SUP'COM
Higher School of Communication of Tunis

جامعة قرطاج
Université de Carthage

User Guide

Smart Park

Realized by:

Fatma Krichen & Malek Elmechi

Supervised by:

Dr. Ing. Mohamed-Bécha Kaâniche

Table of Contents

Table of Figures

1- Introduction

This guide outlines the process of installing and configuring Oracle JDK 21, WildFly 34, and securing the application with SSL using Let's Encrypt for the domain smartparkcot.me

2- Java Installation

- **Download and install Oracle JDK 21:**

```
wget https://download.oracle.com/java/21/latest/jdk-21_linux-x64_bin.deb

sudo apt install ./jdk-21_linux-x64_bin.deb

java -version
```

3- WildFly Installation and Configuration:

- **Download the WildFly 34.0.0**

```
wget https://github.com/wildfly/wildfly/releases/download/34.0.0.Final/wildfly-34.0.0.Final.tar.gz
```

- **Extract the contents of the downloaded.tar.gz archive**

```
tar -xvf wildfly-34.0.0.Final.tar.gz
```

- **Move the WildFly directory to the /opt folder for installation**

```
sudo mv ./wildfly-34.0.0.Final /opt

sudo ln -s /opt/wildfly-34.0.0.Final /opt/wildfly
```

- **Create a system user named 'wildfly' with no login access and set the WildFly directory as its home**

```
sudo useradd -r -d /opt/wildfly -s /usr/sbin/nologin wildfly
```

- **Change the ownership of the WildFly directory to the 'wildfly' user**

```
sudo chown -RH wildfly:wildfly wildfly
```

4- WildFly Configuration

- **Create the directory for WildFly system configuration files**

```
sudo mkdir -p /etc/wildfly/
```

- **Copy the WildFly configuration file to the /etc/wildfly/ directory**

```
sudo cp /opt/wildfly/docs/contrib/scripts/systemd/wildfly.conf /etc/wildfly/
```

- **Copy the default standalone.xml configuration file and create a custom configuration for 'smartparkcot.me'**

```
sudo cp /opt/wildfly/standalone/configuration/standalone.xml
/opt/wildfly/standalone/configuration/smartparkcot.me.xml
```

- **Open the wildfly.conf file to edit the WILDFLY_CONFIG variable and set it to 'smartparkcot.me.xml'**

```
sudo vi /etc/wildfly/wildfly.conf
```

- **Copy the launch.sh script to WildFly's bin directory**

```
sudo cp /opt/wildfly/docs/contrib/scripts/systemd/launch.sh /opt/wildfly/bin/
```

- **Copy the WildFly service file to the systemd directory for service management**

```
sudo cp /opt/wildfly/docs/contrib/scripts/systemd/wildfly.service /usr/lib/systemd/system/
```

- **Create the necessary directory for WildFly runtime processes**

```
sudo mkdir /var/run/wildfly/
```

- **Change the ownership of the /var/run/wildfly/ directory to the 'wildfly' user**

```
sudo chown -RH wildfly:wildfly /var/run/wildfly/
```

5- Firewall Configuration

```
sudo ufw allow 80/tcp

sudo ufw allow 443/tcp
```

6- Activating and configuring SSL with Let's Encrypt

- **Use Certbot to generate an SSL certificate for the domain smartparkcot.me and its subdomains using DNS validation:**

```
sudo certbot certonly --manual --preferred-challenges dns --manual-public-ip-logging-ok --must-staple -d "*.smartparkcot.me" -d smartparkcot.me
```

- **Change permissions to allow access to the SSL certificates directory**

```
cd /etc/letsencrypt/

sudo chmod 777 live
```

- **Change to the directory containing the SSL certificates for smartparkcot.me**

```
cd live/smartparkcot.me
```

- **Convert the SSL certificate files to a PKCS12 (.pfx) file for use in Java KeyStores**

```
sudo openssl pkcs12 -export -out certificate.pfx -inkey privkey.pem -in cert.pem -certfile chain.pem
```

- **Import the .pfx certificate into a Java KeyStore (.jks) file using keytool**

```
sudo keytool -importkeystore -srckeystore certificate.pfx -srcstoretype PKCS12 -srcstorepass
'changeit' -storepass 'changeit' -destkeystore smartparkcot.me.jks -deststorepass ‘Password'
```

- **Move the generated .jks keystore to WildFly's configuration directory**

```
sudo mv ./smartparkcot.me.jks /opt/wildfly/standalone/configuration/
```

- **Remove the .pfx file after the import is complete**

```
sudo rm certificate.pfx
```

- **Restrict permissions on the live directory for added security**

```
sudo chmod 700 live
```

- **Open WildFly's configuration file 'standalone.conf' to add a TLS-related option to JAVA_OPTS**

```
cd /opt/wildfly/bin

vi standalone.conf
```

Add to JAVA_OPTS -Djdk.tls.server.enableStatusRequestExtension=true

- **Change the HTTP port from 8080 to 80 and the HTTPS port from 8443 to 443**

```
vi /opt/wildfly/standalone/configuration/ smartparkcot.me.xml
```

(At the end of file change 8080 ==> 80 and 8433 ==> 443)

- **Grant permissions to allow WildFly and Java to bind to privileged ports (80 and 443)**

```
setcap CAP_NET_BIND_SERVICE=+eip /opt/wildfly/bin/standalone.sh

setcap CAP_NET_BIND_SERVICE=+eip /opt/wildfly/bin/launch.sh

setcap CAP_NET_BIND_SERVICE=+eip /usr/lib/jvm/jdk-21/bin/java (check this path before executing
the command)
```

7- Starting and Verifying the WildFly Server

- **Reload the systemd configuration and start WildFly**

```
sudo systemctl daemon-reload
```

- **Start the WildFly service**

```
sudo systemctl start wildfly
```

- **Verify the status of the WildFly service to ensure it is running correctly**

```
sudo systemctl status wildfly
```

8- SSL Configuration in WildFly

- **Configure the KeyStore and SSL context in WildFly using the CLI**
- **Connect to the WildFly CLI as the 'wildfly' user**

```
sudo -u wildfly /opt/wildfly/bin/jboss-cli.sh --connect
```

- **Add the Keystore to WildFly for SSL/TLS configuration**

```
/subsystem=elytron/key-store=smartparkcotKeyStore:add(path=smartparkcot.me.jks,relative-to=jboss.server.config.dir, credential-reference={clear-text="Password"},type=JKS)
```

- **Add the Key Manager to WildFly and associate it with the Keystore**

```
/subsystem=elytron/key-manager=smartparkcotKeyManager:add(key-store=smartparkcotKeyStore,credential-reference={clear-text="Password"})
```

- **Create an SSL context for WildFly with TLSv1.3 and specific cipher suites**

```
/subsystem=elytron/server-ssl-context=smartparkcotTLSContext:add(key-manager=smartparkcotKeyManager,protocols=["TLSv1.3"],cipher-suite-names="TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256:TLS_AES_128_GCM_SHA256")

batch
```

- **Unset the existing security realm for the HTTPS listener**

```
/subsystem=undertow/server=default-server/https-listener=https:undefine-attribute(name=security-realm)
```

- **Assign the newly created SSL context to the HTTPS listener**

```
/subsystem=undertow/server=default-server/https-listener=https:write-attribute(name=ssl-context,value=smartparkcotTLSContext)

run-batch

reload
```

- **Apply the SSL context to the management interface**

```
/core-service=management/management-interface=http-interface:write-attribute(name=ssl-context, value=smartparkcotTLSContext)

reload
```

- **Secure the management interface with HTTPS**

```
/core-service=management/management-interface=http-interface:write-attribute(name=secure-socket-binding, value=management-https)

reload
```

9- SSL Labs Test Result

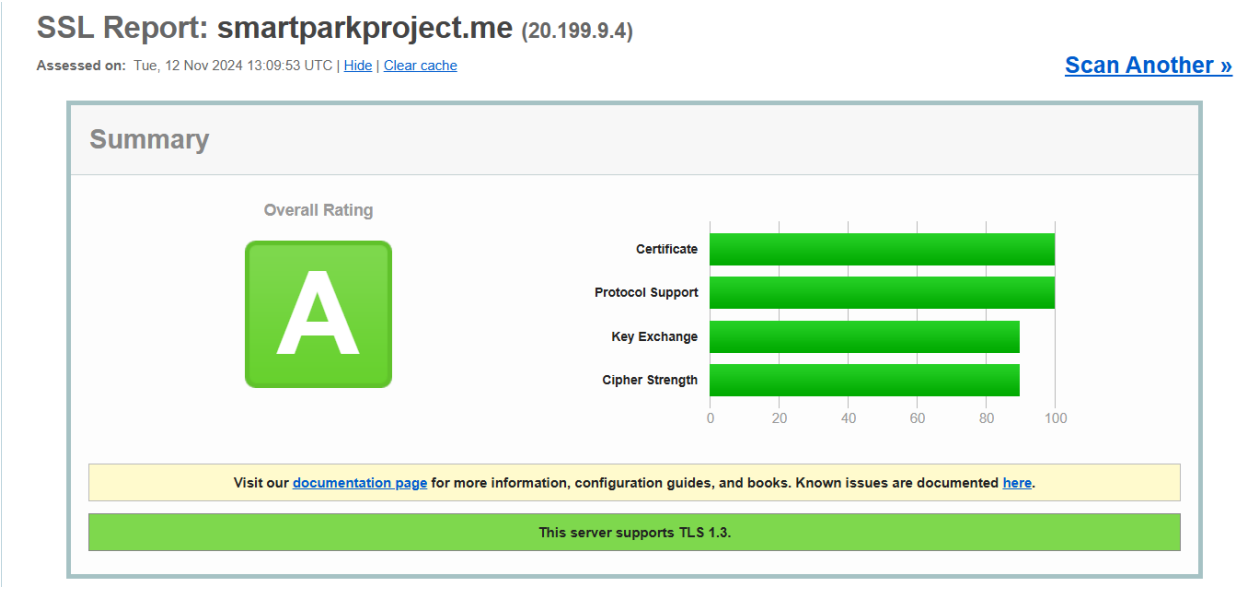
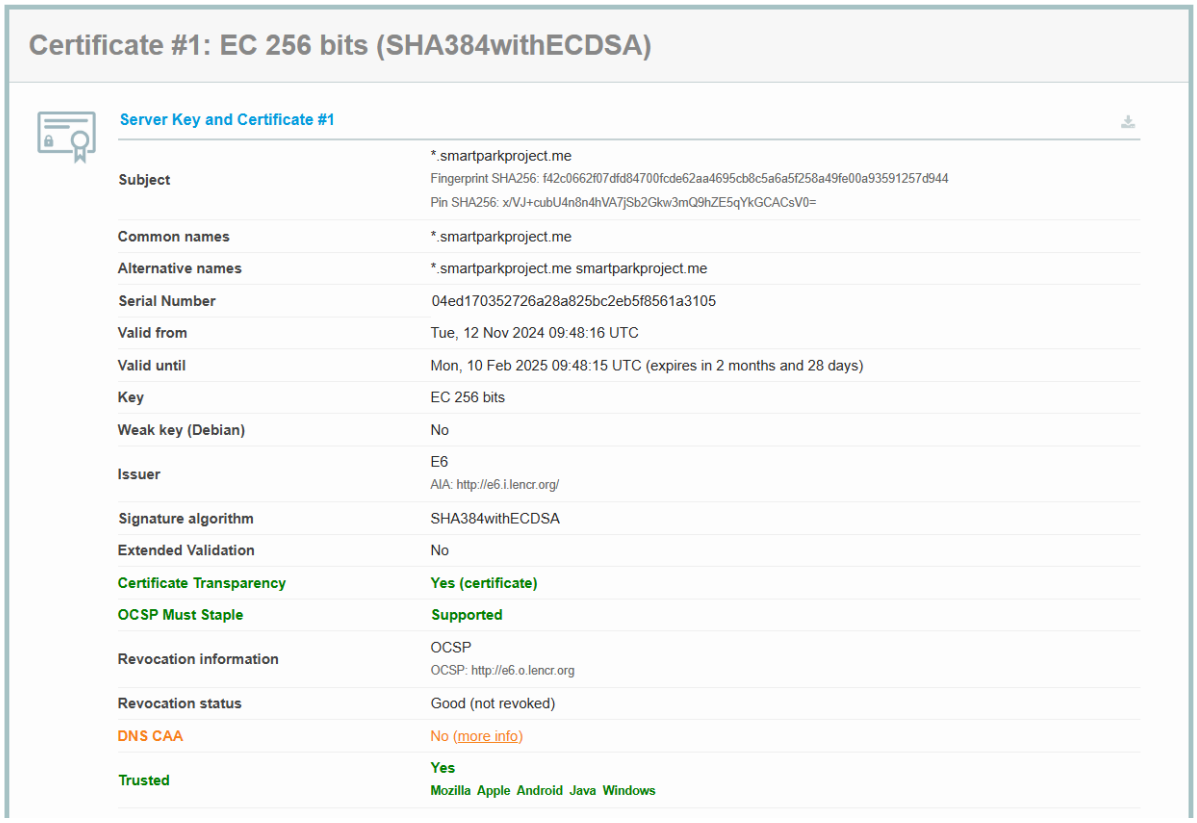


Figure 1 : SSL Report : smartparkproject.me

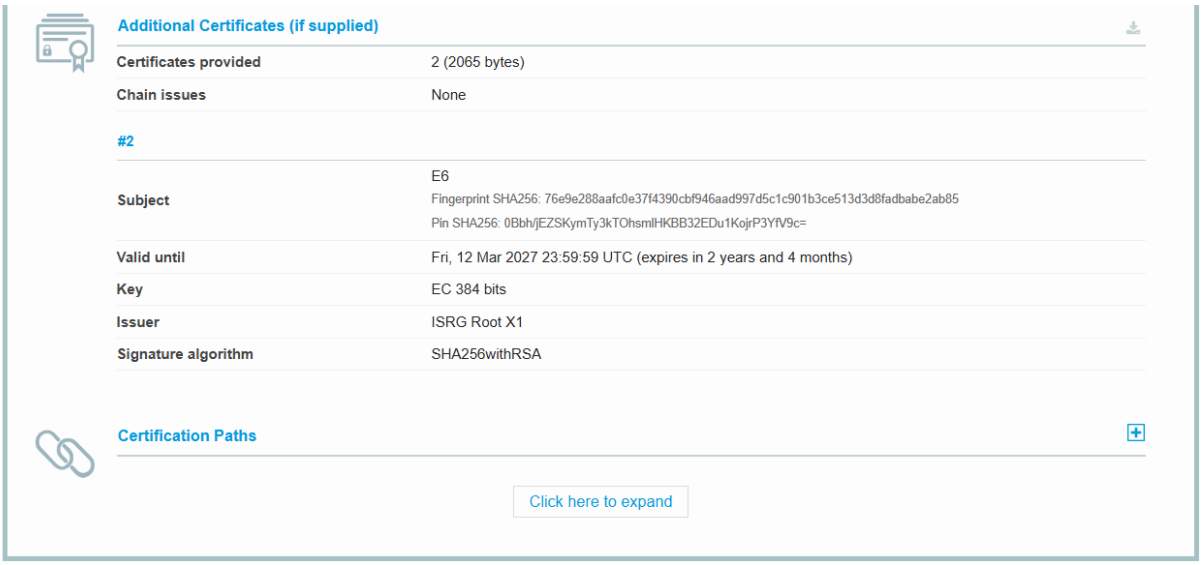


Certificate #1: EC 256 bits (SHA384withECDSA)

Server Key and Certificate #1

Subject	*.smartparkproject.me Fingerprint SHA256: f42c0662f07dfd84700fcde62aa4695cb8c5a6a5f258a49fe00a93591257d944 Pin SHA256: x/VJ+cubU4n8n4hVA7jSb2Gkw3mQ9hZE5qYkGCACsV0=
Common names	*.smartparkproject.me
Alternative names	*.smartparkproject.me smartparkproject.me
Serial Number	04ed170352726a28a825bc2eb5f8561a3105
Valid from	Tue, 12 Nov 2024 09:48:16 UTC
Valid until	Mon, 10 Feb 2025 09:48:15 UTC (expires in 2 months and 28 days)
Key	EC 256 bits
Weak key (Debian)	No
Issuer	E6 AIA: http://e6.i.lencr.org/
Signature algorithm	SHA384withECDSA
Extended Validation	No
Certificate Transparency	Yes (certificate)
OCSP Must Staple	Supported
Revocation information	OCSP OCSP: http://e6.o.lencr.org
Revocation status	Good (not revoked)
DNS CAA	No (more info)
Trusted	Yes Mozilla Apple Android Java Windows

Figure 2: Certificate #1



Additional Certificates (if supplied)

Certificates provided	2 (2065 bytes)
Chain issues	None

#2

Subject	E6 Fingerprint SHA256: 76e9e288aafc0e37f4390cbf946aad997d5c1c901b3ce513d3d8fadbabe2ab85 Pin SHA256: 0Bbh/jEZSKymTy3kTOhsmIHKBB32EDu1KojrP3YfV9c=
Valid until	Fri, 12 Mar 2027 23:59:59 UTC (expires in 2 years and 4 months)
Key	EC 384 bits
Issuer	ISRG Root X1
Signature algorithm	SHA256withRSA

Certification Paths

Click here to expand

Figure 3: Additional Certificates