bbw-it wiki

SSL Konfiguration

SSL (oder neu TSL) wird verwendet, um die Verbindung zwischen einem Browser und einem Web-Server zu verschlüsseln. Für die Verschlüsselung wird dabei üblicherweise das RSA-Verfahren angewendet. Allerdings wird nicht die gesamte Kommunikation mit RSA verschlüsselt, sondern es wird während der Verbindungsaufnahme ein symmetrischer Schlüssel ausgehandelt. Dieser sogenannte Handshake ist RSA-verschlüsselt, während für die Kommunikation danach der symmetrische Schlüssel verwendet wird.

Während des Handshakes präsentiert der Web-Server seinen öffentlichen Schlüssel, damit der Browser verschlüsselte Nachrichten schicken kann. Um sicher zu stellen, dass es sich auch um den richtigen Server handelt, ist der öffentliche Schlüssel dabei Teil eines Zertifikats, welches von einer Zertifizierungsstelle unterschrieben ist.

Um die Unterschrift der Zertifizierungsstelle (Certificate Agency, CA) überprüfen zu können, muss deren öffentlicher Schlüssel im Browser in einer Liste von vertrauenswürdigen CAs gespeichert sein.

In diesem Abschnitt sollen Sie den Umgang mit SSL-Zertifikaten lernen.

Für die Signierung Ihrer Zertifikate verwenden Sie die BBW-CA, welche von Ihrer Lehrperson verwaltet wird. Diese ist nicht in den Browsern registriert. Ihre Zertifikate werden deshalb nicht als absolut vertrauenswürdig akzeptiert werden. Der Ablauf, wie Sie zu einem Zertifikat kommen, ist aber dem "richtigen" Ablauf nachempfunden.

Aufgabe 1

Erstellen Sie ein von der BBW-CA signiertes Zertifikat und installieren Sie es im Apache-Server Ihrer xampp-Installation.

Das Zertifikat soll die folgenden Bedingungen erfüllen (ansonsten sind Sie frei):

> Schlüssellänge (RSA): 2048 Bits
> CN: die IP des Computers, auf welchem Sie die Applikation vorführen werden.
> O: Modul 150
> OU: Ihre Initialen

Eine Einführung zu SSL in Apache finden Sie hier.

Stellen Sie für sich selbst sicher, dass Sie die einzelnen Schritte gut verstehen!

Aufgabe 2

Nehmen Sie irgendeine JSF-Applikation auf Tomcat und bringen Sie diese dazu, ein von der bbw-CA signiertes Zertifikat zu verwenden. Sie können das gleiche Zertifikat verwenden, das Sie schon für Apache gebraucht haben. Dieses müssen Sie dann in einen `keystore` importieren (`keytool` und `keystore` kennen Sie ja schon vom Thema *Digitale Signaturen*).

Kurzanleitung

Beachten Sie, dass die beiden Programme `openssl` und `keytool`, welche Sie für den folgenden Ablauf verwenden müssen, nicht im Suchpfad enthalten sind. Sie müssen also jeweils den ganzen Pfad angeben.

1. PKCS12-File mit Key und Zertifikat erstellen

Um einen private Key und ein Zertifikat, die nicht mit dem keytool erstellt wurden, in einen keystore zu importieren, fasst

man am besten diese beiden Files in einem PKCS12-File zusammen:

```
openssl pkcs12 -export -in [certificate file] -inkey [private key file]
               -out [p12 filename].p12 -name [some alias]
```

2. PKCS12-File in keystore umwandeln

```
keytool -importkeystore
        -deststorepass [new keystore password] -destkeypass [new key password]
        -destkeystore [new keystore filename]
        -srckeystore [p12 filename].p12 -srcstoretype PKCS12 -srcstorepass [p12 password]
        -alias [some alias]
```

Es empfiehlt sich, für den keystore und den key dasselbe Passwort zu definieren.

3. CA-Zertifikat in keystore importieren

Laden Sie sich das Zertifikat der bbw-CA, welche ihr Server-Zertifikat signiert hat herunter und importieren Sie dieses in einen neuen keystore.

```
keytool -keystore [keystore filename] -import -alias root -trustcacerts -file bbwca.crt
```

4. Tomcat konfigurieren

Nun muss das Konfigurationsfile von Tomcat (server.xml) angepasst werden. In Netbeans erhalten Sie dazu bei Rechtsklick auf den Server den Menueintrag *Edit server.xml*

In diesem Konfigurationsfile gibt es schon den Eintrag für einen Connector auf Port 443. Allerdings ist er höchstwahrscheinlich auskommentiert.

Dieser Eintrag muss aktiviert werden und es müssen der Pfad und das Passwort für den keystore angegeben werden.

Dann sollte er etwa so aussehen:

```
<Connector
           protocol="org.apache.coyote.http11.Http11NioProtocol"
           port="8443" maxThreads="200"
           scheme="https" secure="true" SSLEnabled="true"
           keystoreFile="[path to keystore file]"  keystorePass="[keystore password]"
           clientAuth="false" sslProtocol="TLS"/>
```

5. SSL obligatorisch machen

Dazu müssen Sie das File web.xml Ihrer Applikation um den folgenden Teil erweitern:

```
    <security-constraint>
        <display-name>Example Security Constraint</display-name>
        <web-resource-collection>
            <web-resource-name>Protected Area</web-resource-name>
            <url-pattern>/*</url-pattern>
        </web-resource-collection>
        <user-data-constraint>
            <transport-guarantee>CONFIDENTIAL</transport-guarantee>
        </user-data-constraint>
    </security-constraint>
```
