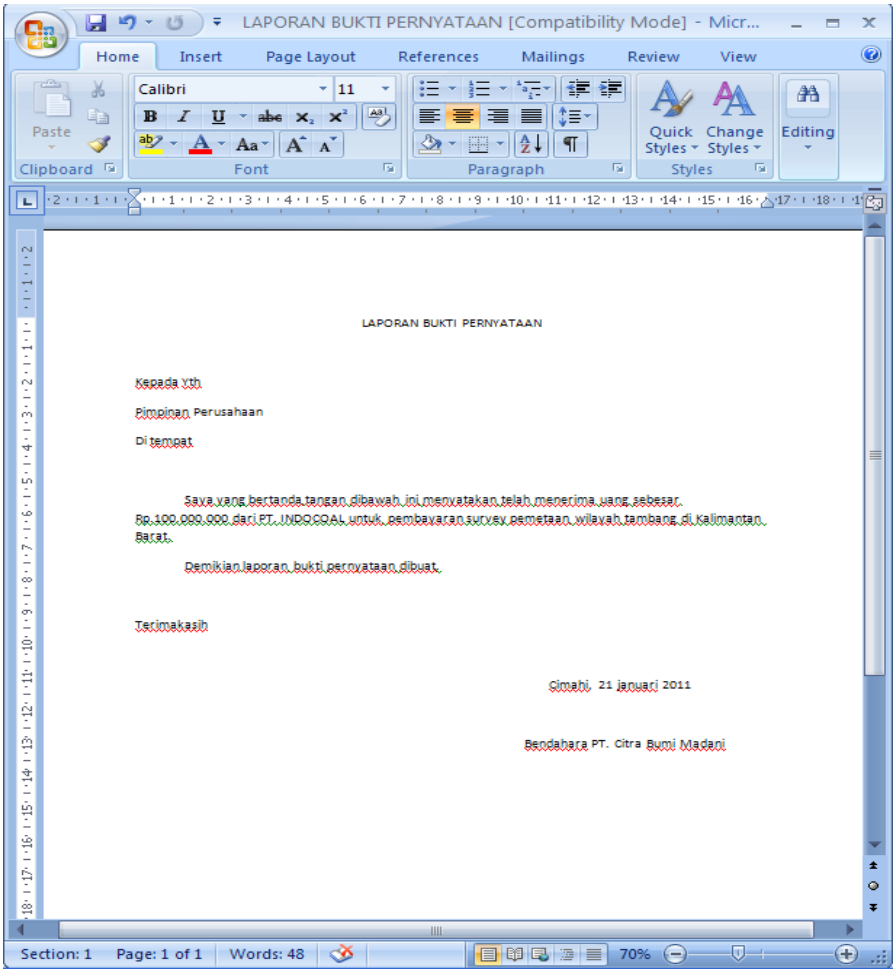
4. 2. 4. 2. Kasus Uji Kedua

Kasus uji yang kedua adalah pengujian otentifikasi file dokumen.

Jika otentifikasi berhasil maka hasilnya pada gambar 4.10 dan gambar 4.11.



Gambar 4. 10. Kasus Uji Terima Pesan



Gambar 4. 11. File Dokumen berhasil dibuka

Jika otentifikasi gagal maka hasilnya pada gambar 4.12 dan file tidak akan terbuka.



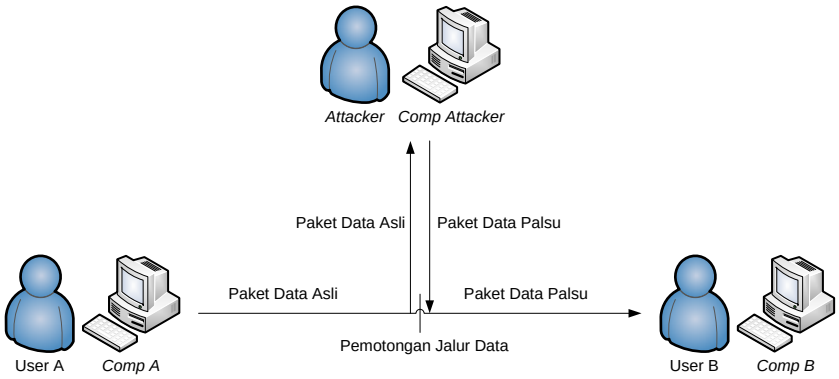
Gambar 4. 12.

4. 2. 4. 3. Kasus Uji Ketiga

Kasus uji yang ketiga adalah simulasi perubahan data oleh pihak ketiga. Simulasi serangan yang dihadapi pada sebuah jaringan. Serangan yang dihadapi bisa dianggap merugikan jika pihak ketiga telah merubah data yang dikirimkan oleh pengirim.

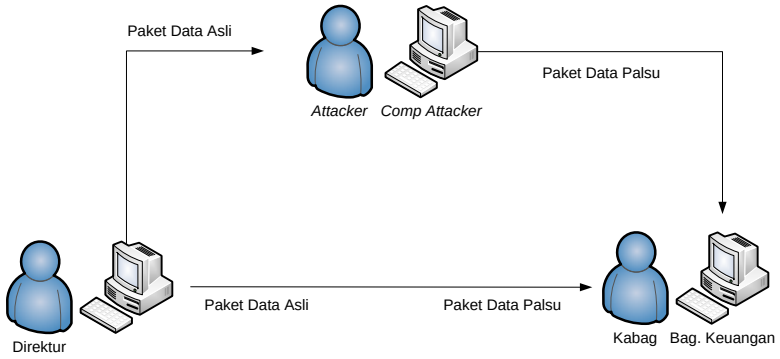
4. 2. 4. 4. Man In The Middle (MITM) Attack

Pola serangan yang akan dipakai adalah pola *Man In The Middle (MITM) Attack*. Pola tersebut dianggap sangat merugikan, karena terdapat pihak ketiga diantara pihak pertama dan pihak kedua. Berikut gambaran pola *Man In The Middle (MITM) Attact* :



Gambar 4. 13. ***Man In The Middle (MITM) Attack***

4. 2. 4. 5. Contoh Simulasi Man In The Middle (MITM) Attack



Gambar 4. 14. Contoh Simulasi ***Man In The Middle (MITM) Attack***

Disini diceritakan bagian keuangan akan mengirimkan sebuah file dokumen ke direktur. Berikut gambaran dokumen yang akan dikirimkan.

LAPORAN BUKTI PERNYATAAN

Kepada Yth

Pimpinan Perusahaan

Di tempat

Saya yang bertanda tangan dibawah ini menyatakan telah menerima uang sebesar Rp.100.000.000 dari PT. INDOCOAL untuk pembayaran survey pemetaan wilayah tambang di Kalimantan Barat.

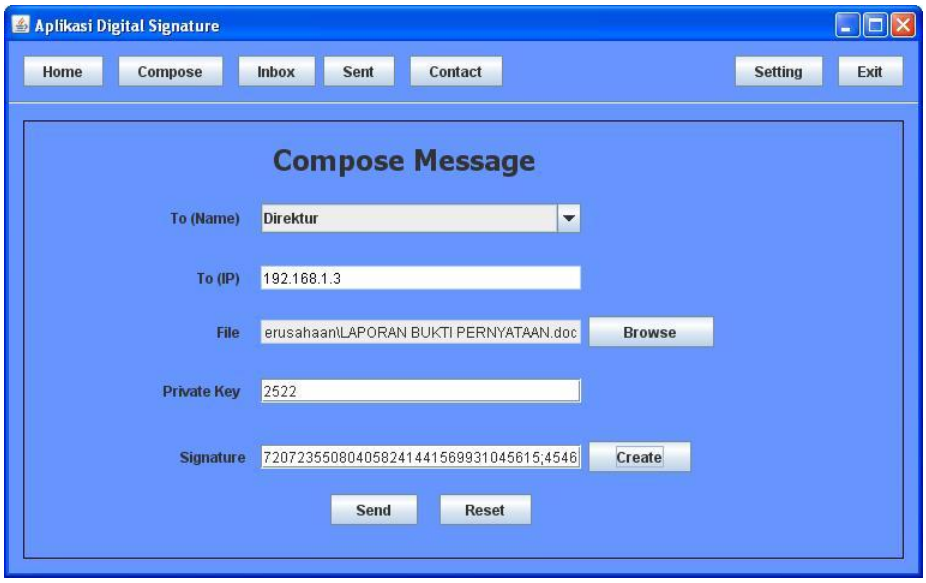
Demikian laporan bukti pernyataan dibuat.

Terimakasih

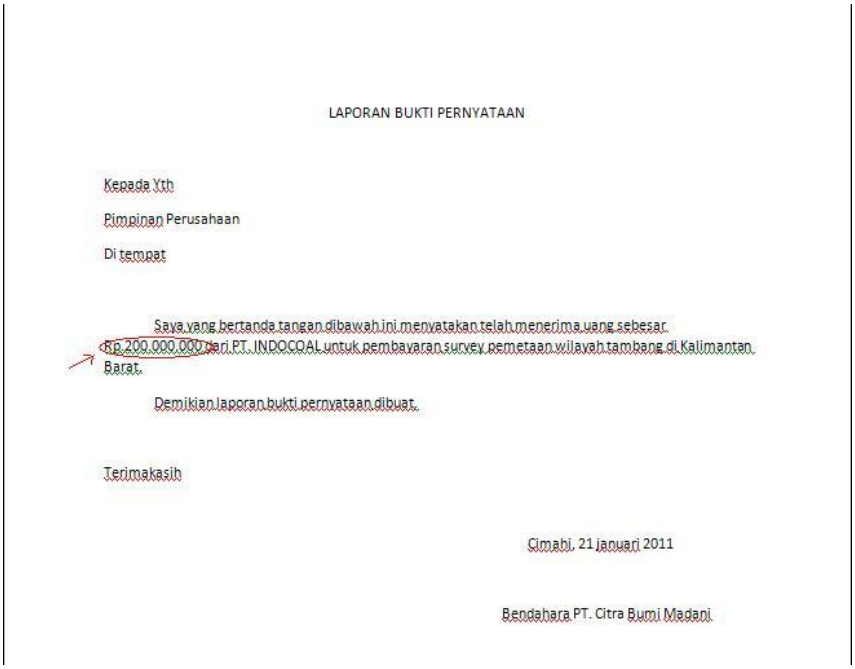
Cimahi, 21 januari 2011

Bendahara PT. Citra Bumi Madani

Dokumen asli



Ditengah-tengah perjalanan data tersebut diambil oleh pihak ketiga yaitu *Attacker*. Data tersebut kemudian dirubah isinya dan kemudian dikirim kembali ke direktur seperti gambar dibawah ini.



LAPORAN BUKTI PERNYATAAN

Kepada Yth

Pimpinan Perusahaan

Di tempat

Saya yang bertanda tangan dibawah ini menyatakan telah menerima uang sebesar Rp.200.000.000 dari PT. INDOCOAL untuk pembayaran survey pemetaan wilayah tambang di Kalimantan Barat.

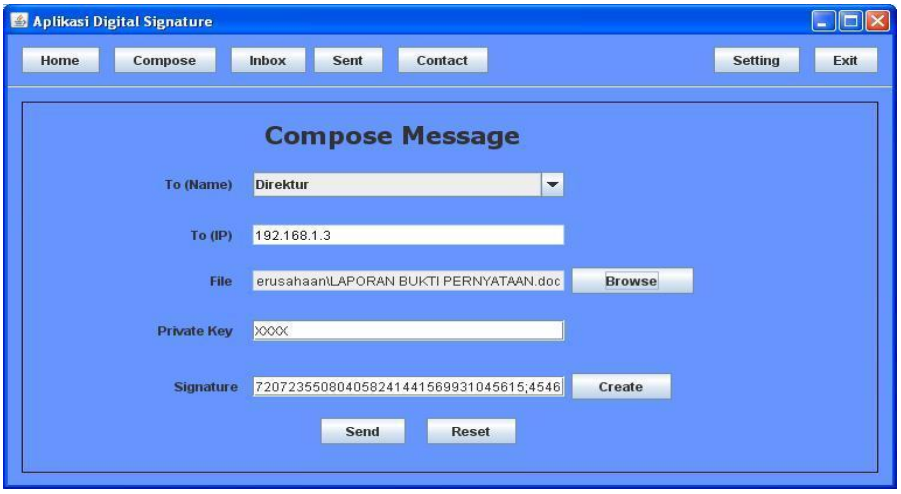
Demikian laporan bukti pernyataan dibuat.

Terimakasih

Cimahi, 21 januari 2011

Bendahara PT. Citra Bumi Madani

Dokumen palsu



Pengiriman dokumen palsu

Direktur kemudian menerima file dokumen yang telah dirubah tersebut dan kemudian mengotentikasinya.



Setelah diotentikasi ternyata hasilnya sebagai berikut :



Gambar 4. 15. Hasil Otentikasi Ditolak

Hasil tersebut menyatakan bahwa dengan data tersebut tidak valid, dikarenakan isi file tersebut telah dirubah oleh *Attacker*. Hal ini menyatakan bahwa walaupun pada saat pengiriman file tersebut memiliki signature yang sama tetapi isi file yang dikirimkan berbeda. Sehingga aplikasi menyatakan bahwa file dokumen tersebut tidak valid, dikarenakan pada saat perbandingan nilai hashing pada dokumen dengan nilai hashing dari dekripsi signature berbeda.

4. 2. 5. Pengujian Konfigurasi Kunci

Konfigurasi Kunci : Kunci privat dan kunci publik

Tabel 4. 4. Pengujian Konfigurasi Kunci

Kasus dan Hasil Uji (data yang dimasukan benar)			
Data Masukan	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
Data konfigurasi kunci :Kunci privat dan kunci publik	Kunci privat dan publik tercantum	Kunci sesuai dengan perhitungan.	[√]Diterima [ ]Ditolak

Kasus dan Hasil Uji (data yang dimasukan salah)			
Data Masukan	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
Data konfigurasi Kunci:Kunci privat dan kunci publik	Tidak mengisi kunci sistem menampilkan pesan error	Kunci diisi	[√]Diterima [ ]Ditolak

4. 2. 6. Pengujian Pesan Baru

Tabel 4. 5. Pengujian Pesan Baru

Kasus dan Hasil Uji (data yang dimasukan benar)			
Data Masukan	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
IP penggguna	Berhasil melakukan	Pengguna dapat	[√]Diterima

	input IP	mengisi IP pengguna sesuai yang diharapkan	[ ]Ditolak
Klik tombol browse	Berhasil pilih dokumen	Pengguna dapat memilih file dokumen	[√]Diterima [ ]Ditolak
Masukan private key	Berhasil input kunci privat	Pengguna dapat memasukan kunci privat	[√]Diterima [ ]Ditolak
Klik tombol create	Berhasil menghasilkan signature	Signature dapat muncul	[√]Diterima [ ]Ditolak
Klik Tombol Send	Dokumen tersimpan di data storage	Dokumen disimpan dalam storage, sesuai yang diharapkan	[√]Diterima [ ]Ditolak
Klik Tombol Cancel	dokumen tidak jadi dikirim	Dokumen tidak jadi dikirim , sesuai dengan yang diharapkan	[√]Diterima [ ]Ditolak

Kasus dan Hasil Uji (data yang dimasukan salah)			
Data Masukan	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
Data IP tidak di isi dan file tidak dipilih	Pengisian IP dan file tidak dipilih, sistem akan menampilkan pesan error	Data yang tidak diisi, dan memberikan pesan “anda memasukan IP	[√]Diterima [ ]Ditolak

		terlebih dahulu ” dan pesan “ anda belum memilih file”.sesuai yang diharapakan	

4. 2. 7. Pengujian Buka Pesan

Tabel 4. 6. Table Pengujian Buka Pesan

Kasus dan Hasil Uji (data yang dimasukan benar)			
Data Masukan	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
Klik pesan yang ada pada form pesan masuk	Berhasil memunculkan form buka pesan	Pengguna dapat membuka, menyimpan IP pengirim dan membalas pesan sesuai yang diharapkan	[√]Diterima [ ]Ditolak
Klik tombol save	Berhasil menyimpan pesan	Pengguna dapat menyimpan pesan yang berisi dokumen	[√]Diterima [ ]Ditolak
masukan publik key	Publik key tercantum atau pengguna dapat input manual	kunci publik berhasil di inputkan	[√]Diterima [ ]Ditolak
Klik Tombol Open	Berhasil membuka pesan yang berisi file	Pesan dapat di buka , sesuai yang	[√]Diterima [ ]Ditolak

	dokumen	diharapkan	
Klik Tombol Reply	Berhasil membalas pesan yang berisi file dokumen	Dokumen berhasil dikirim, sesuai dengan yang diharapkan	[√]Diterima [ ]Ditolak

Kasus dan Hasil Uji (data yang dimasukan salah)			
Data Masukan	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
Kunci publik tidak di inputkan	Pesan tidak bisa dibuka dan dokumen tidak valid	Dokumen tidak dapat terbuka	[√]Diterima [ ]Ditolak

4. 2. 8. Pengujian Add Contact

Tabel 4. 7. Table Pengujian add contact

Kasus dan Hasil Uji (data yang dimasukan benar)			
Data Masukan	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
Klik tombol add	Berhasil memunculkan form untuk memasukan IP baru	contact berguna untuk menyimpan data IP	[√]Diterima [ ]Ditolak
Klik tombol delete	Berhasil menghapus kontak IP	Pengguna dapat menghapus kontak IP	[√]Diterima [ ]Ditolak

Kasus dan Hasil Uji (data yang dimasukan salah)			
Data Masukan	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**

Tombol add tidak di klik	IP baru tidak dapat dimasukan	Tidak memasukan IP proses penyimpanan gagal	[√]Diterima [ ]Ditolak

4. 2. 9. Kesimpulan Hasil Pengujian Alpha

Berdasarkan hasil pengujian yang telah dilakukan memberi kesimpulan bahwa pada peoses masih memungkinkan adanya proses yang belum maksimal diciptakan tetapi secara fungsional sistem sudah dapat menghasilkan *output* yang diharapkan.

4. 2. 10. Pengujian Beta

Pengujian beta merupakan pengujian yang dilakukan secara objektif dimana diuji secara langsung ke lapangan yaitu instansi yang bersangkutan dengan wawancara mengenai kepuasan *user* dengan kandungan *point* syarat *user friendly* untuk selanjutnya dibagikan kepada PT. Citra Bumi Madani

Adapun wawancara pengujian beta untuk Instansi tersebut yaitu :

1. Apakah perangakat lunak yang dibangun memudahkan pengguna dalam melakukan pengiriman dokumen menggunakan *digital signature*.
   a. Sangat Tidak Setuju
   b. Tidak Setuju
   c. Biasa Saja
   d. Setuju
   e. Sangat Setuju
2. Aplikasi perangkat lunak *digital signature* ini cukup akurat dalam mengetahui keontetikan dokumen yang dikirim

a. Sangat Tidak Setuju
b. Tidak Setuju
c. Biasa Saja
d. Setuju
e. Sangat Setuju

3. Apakah anda kesulitan menggunakan perangkat lunak *digital signature* ini

a. Sangat Tidak Setuju
b. Tidak Setuju
c. Biasa Saja
d. Setuju
e. Sangat Setuju

4. Apakah perangkat lunak *digital signature* ini memberikan keefisienan waktu dalam pengiriman dokumen antar bagian dalam perusahaan anda

a. Sangat Tidak Setuju
b. Tidak Setuju
c. Biasa Saja
d. Setuju
e. Sangat Setuju

5. Aplikasi tampilan dari perangkat lunak *digital signature* ini menarik

a. Sangat Tidak Setuju
b. Tidak Setuju
c. Biasa Saja
d. Setuju
e. Sangat Setuju

6. Aplikasi *digital signature* ini sudah *user friendly*

a. Sangat Tidak Setuju
b. Tidak Setuju
c. Biasa Saja
d. Setuju
e. Sangat Setuju

PERTANYAAN DAN HASIL PENGUJIAN

” Apakah perangakat lunak yang dibangun memudahkan pengguna dalam melakukan pengiriman dokumen menggunakan *digital signature*”

Kategori jawaban	Ya	Cukup	Kurang	Tidak
Frekuensi jawaban	5	4	0	0

Persentase nilai

Jumlah Frekuensi	5	4	0	0
Jumlah Persentase	50%	40%	0%	0%

” Aplikasi perangkat lunak *digital signature* ini cukup akurat dalam mengetahui keontetikan dokumen yang dikirim”

Kategori jawaban	Ya	Cukup	Kurang	Tidak
Frekuensi jawaban	8	1	0	0

Persentase nilai

Jumlah Frekuensi	8	1	0	0
Jumlah Persentase	80%	10%	0%	0%

"Apakah anda kesulitan menggunakan perangkat lunak *digital signature* ini"

Kategori jawaban	Ya	Cukup	Kurang	Tidak
Frekuensi jawaban	0	0	1	8

Persentase nilai

Jumlah Frekuensi	0	0	1	8
Jumlah Persentase	0%	0%	10%	80%

"Apakah perangkat lunak *digital signature* ini memberikan keefisienan waktu

Kategori jawaban	Ya	Cukup	Kurang	Tidak
Frekuensi jawaban	6	3	0	0

Persentase nilai

Jumlah Frekuensi	6	3	0	0
Jumlah Persentase	60%	30%	0%	0%

"Aplikasi tampilan dari perangkat lunak *digital signature* ini menarik"

Kategori jawaban	Ya	Cukup	Kurang	Tidak
Frekuensi jawaban	5	4	0	0

Persentase nilai

Jumlah Frekuensi	5	4	0	0
Jumlah Persentase	50%	40%	0%	0%

4. 2. 11. Kesimpulan Hasil Pengujian Beta

Dari pengujian beta yang telah dilakukan yaitu dengan pengujian perhitungan pilihan kategori jawaban dari kuisioner yang tetah dibagikan di lapangan didapat kesimpulan bahwa aplikasi yang dibangun dinilai dapat membantu dalam pengiriman dokumen antar bagian dengan memakai *digital signature*, sesuai kebutuhan, menjadi suatu sarana yang bermanfaat, mengefisienkan waktu, info yang disampaikan cepat, mudah dan akurat.

4. 2. 12. Pengujian Sistem

Bila sistem telah dapat bekerja dengan baik dan dapat menggantikan sistem lama, maka sistem baru dialihkan untuk menggantikan sistem lama. Cara pengalihan sistem yang digunakan adalah dengan pendekatan dengan mengoperasikan sistem baru bersama-sama selama periode tertentu.