|  |  |  |
| --- | --- | --- |
| 年级 | 学号 | 姓名 |
| 2015级 | 15020032003 | 葛畅 |
| 2015级 | 15020032011 | 刘子悦 |
| 2015级 | 15020032030 | 张雪冰 |
|  |  |  |

指导教师：林喜军老师

小组成员：

时间：2017年9月

**密码系统综合设计**

小型CA系统



目录

[1. 实验目的 1](#_Toc493075690)

[2. X509证书格式 1](#_Toc493075691)

[2.1常规界面 1](#_Toc493075692)

[2.2详细信息 2](#_Toc493075693)

[2.3证书路径 4](#_Toc493075694)

[2.4根证书和二级证书的区别 4](#_Toc493075695)

[3. 实验要求 4](#_Toc493075696)

[4. 制作数字证书前期准备 5](#_Toc493075697)

[5. 具体实现 6](#_Toc493075698)

[5.1设置加密提供者 6](#_Toc493075699)

[5.2密钥对生成函数产生密钥对KeyPair generateKeyPair(int seed) 7](#_Toc493075700)

[5.3证书制作函数 7](#_Toc493075701)

[5.4 选取证书存储路径 8](#_Toc493075702)

[5.5 数证制作 8](#_Toc493075703)

[6. 界面介绍 10](#_Toc493075704)

[6.1完整界面 10](#_Toc493075705)

[6.2分块界面介绍 11](#_Toc493075706)

[6.2.1基本信息填写 11](#_Toc493075707)

[6.2.2证书相关信息填写 11](#_Toc493075708)

[6.2.3证书类型 12](#_Toc493075709)

[6.2.4存储路径的选择 12](#_Toc493075710)

[6.2.5数证制作 12](#_Toc493075711)

[6.2.6证书制作过程状态显示栏 12](#_Toc493075712)

[6.3生成证书界面 13](#_Toc493075713)

[6.3.1主要代码功能介绍 14](#_Toc493075714)

[7. 遇到的问题及解决办法 18](#_Toc493075715)

[7.1路径问题 18](#_Toc493075716)

[7.1.1路径读取问题 18](#_Toc493075717)

[7.1.2路径存储问题 19](#_Toc493075718)

[7.2常用RDN属性简写问题 19](#_Toc493075719)

[7.3 int型与String型转换问题 19](#_Toc493075720)

[7.4密钥长度问题 20](#_Toc493075721)

[8. 实验不足和需改进的地方 20](#_Toc493075724)

[9. 实验感悟 21](#_Toc493075729)

[10. 分工 21](#_Toc493075730)

# 实验目的

1.通过制作数字证书的过程，了解PKI的核心-认证机构（Certification Authority,CA）的颁发证书工作机制。

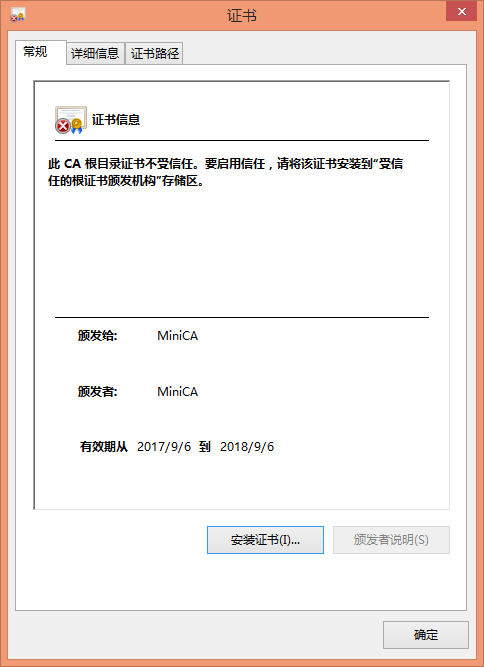
2.通过程序制作数字证书，熟悉X509证书格式，包括所用加密算法，主体名称结构DN（Distinguish Name）等相关信息。

3.通过此项目了解X509证书的定义：”权威机构通过对包含有待认证的主体的一系列信息的待签名证书”（TBS,to be signed）进行数字签名来表示该机构对他的认可。一份包含了待认证的主题的相关信息的TBS再加上CA，使用自己的私钥进行签名，就构成了一份标准的X509证书。

4.通过CA颁发数字证书，了解根证书与二级证书的区别

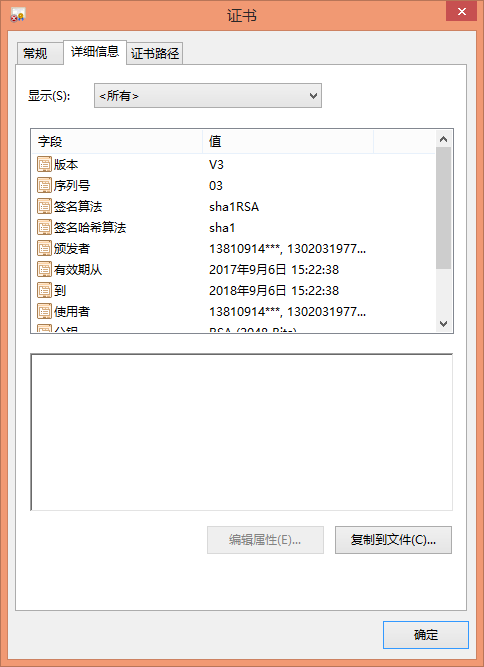
# X509证书格式

## 2.1常规界面



常规选项卡里，主要介绍了证书信息，颁发给和颁发者的信息以及有效日期。

## 2.2详细信息



（1）版本

版本指出该证书使用了哪种版本的X.509标准，版本号会影响证书中的一些特定信息。目前的版本是V3。

（2）序列号

序列号由CA给予每一个证书分配的唯一的数字型编号，当证书被取消时，实际上是将此证书序列号放入由CA签发的CRL（Certificate Revocation List证书作废表，或证书黑名单表）中。这也是序列号唯一的原因。

（3）签名算法和签名哈希算法

签名算法和签名哈希算法是数字签名中所使用的非对称加密体制和哈希函数算法，可以通过根证书中的公钥对这个证书中的指纹进行解密。

（4）颁发者

颁发者表明这个数字证书是由哪个CA公司生成的。

（5）有效期

有效期是证书的生成日期以及证书的截至日期，证书过期之后，认证服务器会重新校验。

（6）使用者

使用者包括使用者的详细信息，有很多的RDN构成的DN序列，包括通用名称，机构单元名称，机构名等信息。

（7）公钥

公钥包括证书持有人的公钥、所用算法和其他相关参数，主要用来对消息进行加密。

（8）指纹及指纹算法

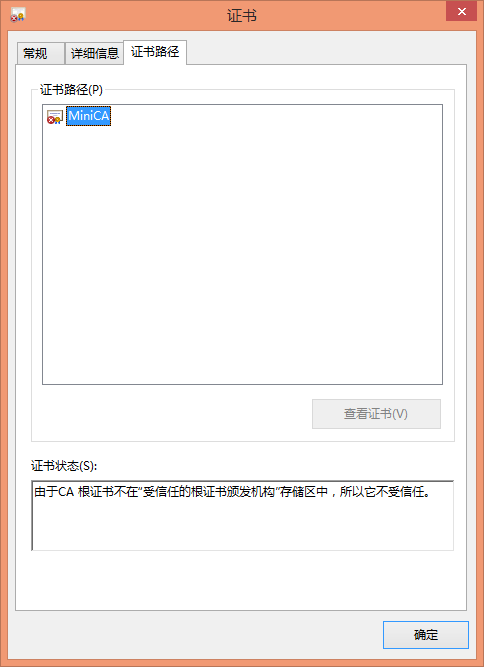
指纹以及指纹算法，在证书发布的时候，发布机构会根据指纹算法先计算出整个证书的hash值，并使用证书发布机构的私钥对其进行签名构成一个指纹，并将指纹与该证书放在一起。

（9）DN结构

此小型CA机构的使用DN结构为：

C=国家 CN=名称 ST=省份 L=城市 OU=组织EmailAddress=电子邮件SN=名字 TELEPHONE\_NUMBER=电话号码 STREET=街道 POBOX=邮政 PostalCode=邮政编码 UNIQUE\_IDENTIFIER=身份证 T=手机号码

## 2.3证书路径



正常情况下应该显示出证书的信任链，从根证书一层一层向下。如下图：

C:\Users\Administrator\AppData\Roaming\Tencent\Users\919426791\TIM\WinTemp\RichOle\@N56Y0O)4JQ$5H1HV7DXA3E.png

## 2.4根证书和二级证书的区别

根证书：根证书的颁发者与持有者相同，签名形式为自签名，即用颁发者的私钥对包含其公钥的证书进行签名。

二级证书：二级证书的颁发者是认证机构（CA），用于签名的私钥属于证书颁发者，证书包含的公钥属于证书持有者。

# 实验要求

1.程序设计者在根证书和二级证书（自定义证书）之间任选一种制作

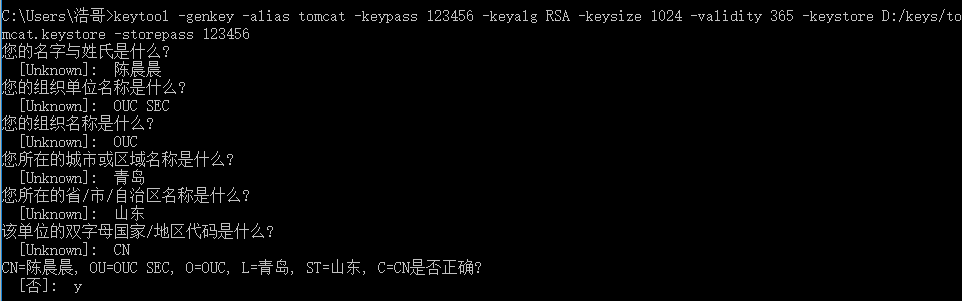
2.利用程序创建图形用户界面（GUI）以实现友好的用户交互，用户在界面上输入希望证书包含的信息，程序获取页面的信息后，生成证书，并存储在用户选择的文件下。

# 制作数字证书前期准备

实验要求程序创建图形用户界面（GUI）实现友好的用户交互。NetBeans IDE GUI构建器，在支持多种语言的同时，使得用户无需深入了解布局管理器，便可以构建具有专业外观的 GUI。在使用时，我们只需将相关组件置于所需的位置，即可设置窗体布局，NetBeans平台自动生成代码，大大简化程序图形界面创建过程。上网查阅相关资料，我们发现java已经将x509证书封装成完整的jar包并且提供众多方法和接口，因此我们决定用java结合Netbeans实现此小型CA系统的开发。通过关键字“java语言数字证书”搜索，我们获得bouncycastle是一种用于java平台的开源轻量级密码术包，支持大量加密算法，适用于数字证书的制作。bouncycastle-jce-jdk13-112 jar包和java自带的security包是支持此程序运行的重要jar包。

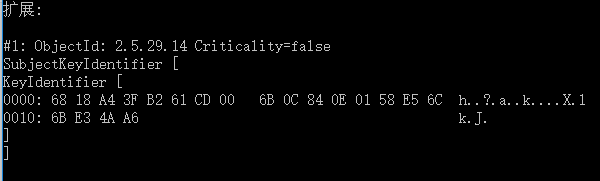
开发语言和平台确定后，我们着手对数字证书深入了解。通过大量资料搜索， “根证书”和“二级证书”成为我们的关注点。并且得知，java自带的工具keytool既可以产生根证书，又可以用此根证书认证二级证书。利用该工具生成证书的过程，我们不仅弄明白了制作证书的过程，而且清楚了根证书和二级证书的区别，制作二级证书即需要获取认证机构的私钥，DN结构，并用其私钥对二级证书进行签名后颁发。

此图为根证书的制作。



下图为查看的制作的根证书的具体信息





# 具体实现

## 5.1设置加密提供者

在程序里调用Security.addProvider（）此函数添加加密提供者，函数参数是加密提供者，加密提供者包含了里面包含了各种加密方法，生成证书格式等。因为此程序引用Bouncy castle包轻量级密码术包，故采用该包里的加密提供者BouncyCastleProvider()；

Security.addProvider(new BouncyCastleProvider());函数参数是加密提供者，里面包含了各种加密方法，生成证书格式等

## 5.2密钥对生成函数产生密钥对KeyPair generateKeyPair(int seed)

1.创建密钥对生成器KeyPairGenerator的对象kpg用以产生证书颁发者和持有者的密钥对，利用类KeyPairGenerator中的方法KeyPairGenerator.getInstance()对kpg进行实例化，该方法的参数为”RSA””DES”等加密算法，本程序采用”RSA”；

2.用长度为seed的字节生成强随机数列，并从界面获得用户设定的密钥长度length，调用kpg.initialize（）方法,将length与强随机数列作为其参数，初始化密钥对生成器kpg；

3.调用方法kpg.generateKeyPair()生成密钥对keypair并作为函数值返回；

## 5.3证书制作函数

X509Certificate generateCert(KeyPair keyPair\_root,KeyPair keyPair\_user)，参数类型为公钥对。若用户选择制作根证书，则keypair\_root=keypair\_user，否则分别传递根CA的密钥对与证书持有者的密钥对；

1.实例化生成X509V3证书类X509V3CertificateGenerator的对象certGen，并创建X509Certficate 类型的证书cert

2.设置证书的序列号serialnum，调用certGen.setSerialNumber()方法，该函数参数为用字符串实例化后的BigInteger类型。证书的序列号的唯一性，决定每生成一份证书，该证书的序列号都是不同的。因此本程序采取获取程序运行时的系统当前时间为该证书的序列号。并将当前转换成字符串类型，赋值给serialnum,，用其初始化BigInteger对象，作为参数传递给该方法。

Date now = new Date();//now为系统当前时间SimpleDateFormat dateFormat=new SimpleDateFormat("yyyyMMddHHmmss");

//yyyyMMddHHmmss为日期的的格式private final String serialnum = dateFormat.format( now );

//将serialnum赋值为当前日期的字符串形式

certGen.setSerialNumber(new BigInteger(serialnum));//设置证书的序列号

3.设置证书有效日期：在本程序中，该证书有效日期的起止日期curday为程序运行的当前系统时间。首先利用函数getDay()获取用户在界面输入的证书有效天数num，将curday和num作为参数传递给函数addDate()求得证书有效日期的截止日期effday。

certGen.setNotBefore(curday);

certGen.setNotAfter(effday);

4.根据用户的选择设置证书颁发者的DN, IssuerDN和证书持有者的DN, subjectDN。用户若选择制作CA证书，则IssuerDN=subjectDN=用户输入的身份信息；用户若选择制作自定义证书，则本程序默认为颁发者CA的DN信息为CN=Miss, OU=Network Security, O=OUC, C=China，subjectDN=用户输入的身份信息。

5.设置该证书持有者的公钥

certGen.setPublicKey(keypair\_user.getPublic());

6.指定该证书的签名算法。本程序采用SHA1签名算法。CertGen.setSigntureAlgorithm(“SHA1WithRSA”);

7.利用CA的私钥对该证书进行签名，最终生成证书cert.即cert=certGen.generateX509Certificate(keyPair\_root.getPrivate(),

”BC”);//该函数返回X509格式的证书；

## 5.4 选取证书存储路径

该程序通过NetBeans中Swing控件“文件选择器”实现，事件为鼠标点击文件选择器，来执行该函数，并返回全局变量绝对路径path；

## 5.5 数证制作

鼠标点击“数证制作”按钮实现程序获取界面用户输入的证书相关信息，并实现证书的存储

void btnMakeMouseClicked(java.awt.event.MouseEvent evt)；

1.调用getStringInfo(String[] string)函数获取用户输入的身份信息

该函数通过一组类似如下的if语句，判断各行信息栏前的复选框选中与否，若被选中，读取该信息栏的主要内容，存入subjectDN字符串中作为证书持有者的DN信息。JCBName为某个复选框的变量名称。

if(jCBName.isSelected()){

string[0]="CN="+jTFname.getText();

subjectDN=subjectDN+string[0]+"，，";

}

2.调用一次密钥对生成函数generateKeyPair()，生成key\_root根CA的密钥对，若用户选择自定义证书制作，再调用一次该函数生成证书持有者的密钥对keyPair\_user.

3.将生成的密钥对传递给证书生成函数generateCert(),生成证书cert/

4.创建一个文件输出流对象fos，在路径path下新建一个以.cer为后缀的文件。证书名字任取。调用fos.write()函数将新生成的cert写入fos中。因为实验要求采取DER编码格式，故通过调用函数cert.getEncoded()实现DER编码。最终写入文件的格式为fos.write(cert.getEncoded());

# 界面介绍

## 6.1完整界面



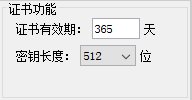
## 6.2分块界面介绍

### 6.2.1基本信息填写



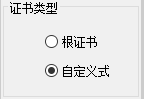
功能介绍：该界面提供证书持有人的身份信息，用户在最前方的复选框勾选想要呈现的字段名称，并在所勾选的一栏更改初始化信息，程序通过复选框的选中状态，获取用户的输入；该界面是以滚动窗格的形式呈现，如小图所示，用户需要向下拉动窗口方可呈现所有的字段名称。

### 6.2.2证书相关信息填写



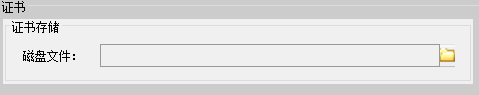
用户在证书有效日期一栏填写该证书的有效日期，组合框里包含多种合法的密钥长度，用户自选所需密钥的长度

### 6.2.3证书类型



该程序可制作两种证书，CA根证书和CA颁发的二级证书，用户通过按钮，选取想要产生的证书类型。

### 6.2.4存储路径的选择



用户点击黄色的小图标，弹出存储路径对话框，用户选取希望保存证书的路径，黄色图标前的空白区域自动呈现用户的选择。

### 6.2.5数证制作



用户填写完证书所需信息后，点击此按钮便可制作证书。

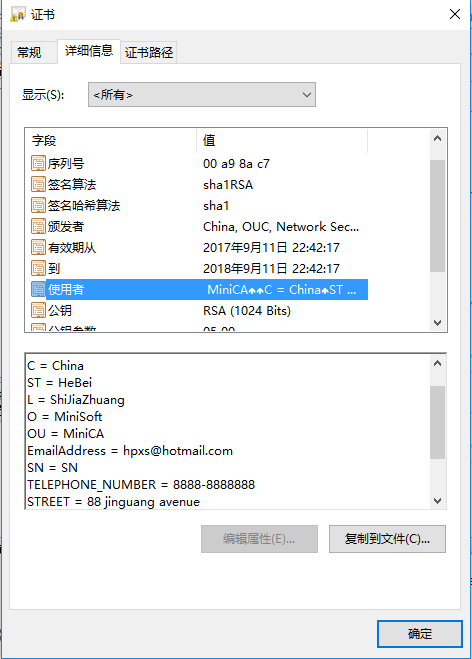
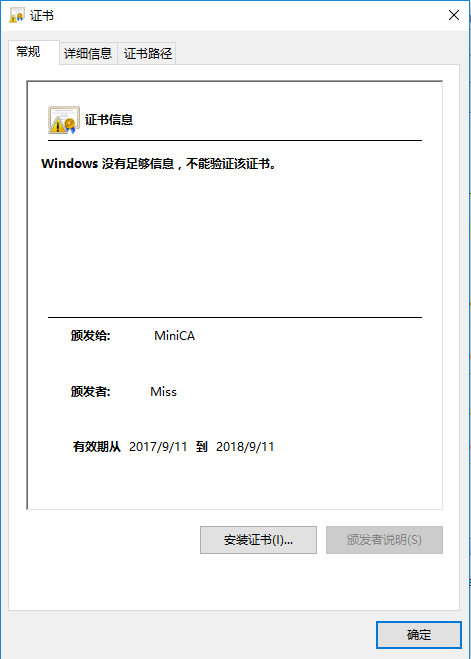
### 6.2.6证书制作过程状态显示栏

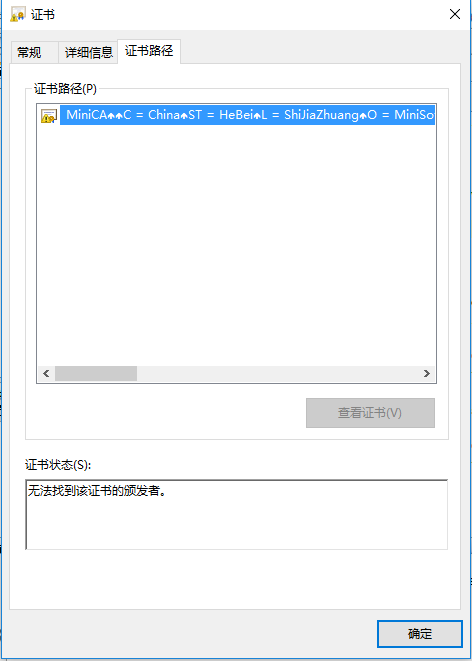


该栏不需要用户填写任何信息。用户点击数证制作按钮后，程序开始证书制作，证书制作的状态会在该栏显示，提示用户。

## 6.3生成证书界面

注：本实验报告展示的证书，基本信息一栏全部勾选且为初始化信息，证书有效日期为365天，密钥长度为1024bits，证书类型为自定义证书。





### 6.3.1主要代码功能介绍

**1.文件选择器**

private void btnFileChooseMouseClicked(java.awt.event.MouseEvent evt) {

JFileChooser filechooser= new JFileChooser(); filechooser.setFileSelectionMode(JFileChooser.DIRECTORIES\_ONLY);

filechooser.setDialogTitle("Select path to save");

int result = filechooser.showOpenDialog(this);

if (result == JFileChooser.APPROVE\_OPTION){

path = filechooser.getSelectedFile().getAbsolutePath();

PathShow.setText(path);

}

}

用户点击文件图标便会触发该事件，生成一个文件选择器对象，用户选择存储路径后，获得该路径path,并显示在界面上。

**2.获取用户输入的基本身份信息**

private void getStringInfo(String[] string){

jTextAreaShow.setText("加载内部证书...");

if(jCBName.isSelected()){

string[0]="CN= "+jTFname.getText();

subjectDN=subjectDN+string[0]+"，，";

}

if(jCBCountry.isSelected()){

string[1]="C = "+jTFcountry.getText();

subjectDN=subjectDN+string[1]+"，";

}

if(jCBProvince.isSelected()){

string[2]="ST = "+jTFprovince.getText();

subjectDN=subjectDN+string[2]+"，";

}

if(jCBCity.isSelected()){

string[3]="L = "+jTFcity.getText();

subjectDN=subjectDN+string[3]+"，";

}

if(jCBCommunity.isSelected()){

string[4]="O = "+jTFcommunity.getText();

subjectDN=subjectDN+string[4]+"，";

}

if(jCBUnit.isSelected()){

string[5]="OU = "+jTFunit.getText();

subjectDN=subjectDN+string[5]+"，";

}

if(jCBMail.isSelected()){

string[6]="EmailAddress = "+ jTFemail.getText();

subjectDN=subjectDN+string[6]+"，";

}

if(jCBSurName.isSelected()){

string[7]="SN = "+jTFsurname.getText();

subjectDN=subjectDN+string[7]+"，";

}

if(jCBTeleNum.isSelected()){

string[8]="TELEPHONE\_NUMBER = "+jTFtelephone.getText();

subjectDN=subjectDN+string[8]+"，";

}

if(jCBStreet.isSelected()){

string[9]="STREET = "+jTFstreet.getText();

subjectDN=subjectDN+string[9]+"，";

}

if(jCBPoBox.isSelected()){

string[10]="POBox = "+jTFpobox.getText();

subjectDN=subjectDN+string[10]+"，";

}

if(jCBPoCode.isSelected()){

string[11]="PostalCode = "+jTFpocode.getText();

subjectDN=subjectDN+string[11]+"，";

}

if(jCBID.isSelected()){

string[12]="UNIQUE\_IDENTIFIER = "+jTFID.getText();

subjectDN=subjectDN+string[12]+"，";

}

if(jCBPhoneNum.isSelected()){

string[13]="T = "+jTFphone.getText();

subjectDN=subjectDN+string[13]+"，";

}

}

该函数通过if语句，判断基本信息栏的复选框是否被选中，若选中，则获取用户输入的信息，并存储在字符串类型的subjectDN中，作为证书的DN信息。

**3.获取并设置用户输入的证书有效日期**

public int getDay(){

int a;

String text;

text=jTextDAY.getText();

a= Integer.valueOf(text);

return a;

}

该函数获取用户输入的证书有效天数

public static Date addDate(Date d,long day) {

long time = d.getTime();

day = day\*24\*60\*60\*1000;

time+=day;

return new Date(time);

}

程序默认用户制作证书的时间为证书有效的起始日期，该函数通过用户输入的证书有效天数与有效起始日期，设置证书有效的截止日期。

**4.密钥对生成函数**

public KeyPair generateKeyPair(int seed) throws NoSuchAlgorithmException {

KeyPairGenerator kpg = KeyPairGenerator.getInstance("RSA");

length=Integer.parseInt(keyLength.getSelectedItem().toString());

kpg.initialize(length, new SecureRandom(new byte[seed]));

KeyPair keyPair = kpg.generateKeyPair();

return keyPair;

}

该函数先用"RSA"加密算法创建KeyPairGenerator（生成密钥对）单类实例，若不存在，则重新创建实例，存在调出则原有实例，利用一个小函数得到用户输入的密钥长度，用密钥长度以及长为seed长度的字节生成强随机数序列，去初始化密钥对生成器，keyPair保存生成的密钥对

**5.证书生成函数**

public X509Certificate generateCert(KeyPair keyPair\_root,KeyPair keyPair\_user) {

X509V3CertificateGenerator certGen = new X509V3CertificateGenerator(); X509Certificate cert = null;

certGen.setSerialNumber(new BigInteger(serialnum));

if(jRBROOT.isSelected())

certGen.setIssuerDN(new X509Name(subjectDN));

else

certGen.setIssuerDN(new X509Name("CN=Miss, OU=Network Security, O=OUC, C=China"));

Date curday=new Date();

long num=getDay();

Date effday=addDate(curday,num);

certGen.setNotBefore(curday);

certGen.setNotAfter(effday);

subjectDN =subjectDN.substring(0,subjectDN.length()-1);

String newStr = new String(subjectDN.getBytes(), "UTF-8");

certGen.setSubjectDN(new X509Name(newStr));

certGen.setPublicKey(keyPair\_user.getPublic()); certGen.setSignatureAlgorithm("SHA1WithRSA");

cert = certGen.generateX509Certificate(keyPair\_root.getPrivate(), "BC");

return cert;

}

该函数先实例化生成证书对象，声明一个X509证书对象，并设置为空，利用实例化的生成证书对象，设置该X509证书的序列号，有效日期，以及证书的签名算法，此函数根据用户选择的是根证书还是自定义证书，分别设置证书颁发者的DN信息，并将签名和证书持有者的公钥绑定，并返回生成的证书

**6.数证制作函数**

private void btnMakeMouseClicked(java.awt.event.MouseEvent evt) {

getStringInfo(info);

KeyPair keyPair\_root = null;

KeyPair keyPair\_user = null;

X509Certificate cert=null;

keyPair\_root = generateKeyPair(10); //10作为种子传递

//根证书

if(jRBROOT.isSelected()){

cert = generateCert(keyPair\_root,keyPair\_root);

jTextAreaShow.setText("加载内部证书...\n" + "生成证书成功");

}

//签发证书

else{

keyPair\_user = generateKeyPair(100);

cert = generateCert(keyPair\_root,keyPair\_user);

jTextAreaShow.setText("加载内部证书...\n" + "生成证书成功");

}

String certPath = path+"My"+".cer";

FileOutputStream fos = null;

fos = new FileOutputStream(certPath);

fos.write(cert.getEncoded());//DER编码

fos.close();

}

该函数在用户点击数证制作按钮后，调用密钥对生成函数和证书生成函数，并设置证书存储路径，实现证书的最终生成和存储。

# 遇到的问题及解决办法

## 7.1路径问题

### 7.1.1路径读取问题

在完成界面的设计后，我们首先实现鼠标点击文件选择器以实现正确选取证书存储路径的这一功能，NetBeans关于文件选择器的教程只是演示如何通过文件选择器实现文件的打开和关闭，我们的目的是获得选择文件的绝对路径。关键词搜索“获取文件选择器的”的绝对路径，我们得到了如下语句

filechooser.setFileSelectionMode(JFileChooser.DIRECTORIES\_ONLY);

//此语句规定文件选择器的属性为存储

int result = filechooser.showOpenDialog(this);

if (result == JFileChooser.APPROVE\_OPTION)

path=filechooser.getSelectedFile().getAbsolutePath();

//此语句获得绝对路径给全局变量path。

最初调试时，证书的相关信息作为常量直接传入相关制作证书函数，没有实现读取界面用户输入信息，以测试文件选择器。多次测试，发现用文件输出流写入文件时，会出现存储失败。经过仔细检查，发现在数证制作函数里获取绝对路径时并没有使用全局变量path,由于疏忽，另设一个变量获得存储路径显示区域的内容并作为路径传递给文件输出流对象。显示区域的路径只是一串字符串，不同于据对路径，因此无法实现证书的存储。

### 7.1.2路径存储问题

选择生成CA证书存放位置时，需选择路径。Java语言有专门的界面JFileChooser，通过常规设置，即可获得选择的路径。但测试中发现只单纯生成文件（如下图），并非正确的CA证书格式，需在路径和名称后加”.cert”方可。

C:\Users\Administrator\AppData\Roaming\Tencent\Users\919426791\TIM\WinTemp\RichOle\)RFEE[_YZI}M~MB]$1YBCJB.png

## 7.2常用RDN属性简写问题

我们所熟悉的证书DN结构只有CN=名字, OU=次级组织, O=组织, C=国家，L=城市，最初查资料时，资料认为可以自己定义，例如ST=街道，按此模式，



程序运行时显示以上错误，出现不识别的DN属性名称。

想到之前在查阅资料时，见过语句：

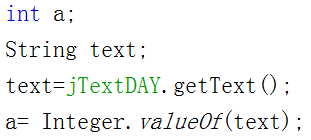
Vector oils=new Vector();

oils.addElement(X509Name.C);

误认为是通过此语句实现扩展DN属性名称。便在程序中使用此语句，因为NetBeans平台具有提示功能，在输入X509Name.时，呈现该平台所能识别的DN属性名称。便以此为依据，进行属性名称。修改后的属性名称依旧无法识别，不同属性名称需用逗号隔开，我们默认为英语逗号，在经过多次调试后，我们尝试着把英文逗号换成中文逗号，发现所有的DN属性均能识别。只是中文逗号是以乱码的形式出现。

## 7.3 int型与String型转换问题

在获得证书有效天数时，由于组件为jText，只能获得String类型的数据，但我们需要long型数据，由于第一次接触java语言，对java的常用类型转换不太熟悉，本以为需要像C/C++一样进行单个字符读取并求得其值，通过查找资料获知java可以直接从String类型直接转为Integer型。



## 7.4密钥长度问题

老师给的参考程序中，密钥长度最小为384，但实验中出现下图错误。

C:\Users\Administrator\AppData\Roaming\Tencent\Users\919426791\TIM\WinTemp\RichOle\DTBAL[EPXX5O_@A[JFQ5UW3.png

密码学课上，RSA的密钥长度最短也为512，所以我们把384的选项删除后均正确。



# 实验不足和需改进的地方

本组实验最大的不足是没能解决编码问题。存储使用者信息的String类对象需包含多个属性，属性从不同的组件中获取，将不同的属性区分开，需用逗号隔开，我们分别测试中文状态和英文状态逗号，发现英文状态下的逗号，出现下图1错误，或有时不将各个属性分开，而只是把整个字符串（包含逗号）看成一个参数。

C:\Users\Administrator\AppData\Roaming\Tencent\Users\919426791\TIM\WinTemp\RichOle\D0Q~8ITGCIL~Q@]{F22F9CS.png

图1

我们使用中文逗号后，虽然可以将各个属性分开，但逗号和中文都属于乱码状态（如图2），我们查阅很多网站，搜集涉及编码和字符串转化等问题，但都没能解决，只能全部属性使用英文，避免乱码。

C:\Users\Administrator\AppData\Roaming\Tencent\Users\919426791\TIM\WinTemp\RichOle\%$GA~OX7W})Z2TRS)L}O4AT.png

图2

由于我们刚接触netbeans和java语言，短时间内没能解决乱码问题，仍需改进，提高实用性。



# 实验感悟

我们利用密码学和java的相关知识进行实践，以小组的形式，学习了制作数字证书的技术。小组形式的实验不仅考察理论知识，对合理分工和合作也有一定的要求，同时，提供了很好的知识交流机会。

在准备实验的过程中，我们对于java语言和可视化界面可以说是零基础，经历了看不懂语句、代码、注释的茫然，只能一句一句的去百度、去翻译。我们上网查找了很多资料，了解到我们这个实验所实现的功能比较基础，我们现在所掌握的知识也知识计算机世界里的冰山一角，在接下来的学习中，还会遇到更难的知识，更陌生的领域。同时，在查看很多源代码和用户手册时，发现大多都是英文版的，我们也感觉到了学好英语的重要性。在之后的学习中，希望在面对困难和失败时，要像在这个实验中的小组一样，相互鼓励，相互帮助，迎接并战胜所有挑战。

将以前学习的密码学知识应用到实际，对密码学和数字证书又有了新的认识。不仅能加深理解和巩固课堂上老师所学的知识，而且还能了解java和利用netbeans制作可视化界面的基本知识，为今后学习后续课程打好良好的基础。

# 分工

|  |  |  |
| --- | --- | --- |
| 15020032003 | 葛畅 | 搜集资料，在eclipse中测试生成证书，后期实现证书序列号实时更新以及数字证书的存储 |
| 15020032011 | 刘子悦 | 搜集资料，利用Netbeans画出界面，并实现了证书类型的选择以及程序运行过程的监控 |
| 15020032030 | 张雪冰 | 搜集资料，从界面中读取用户输入的内容,实现了将代码和界面结合起来，生成证书 |