Technische Hochschule OWL
Dep. of Electrical Engineering and Computer Science
Prof. Dr. Heiss

Network Security (NWS) **Lab 7**

1.) Extend the provided `MiniChat` application by a TLS layer implementation, that allows chat clients to authenticate a chat server, and that provide data confidentiality for the exchanged messages.

2.) Follow the following steps to provide a PKI for the Mini Chat application with the help of the command line utility `keytool`, which you may find in the `bin` directory of your Java installation.

2.1.) CA - Setup (`ca.ks`, `ca.der`)

```
> keytool -genkeypair -keystore ca.ks -storetype pkcs12 -storepass caSecret
   -alias cakey -keyalg RSA -keysize 2048 -validity 3652

> keytool -exportcert -keystore ca.ks -storepass caSecret -alias cakey
   -file ca.der
```

2.2.) Client - Generating a truststore (`clientTrustStore.ks`)

```
> keytool -importcert -noprompt -keystore clientTrustStore.ks
   -storetype pkcs12 -storepass clientSecret -alias ca -file ca.der
```

2.3.) Server - Creating a keystore, a keypair and a CSR (`serverKeyStore.ks`, `server.csr`)

```
> keytool -genkeypair -keystore serverKeyStore.ks -storetype pkcs12
   -storepass serverSecret -alias server -keyalg RSA -keysize 2048
   -validity 365

> keytool -certreq -keystore serverKeyStore.ks -storepass serverSecret
   -alias server -file server.csr
```

2.4.) CA - Server certificate signing (`server.der`)

```
> keytool -gencert -keystore ca.ks -storepass caSecret -alias cakey
   -infile server.csr -outfile server.der
```

2.5.) Server - Importing Certificate into Keystore (`serverKeyStore.ks`)

(i) Import CA-Certificate into Keystore:

```
> keytool -importcert -noprompt -keystore serverKeyStore.ks
   -storepass serverSecret -alias ca -file ca.der
```

(ii) Import Server-Certificate into Keystore:

```
> keytool -importcert -keystore serverKeyStore.ks -storepass serverSecret
   -alias server -file server.der
```

3.) With respect to the TLS enabled `MiniChat` application:

- Determine the cipher suites supported by the client.
- Restrict the client to use TLS v1.3 only.
- Determine the cipher suites supported by the TLS v1.3 only client.

4.) Add client authentication to the `MiniChat` application.

5.) Generate key pairs and certificates for two clients. Furthermore, simulate the revocation of a certificate.

5.1.) Client 1 - Repeat the necessary steps from exercise 2 to generate:

`client1KeyStore.jks`, `client1.csr`, `client1.der`

5.2.) Client 2 - Repeat the necessary steps from exercise 2 to generate:

`client2KeyStore.jks`, `client2.csr`, `client2.der`

5.3.) CA - Revoke Client1's certificate (`ca.crl`)

```
> keytool -gencrl -keystore ca.jks -alias cakey -id <SN of Client1.der>
  -file ca.crl
```

6.) Implement a subclass of the `X509TrustManager` interface that can be used with the `SSLContext` instance in the `MiniChatServer` implementation.

This implementation of the `X509TrustManager` shall provide a debug output of all certificates that are referenced by `checkClientTrusted()` method calls in the first parameter. Furthermore, all implemented methods of the `X509TrustManager` interface shall be delegated to corresponding method calls of of a `X509TrustManager` implementation, that is available from the method call:

`TrustManagerFactory.getInstance(”SunX509”)`

7.) Extend the `X509TrustManager` implementation above, such that the method `checkClientTrusted()` also checks, if client certificates have been revoked (i.e. are contained in a CRL).
(Hint: Use the `CertificateFactory` class, to generate an instance of a `CRL` class from a DER encoded CRL.)