比亚迪培训讲义
在 J2EE 中使用数字证书

深圳市金蝶中间件有限公司
2007 年 12 月

Table of Contents

1 数字证书使用的场合

- 加密传输
- 机器比较固定
- 使用 USB Key

2 数字证书的颁发

2.1 概述

数字证书是一个经证书授权中心数字签名的包含公开密钥拥有者信息以及公开密钥的文件。证书授权中心（CA）对证书的数字签名过程即为证书的颁发过程。

CA 非常重要！企业范围内，建议自建 CA，或者采用可信任的证书颁发机构的 ROOT CA。

下面介绍使用 Openssl（http://www.openssl.org）结合 JDK 自带的 keytool 工具来产生 CA 证书、服务器证书以及可导入浏览器的 PKCS#12 格式个人证书。

2.2 创建自签名 CA

设置系统环境变量 Path 指向 Openssl 的 bin 目录
建立工作目录：

mkdir ca
cd ca

2.2.1 生成 ca 私钥

openssl dsaparam -out dsaparam 1024

openssl gendsa -out dsakey dsaparam

```
D:\server\Apache2.2\bin\CA>openssl gendsa -out dsakey dsaparam
Loading 'screen' into random state - done
Generating DSA key, 1024 bits
```

2.2.2 生成 ca 待签名证书

openssl req -new -out ca-req.csr -key dsakey -config ..\openssl.cnf

```
D:\server\Apache2.2\bin\CA>openssl req -new -out ca-req.csr -key dsakey -config
..\openssl.cnf
Loading 'screen' into random state - done
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) []:cn
State or Province Name (full name) []:guangdong
Locality Name (eg, city) []:shenzhen
Organization Name (eg, company) []:apusic
Organizational Unit Name (eg, section) []:support
Common Name (eg, your websites domain name) []:sam
Email Address []:sam@apusic.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
```

2.2.3 用 CA 私钥进行自签名，得到自签名的 CA 根证书

openssl x509 -req -in ca-req.csr -out ca-cert.cer -signkey dsakey -days 365

```
D:\server\Apache2.2\bin\CA>openssl x509 -req -in ca-req.csr -out ca-cert.cer -si
gnkey dsakey -days 365
Loading 'screen' into random state - done
Signature ok
subject=/C=cn/ST=guangdong/L=shenzhen/O=apusic/OU=support/CN=sam/emailAddress=sa
m@apusic.com
Getting Private key
```

至此，自签名 CA 根证书制作完毕。当前目录下将产生四个文件，分别是：

- ca-cert.cer
- ca-req.csr

- dsakey
- dsaparam

2.2.4 openssl.cnf 配置文档

```
#
# SSLeay example configuration file.
# This is mostly being used for generation of certificate requests.
#

RANDFILE              = .rnd

####################################################################
[ ca ]
default_ca      = CA_default            # The default ca section

####################################################################
[ CA_default ]

dir = ssl
certs = $dir\certs
crl_dir = $dir\crl
database = $dir\index.txt
new_certs_dir = $dir
certificate = $dir\cacert.pem
serial = $dir\serial
crl = $dir\crl.pem
private_key = $dir\privkey.pem
RANDFILE = $dir\privkey.rnd

# For the CA policy
[ policy_match ]
countryName             = optional
```

```
stateOrProvinceName= optional
organizationName      = optional
organizationalUnitName      = optional
commonName            = supplied
emailAddress          = optional

# For the 'anything' policy
# At this point in time, you must list all acceptable 'object'
# types.
[ policy_anything ]
countryName           = optional
stateOrProvinceName= optional
localityName          = optional
organizationName      = optional
organizationalUnitName      = optional
commonName            = supplied
emailAddress          = optional

####################################################################
[ req ]
default_bits          = 1024
default_keyfile       = privkey.pem
distinguished_name    = req_distinguished_name
attributes            = req_attributes

[ req_distinguished_name ]
countryName                   = Country Name (2 letter code)
countryName_min                       = 2
countryName_max                       = 2

stateOrProvinceName           = State or Province Name (full name)

localityName                  = Locality Name (eg, city)

0.organizationName            = Organization Name (eg, company)

organizationalUnitName                = Organizational Unit Name (eg, section)

commonName                    = Common Name (eg, your website's domain name)
commonName_max                        = 64

emailAddress                  = Email Address
emailAddress_max              = 40

[ req_attributes ]
challengePassword             = A challenge password
challengePassword_min                 = 4
challengePassword_max                 = 20
```

```
[ x509v3_extensions ]

# under ASN.1, the 0 bit would be encoded as 80
nsCertType                      = 0x40

#nsBaseUrl
#nsRevocationUrl
#nsRenewalUrl
#nsCaPolicyUrl
#nsSslServerName
#nsCertSequence
#nsCertExt
#nsDataType
```

2.3 颁发服务器证书

服务器端证书用来向客户端证明服务器的身份，也就是说在 SSL 协议握手的时候，服务器发给客户端的证书。生成服务器证书时用到了 JDK 的密钥管理工具 Keytool.

建立工作目录：

cd ..
mkdir server
cd server

2.3.1 生成服务器私钥对及自签名证书

生成的文件将保存在密钥库 mykeystore 中。

keytool -genkey -alias myserver -keyalg RSA -keysize 1024 -keypass keypass -storepass keypass -dname "cn=localhost, ou=support, o=apusic, l=shenzhen, st=guangdong, c=CN" -keystore mykeystore

```
D:\server\Apache2.2\bin\server>keytool -genkey -alias myserver -keyalg RSA -keys
ize 1024 -keypass keypass -storepass keypass -dname "cn=localhost, ou=support, o
=apusic, l=shenzhen, st=guangdong, c=CN" -keystore mykeystore
```

2.3.2 生成服务器待签名证书

keytool -certreq -alias myserver -sigalg SHA1withRSA -file server.csr -keypass keypass -storepass keypass -keystore mykeystore

```
D:\server\Apache2.2\bin\server>keytool -certreq -alias myserver -sigalg SHA1with
RSA -file server.csr -keypass keypass -storepass keypass -keystore mykeystore
```

2.3.3 请求 CA 签名服务器待签名证书，得到经 CA 签名的服务器证书

openssl x509 -req -in server.csr -out server-cert.cer -CA ..\ca\ca-cert.cer -CAkey ..\ca\dsakey -days 365 -set_serial 02

```
D:\server\Apache2.2\bin\server>openssl x509 -req -in server.csr -out server-cert
.cer -CA ..\ca\ca-cert.cer -CAkey ..\ca\dsakey -days 365 -set_serial 02
Loading 'screen' into random state - done
Signature ok
subject=/C=CN/ST=guangdong/L=shenzhen/O=apusic/OU=support/CN=localhost
Getting CA Private Key
```

2.3.4 把 CA 根证书导入密钥库 mykeystore

keytool -import -alias caroot -file ..\ca\ca-cert.cer -noprompt -keypass keypass -storepass keypass -keystore mykeystore

```
D:\server\Apache2.2\bin\server>keytool -import -alias caroot -file ..\ca\ca-cert
.cer -noprompt -keypass keypass -storepass keypass -keystore mykeystore
认证已添加至keystore中
```

2.3.5 把经过 CA 签名的服务器证书导入密钥库 mykeystore

keytool -import -alias myserver -file server-cert.cer -noprompt -keypass keypass -storepass keypass -keystore mykeystore

```
D:\server\Apache2.2\bin\server>keytool -import -alias myserver -file server-cert
.cer -noprompt -keypass keypass -storepass keypass -keystore mykeystore
认证回复已安装在 keystore中
```

至此，服务器证书已经生成，并且已经将服务器证书、密钥、根证书导入到了密钥库 mykeystore。本环节产生三个文件：

- mykeystore
- server.csr

- server-cert.cer

```
D:\server\Apache2.2\bin\server>dir
 驱动器 D 中的卷是 program
 卷的序列号是 5C1E-60E7

 D:\server\Apache2.2\bin\server 的目录

2007-12-27  15:52    <DIR>          .
2007-12-27  15:52    <DIR>          ..
2007-12-27  15:54             3,001 mykeystore
2007-12-27  15:52               761 server-cert.cer
2007-12-27  15:50               672 server.csr
               3 个文件          4,434 字节
               2 个目录  1,405,890,560 可用字节
```

2.4 颁发客户端证书

个人证书用来向服务器证明个人的身份，也就是说在 SSL 协议握手的时候，客户端发给服务器端的证书。同时个人证书中包含个人信息如用户名等，如果需要，这个用户名将作为登录服务器的用户名。

建立工作目录：

cd ..
mkdir client
cd client

2.4.1 生成客户端私钥

openssl genrsa -out clientkey 1024

```
D:\server\Apache2.2\bin\client>openssl genrsa -out clientkey 1024
Loading 'screen' into random state - done
Generating RSA private key, 1024 bit long modulus
.................................++++++
.....++++++
e is 65537 (0x10001)
```

2.4.2 生成客户端待签名证书

openssl req -new -out client.csr -key clientkey -config ..\openssl.cnf

```
D:\server\Apache2.2\bin\client>openssl req -new -out client.csr -key clientkey -
config ..\openssl.cnf
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) []:cn
State or Province Name (full name) []:guangdong
Locality Name (eg, city) []:shenzhen
Organization Name (eg, company) []:apusic
Organizational Unit Name (eg, section) []:support
Common Name (eg, your websites domain name) []:liweibin
Email Address []:sam@apusic.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
```

2.4.3 请求 CA 签名客户端待签名证书，得到经 CA 签名的客户端证书

openssl x509 -req -in client.csr -out client.cer -CA ..\ca\ca-cert.cer -CAkey ..\ca\dsakey -days 365 -set_serial 02

```
D:\server\Apache2.2\bin\client>openssl x509 -req -in client.csr -out client.cer
-CA ..\ca\ca-cert.cer -CAkey ..\ca\dsakey -days 365 -set_serial 02
Loading 'screen' into random state - done
Signature ok
subject=/C=cn/ST=guangdong/L=shenzhen/O=apusic/OU=support/CN=liweibin/emailAddre
ss=sam@apusic.com
Getting CA Private Key
```

生成客户端的个人证书 client.p12

openssl pkcs12 -export -clcerts -in client.cer -inkey clientkey -out client.p12

```
D:\server\Apache2.2\bin\client>openssl pkcs12 -export -clcerts -in client.cer -i
nkey clientkey -out client.p12
Loading 'screen' into random state - done
Enter Export Password:
Verifying - Enter Export Password:
```

至此，个人证书已经制作完毕。本环节产生四个文件，分别是：

- clientkey
- client.csr
- client.cer
- client.p12

2.5 CA 根证书导入客户端

在这里 CA 的根证书用来在 SSL 握手时验证服务器发给客户端浏览器的证书。如果没有此证书，浏览器将无法自动验证服务器证书，因此浏览器将弹出确认信息，让用户来确认是否信任服务器证书。 在客户端的 IE 中使用"工具 -> Internet 选项 -> 内容 -> 证书 -> 导入"把我们生成的 CA 根证书 ca\ca-cert.cer 导入，使其成为用户信任的 CA。

2.6 个人证书导入客户端

在客户端的 IE 中使用"工具 -> Internet 选项 -> 内容 -> 证书 -> 导入"把我们生成的 CA 根证书 client.p12 导入，使其成为用户信任的 CA。

3 在 J2EE 中使用证书

3.1 配置 SSL 双向认证

Apusic 应用服务器默认配置下不支持双向认证，要支持 SSL 双向认证，需要对配置作如下修改：

3.1.1 服务器端密钥库和信任库

在本文中密钥库和信任库放在一起，都放在 server\mykeystore 中，在前面已经创建了 mykeystore，并且把服务器证书和 CA 根证书导入了 mykeystore。现在需要把 mykeystore 拷贝到 DOMAIN_HOME\config 目录下。

copy server\mykeystore %DOMAIN_HOME%\config

注意：使用 Apusic 4.0.3 时，这里的 DOMAIN_HOME 表示 Apusic 安装目录；使用 Apusic 5.0 + 时，表示 Apusic 安装目录下 domains 目录中当前所使用到的 domain 目录。

3.1.2 修改 Muxer 服务

打开 DOMAIN_HOME\config\apusic.conf 文件，默认 Muxer 服务的配置如下：

```
<SERVICE
    CLASS="com.apusic.net.Muxer"
    >
    <ATTRIBUTE NAME="Port" VALUE="6888"/>
    <ATTRIBUTE NAME="Backlog" VALUE="50"/>
    <ATTRIBUTE NAME="Timeout" VALUE="300"/>
    <ATTRIBUTE NAME="SSLEnabled" VALUE="True"/>
    <ATTRIBUTE NAME="SecurePort" VALUE="6889"/>
    <ATTRIBUTE NAME="KeyStore" VALUE="config/sslserver"/>
    <ATTRIBUTE NAME="KeyPassword" VALUE="keypass"/>
</SERVICE>
```

改成如下：

```
<SERVICE
    CLASS="com.apusic.net.Muxer"
    >
    <ATTRIBUTE NAME="Port" VALUE="6888"/>
    <ATTRIBUTE NAME="Backlog" VALUE="50"/>
    <ATTRIBUTE NAME="Timeout" VALUE="300"/>
    <ATTRIBUTE NAME="MaxWaitingClients" VALUE="500"/>
    <ATTRIBUTE NAME="WaitingClientTimeout" VALUE="5"/>
    <ATTRIBUTE NAME="SSLEnabled" VALUE="True"/>
    <ATTRIBUTE NAME="SecurePort" VALUE="6889"/>
    <ATTRIBUTE NAME="KeyStore" VALUE="config/mykeystore"/>
```

```
    <ATTRIBUTE NAME="KeyPassword" VALUE="keypass"/>

    <ATTRIBUTE NAME="MutualAuthPort" VALUE="443"/>
    <ATTRIBUTE NAME="NeedClientAuth" VALUE="True"/>
    <ATTRIBUTE NAME="TrustStore" VALUE="config/mykeystore"/>
    <ATTRIBUTE NAME="TrustStorePassword" VALUE="keypass"/>
    <ATTRIBUTE NAME="TrustStoreType" VALUE="JKS"/>
</SERVICE>
```

3.1.3 修改 SecurityService 服务

打开 DOMAIN_HOME\config\apusic.conf 文件，默认 SecurityService 服务的配置如下：

```
<SERVICE
    CLASS="com.apusic.security.SecurityService"
    >
</SERVICE>
改成如下：
<SERVICE
    CLASS="com.apusic.security.SecurityService"
    >
    <ATTRIBUTE NAME="ClientRootCA" VALUE="config/mykeystore"/>
</SERVICE>
```

3.2 在程序中获取证书信息

在 JSP 页面中，访问证书很简单，代码样例如下：

```
X509Certificate[]
certs=(X509Certificate[])request.getAttribute("javax.servlet.request.X509Certificate");
X500Principal cert = certs[0].getIssuerX500Principal();
String cn = cert.getName("CN");
```

可以获得的证书相关属性值包括：CN、L、ST、O、OU、C、STREET、DC 和 UID 等。进一步的详细使用方法请参考 JDK API（http://gceclub.sun.com.cn/Java_Docs/jdk6/html/zh_CN/api/javax/security/auth/x500/X500Principal.html）。

4 练习

1 使用 OpenSSL 创建一个自签名证书

2 创建并颁发一个服务器证书

3 创建并颁发一个客户端证书

4 编写一个通过数字证书进行登录的应用

5 附录

5.1 SSL v3 的处理步骤

SSL status: "SSLv3 write client hello A"
SSL status: "SSLv3 read server hello A"
SSL status: "SSLv3 read server certificate A"
SSL status: "SSLv3 read server key exchange A"
SSL status: "SSLv3 read server done A"
SSL status: "SSLv3 write client key exchange A"
SSL status: "SSLv3 write change cipher spec A"
SSL status: "SSLv3 write finished A"
SSL status: "SSLv3 flush data"
SSL status: "SSLv3 read finished A"

5.2 命令行调试 SSL 证书

openssl s_client -state -connect localhost:443 -cert client\client.cer -key client\clientkey -CAfile CA\ca-cert.cer