Universidad Carlos III de Madrid

University Carlos III of Madrid

Tools for working with digital certificates

Secure e-commerce
OpenCourseWare

José María de Fuentes, Ana Isabel González-Tablas, Arturo Ribagorda

Secure e-commerce. Working with digital certificates

Work description

Public key certificates will be required in the implementation of the practice work. This document should help the students, allowing them to use two PKI-related tools.

Some use guides are provided within this document. However, students must employ the user guides and official manuals of the selected tools for solving further questions.

Keytool. Introduction

Keytool is a software tool to manage public key certificates and private keys. It allows managing keystores of private keys and others with the public key certificates from trusted authorities and other entities.

Keytool its included within regular Java distributions, and can be executed from the command line: `keytool [command]`

OpenSSL. Introduction

OpenSSL allows creating a Public Key Infrastructure (PKI), that is, a Certification Authority (CA) to manage all public key certificates for different entities.
Source code and documentation: http://www.openssl.org/
Win32 binary: http://www.slproweb.com/products/Win32OpenSSL.html

Questions that should be answered to assure that global knowledge has been got from this practice work.

About RSA public/private key pair generation

- Why is the keystore password-protected?
- Are you sure you understand all fields of a X.509 public key certifícate?
- Where is the private key stored?
- How many different entries are allowed by any keystore? What does each one mean?

About certificate importing

- Once a certifícate has been imported, what kind of keystore entry is?

About the certification process

- What is a certificate request?
- Why do we have to use a password in this process?
- How is this certification process completed?

About this tools use in this practice

- Which certificates and keys does each entity require to establish a secure connection with other entities?

Exercise: PKI creation and keytool integration

The steps to create a PKI with OpenSSL, and to integrate the results with Keytool, are explained below. Execute the indicated commands and check that everything runs without problems. Should any command results in errors, please refer to the user guide.

Note: You must assure that you understand what is going on at each step.

1. **Create the Certification Authority with OpenSSL:**

Create a folder and copy the openssl.cnf of our Operating System in that folder. Edit that file with these data:

```
[req ]
default_bits    = 2048
default_keyfile = private/cakey.pem
default_md      = md5
prompt          = no
distinguished_name = root_ca_distinguished_name
x509_extensions = root_ca_extensions
[ root_ca_distinguished_name ]
commonName      = CA_Ejemplo
stateOrProvinceName = Madrid
countryName     = ES
emailAddress    = prueba@sce.com
organizationName = SCE_Ejemplo
[ root_ca_extensions ]
basicConstraints = CA:true
[ ca ]
default_ca      = CA_default
[ CA_default ]
dir             = .
certificate     = $dir/cacert.pem
database        = $dir/index.txt
new_certs_dir   = $dir/newcerts
private_key     = $dir/private/cakey.pem
serial          = $dir/serial
default_crl_days= 30
default_days    = 365
default_md      = md5
policy          = policy_anything
x509_extensions = certificate_extensions

[ policy_anything ]
commonName              = supplied
stateOrProvinceName     = supplied
countryName             = supplied
emailAddress            = optional
organizationName        = supplied
organizationalUnitName  = optional

[ certificate_extensions ]
basicConstraints=CA:FALSE
```

Once done, create the following folders and files and create the CA public key certificate:

```
$ mkdir certs private newcerts crl
$ touch index.txt
$ echo '01' > serial
$ openssl req -config ./openssl2.cnf -x509 -newkey rsa:2048 -out
cacert.pem -outform PEM
Generating a 2048 bit RSA private key
..............................+++
.............................................+++
writing new private key to 'private/privkey.pem'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
$
```

2. Create the keypair and a self-signed certificate with keytool:

```
$ keytool -genkeypair -alias sceuser -keystore /tmp/almacen
Write the keystore password:  passPrueba
ÂWrite your name and surname
  [Unknown]:  SCE
Name of your organization unit
  [Unknown]:  SCE
Name of your organization
  [Unknown]:  SCE
Name of your city or village
  [Unknown]:  Madrid
Name of your province or county
  [Unknown]:  Madrid
Code (2 characters) of your country
  [Unknown]:  ES
Is it correct: CN=SCE, OU=SCE, O=SCE, L=Madrid, ST=Madrid,
C=ES?
  [no]:  yes

Write password for <mykey>
        (INTRO if it is the same as that of the keystore):
```

3. Create the certification request for that certificate:

```
$ keytool -keystore /tmp/almacen -certreq -file peticion.csr
Write the keystore password:  passPrueba
```

4. Sign that certification request with OpenSSL:

```
$ openssl ca -config openssl2.cnf -in peticion.csr -out firmado.pem
Using configuration from openssl2.cnf
Enter pass phrase for ./private/cakey.pem:
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
countryName           :PRINTABLE:'ES'
stateOrProvinceName   :PRINTABLE:'Madrid'
localityName          :PRINTABLE:'Madrid'
organizationName      :PRINTABLE:'SCE'
organizationalUnitName:PRINTABLE:'SCE'
commonName            :PRINTABLE:'SCE'
Certificate is to be certified until Dec 21 07:32:17 2009
GMT (365 days)
Sign the certificate? [y/n]:y
1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
```

5. Convert that certificate from PEM format to DER:

```
$ openssl x509 -in firmado.pem -out firmado.cer
```

6. Import the CA´s certificate into the keystore:

```
$ keytool -keystore /tmp/almacen -alias openssl-ca -import -file cacert.pem
Write the keystore password:  passPrueba
Owner: O=Ejemplo_SCE, EMAILADDRESS=prueba@sce.com,
C=ES, ST=Madrid, CN=CA_Ejemplo
Issuer: O=Ejemplo_SCE, EMAILADDRESS=prueba@sce.com, C=ES,
ST=Madrid, CN=CA_Ejemplo
Serial number: ae167491dfc4cb6b
Valid from: Tue Dec 21 08:15:46 CET 2010 to: Thu Jan
20 08:15:46 CET 2011
Certificate hashes:
         MD5:
9D:9C:CB:5C:C1:66:1A:B2:1B:3D:EF:DE:B2:19:B5:DB
         SHA1:
8C:9E:C6:F1:83:E5:84:E2:86:EB:87:F7:27:6D:A7:AC:42:98:12:B0
ÂTrust this ertificate? [no]:  yes
The certificate has been imported into the keystore
$
```

7. Import the signed certificate into the keystore:

```
$ keytool -keystore /tmp/almacen -import -file firmado.cer
Write the keystore password:  passPrueba
The certificate has been imported into the keystore
$
```

8. Check that both certificates have been correctely installed:

```
$ keytool -list -keystore /tmp/almacen
Write the keystore password:  passPrueba

Keystore type: jks
Keystore provider: SUN

Keystore contains 2 entries

openssl-ca, 21-dic-2007, trustedCertEntry,
Certificate hash (MD5):
9D:9C:CB:5C:C1:66:1A:B2:1B:3D:EF:DE:B2:19:B5:DB
SCE, 21-dic-2004, keyEntry,
Certificate hash (MD5):
B0:F1:75:AE:C4:FE:2B:0C:C7:E9:E6:3D:C2:7D:1C:D9
$
```

9. From now on, we can export that certificate with the `-export` option of Keytool.