

# Exercise 6

*Creating Keystores for WS-Security*

## **Prior Knowledge**

*Understand Private Key Crypto and Certificates (at a high level)*

## **Objectives**

*Create the keystores we will use for the WS-Security Exercise*

## **Software Requirements**

- Java Development Kit 7

1. Check that the keytool command is working

On a command line type keytool

You should see

```
keytool usage:  
... [LOTS MORE]
```

2. Create a directory (e.g. ~/keys/) and change to that directory



Now let's create a client key (for Signing)

Type:

```
keytool -genkeypair -alias client -storepass  
clientpass -keystore clientkeystore.jks
```

**All on one line!**

You will be prompted as follows:

```
What is your first and last name?  
[Unknown]: Oxford Student  
What is the name of your organizational unit?  
[Unknown]: DCS  
What is the name of your organization?  
[Unknown]: Oxford University  
What is the name of your City or Locality?  
[Unknown]: Oxford  
What is the name of your State or Province?  
[Unknown]: Oxon  
What is the two-letter country code for this unit?  
[Unknown]: GB  
Is CN=Oxford Student, OU=DCS, O=Oxford University, L=Oxford,  
ST=Oxon, C=GB correct?  
[no]: yes  
  
Enter key password for <client>  
(RETURN if same as keystore password):
```

Press enter to use the same as the keystore password ("clientpass")

You don't have to use my details!

3. Now let's create a server keystore (for encryption):

```
keytool -genkeypair -alias server -keyalg RSA \  
-keystore serverkeystore.jks \  
-storepass serverpass
```

4. Once again fill in the details (this time in a more "server-ish" way perhaps?)
5. Now we need to get these two keystores to trust each other (since there is no uber-CA). Export the client certificate.

```
keytool -export -alias client -keystore clientkeystore.jks \  
-file client.cert  
Enter keystore password: [clientpass]  
Certificate stored in file <client.cert>
```



6. Now import into the server keystore:

```
keytool -import -file client.cert -keystore serverkeystore.jks \  
-alias client  
Enter keystore password:  
Owner: CN=Oxford Student, OU=DCS, O=Oxford University, L=Oxford,  
ST=Oxon, C=GB  
Issuer: CN=Oxford Student, OU=DCS, O=Oxford University, L=Oxford,  
ST=Oxon, C=GB  
Serial number: 5379flec  
Valid from: Mon Nov 24 09:21:56 GMT 2014 until: Sun Feb 22  
09:21:56 GMT 2015  
Certificate fingerprints:  
    MD5: 6B:75:0E:B5:47:3B:66:BB:6D:F9:F9:ED:0B:26:CB:71  
    SHA1:  
C1:F1:CA:86:FE:CF:D1:7A:92:76:F9:16:AB:C8:2C:B0:D5:A8:0F:05  
    SHA256:  
A5:CA:3C:1E:2A:A8:FE:78:59:B6:4E:88:77:EE:08:C0:B1:7C:5C:2F:F6:7E:  
A4:8B:97:96:2C:62:0F:21:10:93  
    Signature algorithm name: SHA256withRSA  
    Version: 3  
  
Extensions:  
  
#1: ObjectId: 2.5.29.14 Criticality=false  
SubjectKeyIdentifier [  
KeyIdentifier [  
0000: 4D 15 BD FF F5 20 E8 2E    28 5C 21 86 F5 A9 07 8B  M....  
..(\!.....  
0010: 17 62 B7 E2                      .b..  
]  
]
```

Trust this certificate? [no]: yes  
Certificate was added to keystore

7. Do the opposite – export the server’s certificate and import into the client’s keystore.
8. Validate you have successfully done everything by listing the contents of each keystore. For example:

```
keytool -list -keystore serverkeystore.jks  
Enter keystore password:  
  
Keystore type: JKS  
Keystore provider: SUN  
  
Your keystore contains 2 entries  
  
client, 24-Nov-2014, trustedCertEntry,  
Certificate fingerprint (SHA1):  
C1:F1:CA:86:FE:CF:D1:7A:92:76:F9:16:AB:C8:2C:B0:D5:A8:0F:05  
server, 24-Nov-2014, PrivateKeyEntry,  
Certificate fingerprint (SHA1):  
F9:01:03:4D:8F:17:C1:4E:57:C0:89:47:D6:E1:B6:92:66:1F:B7:51
```

**That’s all folks!**

