

## Practical Exercise: Smartcard-based authentication in HTTP

September 20, 2017

Due date: no date

## Changelog

- v1.0 - Initial Version.

## 1 Introduction

Smartcards can be used to authenticate users and this has been observed for the purpose of authenticating a user logging in to the local machine (with PAM). However, when using web technologies the concept can be easily extended in order to allow authentication of users remotely. By remotely we consider the case where the server providing a web service or web page is not collocated with the client's browser, and the Smartcard is not present to the server.

This laboratory guide will focus in the setup of a secure server, using a custom Certification Authority (CA). By importing the CA public key (self-signed public key certificate), clients can validate the authenticity of the server. At the same time, it can be considered that some contents are restricted to a limited number of individuals, and access to those contents can be controlled by means of a Smartcard.

### IMPORTANT

**This guide assumes that you are using the Virtual Machine provided in the beginning of these classes.** If you are not using it, you will need to install and setup the Portuguese Citizen Card tools, `apache2`, `php`, `xca`, and `qt3`. Because you will be inserting a custom CA in the web browser, **we strongly discourage executing this guide in a real system.**

## 2 Creation of a Certification Authority

Certification Authorities are vital for authentication of services across the Internet. They are considered to be trusted and this trust is inherited by the certificates they sign. Although CAs commonly have global reach and are to be globally trusted, this is not really a requirement and custom CAs can be created. If clients install the custom CA and trust it, all certificates issued by the CA will also be trusted, just like any other commercial CA. This is useful for services which have a limited number of users accessing it.

In this step we will create a custom CA in order to generate signed certificates for a personal web server. For this purpose we will rely on the `XCA` software, which can be easily installed using `apt-get`.

The first step is to launch the `XCA` application and create a new database. Do not forget to specify a password!

Afterwards, generate a new key for the CA, named **CAKey**. Usually, CA keys are considerably stronger than those of server certificates. Consider 4096 bits if possible.

Then create a CA by selecting *Certificates*→*New Certificate*. Do not forget to select the key you just created, as well as the default CA template (you must select it and then choose *Apply all*). Also, you must fill the identification data for the CA and define the validity of the CA (a few years).

After having the CA up and running, we can generate new Certificate Signing Requests (CSR), and sign these request with the CA just created. Go to the tab *Certificate Signing Requests* and click *New Request*. This time, apply the *HTTP Server* template and fill the remaining fields. Use **localhost** as the common name.

With the CSR, the certificate can be finally generated by signing the CSR with the **CAKey**. Right-click on the certificate and select *sign*. Please make sure you sign the CSR with the **CAKey**!

The final step is to export the CA Certificate, and both the Server Certificate and Private Key. Export everything as PEM. Do not export the CA Private Key! This key is secret and should **NEVER** be distributed!



### 3 Installation of a server certificate

Installation of the certificates and keys requires copying the files to specific places. The CA's certificate should go to **/etc/ssl/certs**. The Server certificate and private key should go to **/etc/ssl/private**.

Clients should also install the CA certificate in their keystore. For this purpose, it is simply required to change the extension of the certificate to **|.pem|**, open the CA certificate with a browser, and accept it.

For the server we will be using **apache2**, which must also be installed. Afterwards, the **SSL** module must be enabled by issuing:

```
a2enmod ssl
```

In the folder **/etc/apache2/sites-available** there is a file named **|default-ssl|**. Copy this file to **/etc/apache2/sites-enabled** (or do a symlink). This file defines the SSL configuration the server will use, and must be edited in order to consider the cryptographic material just created. Be sure to modify the following variables:

- **SSLCertificateFile**: This should refer the PEM file containing the server certificate.
- **SSLCertificateKeyFile**: This should refer the PEM file containing the server private key.
- **SSLCACertificatePath**: This should refer **/etc/ssl/certs**.

Edit all required variables and restart the server by issuing:

```
service apache2 restart
```

Verify that the browser can access the servers by typing the URL **https://localhost**.

If everything is correct, there should be no warning or error, and the page should be secure. The practical result is that the server is authenticated and the connection is secure. The browser accepts the server's certificate because it was issued by a trusted CA. If the **common-name** field of the certificate is different from **localhost**, the browser will show a warning message.

### 4 Using Smartcards for authenticating users

The next step is to authenticate the client, and in this particular case, identify the individual user by means of the Portuguese Citizen Card.

The first step is to download the CA and subCA of the Portuguese CC, and install these certificates into the `/etc/ssl/certs/PTEIDCC` directory. The files cannot be copied directly as they are in DER format. Use the following command to convert each file to PEM:

```
openssl x509 -in infile.cer -out outfile.pem -inform DER -outform PEM
```

Then all files must be concatenated to the same file. In this case we will use a file named `PT.pem`, which can be produced by issuing:

```
cat *.pem >/etc/ssl/certs/PTEIDCC/PT.pem
```

The files are available at [https://pki.cartaodecidadao.pt/publico/certificado/cc\\_ec\\_cidadao\\_autenticacao/](https://pki.cartaodecidadao.pt/publico/certificado/cc_ec_cidadao_autenticacao/).


Next proceed to configure the server by editing the file `default-ssl`. In this file, inside the `VirtualHost *:443` section, create a new entry with the following content:

```
<Location /secure>
    SSLVerifyClient require
    SSLVerifyDepth 10
    SSLOptions +OptRenegotiate +StdEnvVars +ExportCertData
    SSLCACertificateFile /etc/ssl/certs/PTEIDCC/PT.pem
</Location>
```



Create the directory `/var/www/secure` and put a file named `index.php` with a success message.

Now try to access the url `https://localhost/secure`. Without a Smartcard installed, the browser should present an error as it is unable to provide a valid certificate for the server. In this case, the server will only allow certificates signed by a certificate present in the `PT.pem` file.

Using Wireshark, examine the packets exchanged between the browser and the server 

In order to successfully access the page you must go to Firefox preferences and choose:

*Advanced*→*Security Devices*→*Load*

Provide any name and specify the value `/usr/local/lib/libpteidpkcs11.so` as the filename. The Smartcard should be available, and if the Smartcard is inserted in the reader, your certificates should be listed when choosing the option *Advanced*→*View Certificates*.

Retry accessing the secure address. Access should be granted.

Using Wireshark inspect the messages that are exchanged between the browser and the server.

An important aspect of authenticating users through a Smartcard is to clearly identify the individual (Portuguese Citizen) at the application layer (e.g., PHP application). For this purpose, edit the `index.php` file that you previously created and add the following content:

```
<html>
  <head><meta charset="UTF-8"></head>
  <body>
    <pre>
<?php print_r($_SERVER); ?>
    </pre>
  </body>
</html>
```



Access the secure web page again and verify the variables that are exposed to the PHP application. It should be noticed that the application can further restrict access to the service by verifying any of these fields. However, it will have the assurance that the user was successfully authenticated by the server.

## 5 References

- Portuguese Citizen Card web site: <http://www.cartaodecidadao.pt>
- Apache2 ModSSL, [http://httpd.apache.org/docs/2.2/mod/mod\\_ssl.html](http://httpd.apache.org/docs/2.2/mod/mod_ssl.html)
- XCA, <http://sourceforge.net/projects/xca>