# HW #7 - Security

**Step One: Creating Root CA**

**Cloning the Repository:**

git clone https://bitbucket.org/stefanholek/pki-example-1

Navigating to the Repository Directory:

cd pki-example-1

**Creating Directories for Root CA:**

mkdir ca\root-ca\private

mkdir ca\root-ca\db

mkdir crl

mkdir certs

**Initializing the Database for Root CA:**

New-Item -Path ca\root-ca\db\root-ca.db -ItemType "file"

New-Item -Path ca\root-ca\db\root-ca.db.attr -ItemType "file"

"01" | Set-Content -Path ca\root-ca\db\root-ca.crt.srl

"01" | Set-Content -Path ca\root-ca\db\root-ca.crl.srl

**Generating Root CA's Private Key and CSR:**

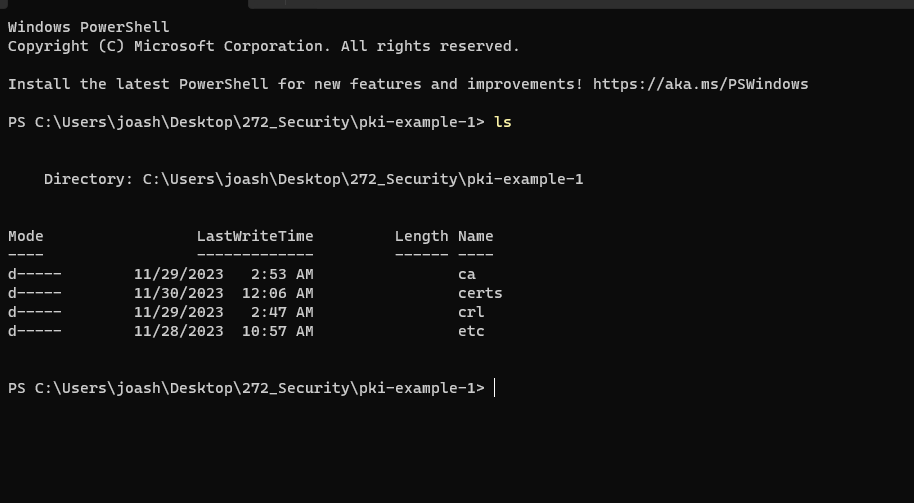
openssl req -new -config etc/root-ca.conf -out ca/root-ca.csr -keyout ca/root-ca/private/root-ca.key

**Self-Signing the Root CA Certificate:**

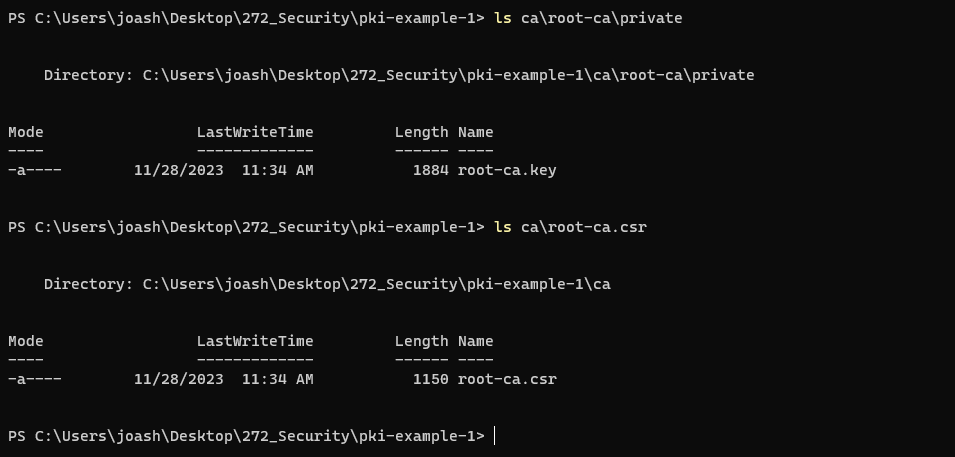
openssl ca -selfsign -config etc/root-ca.conf -in ca/root-ca.csr -out ca/root-ca.crt -extensions root\_ca\_ext

**Confirmation of step 1:**

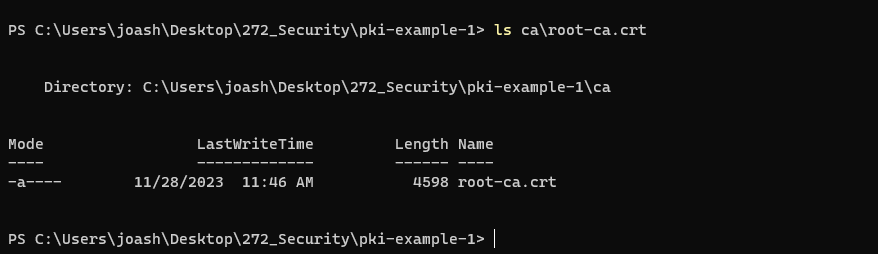
**Check the Created Directories:**



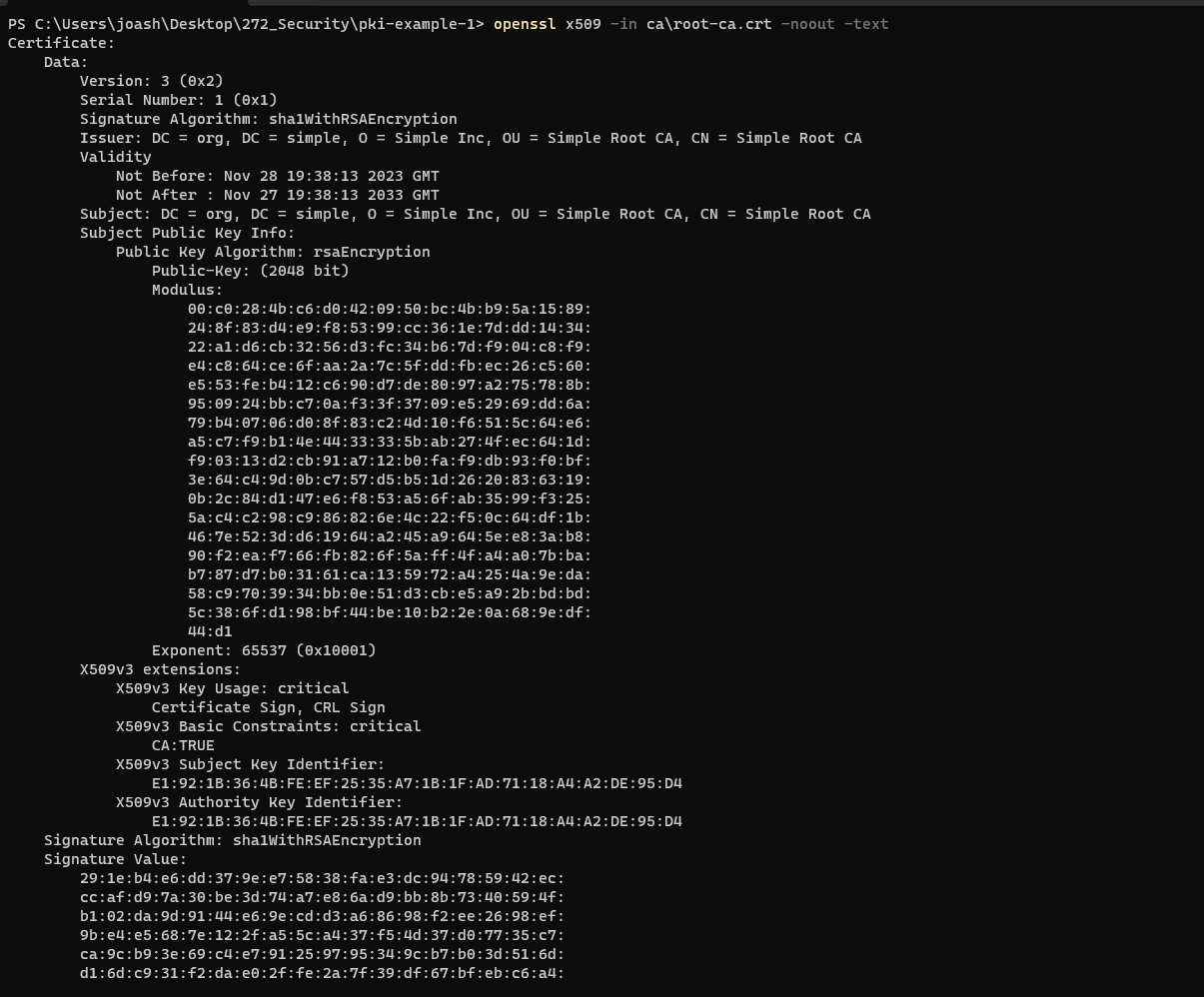
**Check Root CA's Private Key and CSR: Verify the presence of the Root CA's private key and CSR**

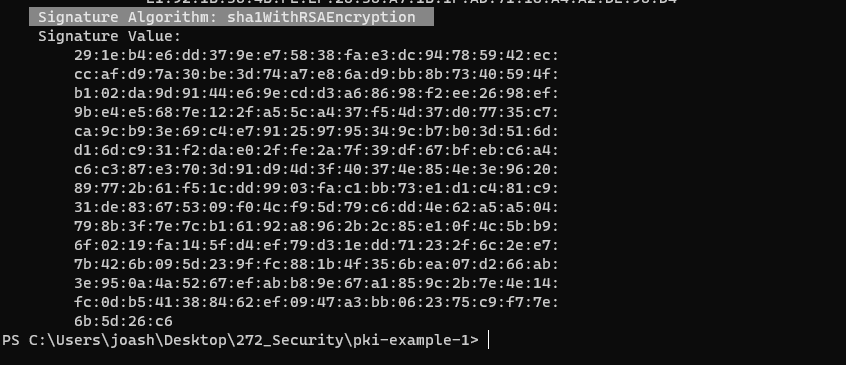


**Verify the Root CA Certificate: Confirm the existence of the self-signed Root CA certificate**.



**Inspect the Root CA Certificate: View the details of the Root CA certificate.**





**Step 2: Creating and setting up the Signing CA**

**Creating Directories for the Signing CA:**

mkdir -p ca\signing-ca\private ca\signing-ca\db crl certs

**Initializing the Database for the Signing CA:**

New-Item -Path ca\signing-ca\db\signing-ca.db -ItemType "file"

New-Item -Path ca\signing-ca\db\signing-ca.db.attr -ItemType "file"

"01" | Set-Content -Path ca\signing-ca\db\signing-ca.crt.srl

"01" | Set-Content -Path ca\signing-ca\db\signing-ca.crl.srl

**Creating the CSR for the Signing CA:**

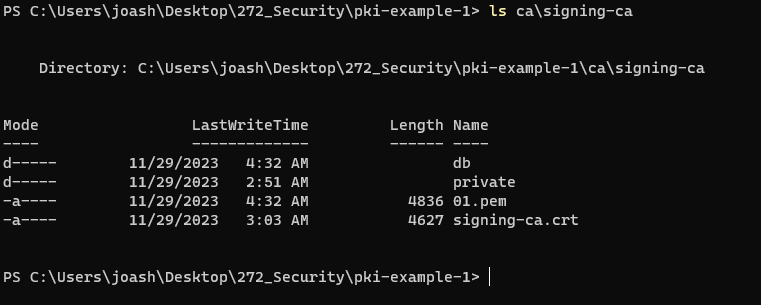
openssl req -new -config etc/signing-ca.conf -out ca/signing-ca.csr -keyout ca/signing-ca/private/signing-ca.key

**Signing the CSR to Create the Signing CA Certificate:**

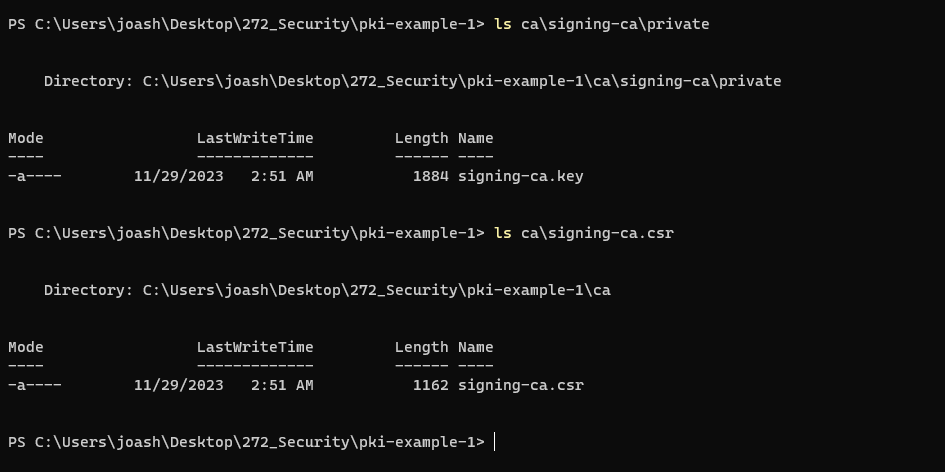
openssl ca -config etc/root-ca.conf -in ca/signing-ca.csr -out ca/signing-ca.crt -extensions signing\_ca\_ext

**Confirming creation and setting up the Signing CA**

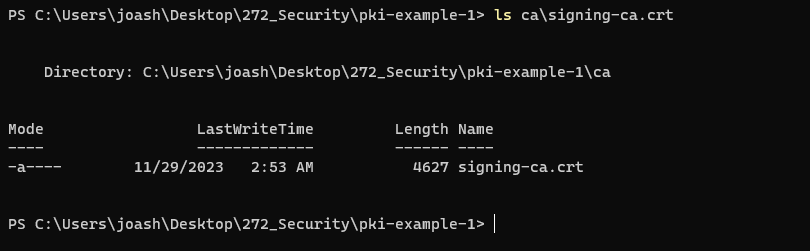
**Check for Signing CA Directories: Verify that the directories for the Signing CA were created.**



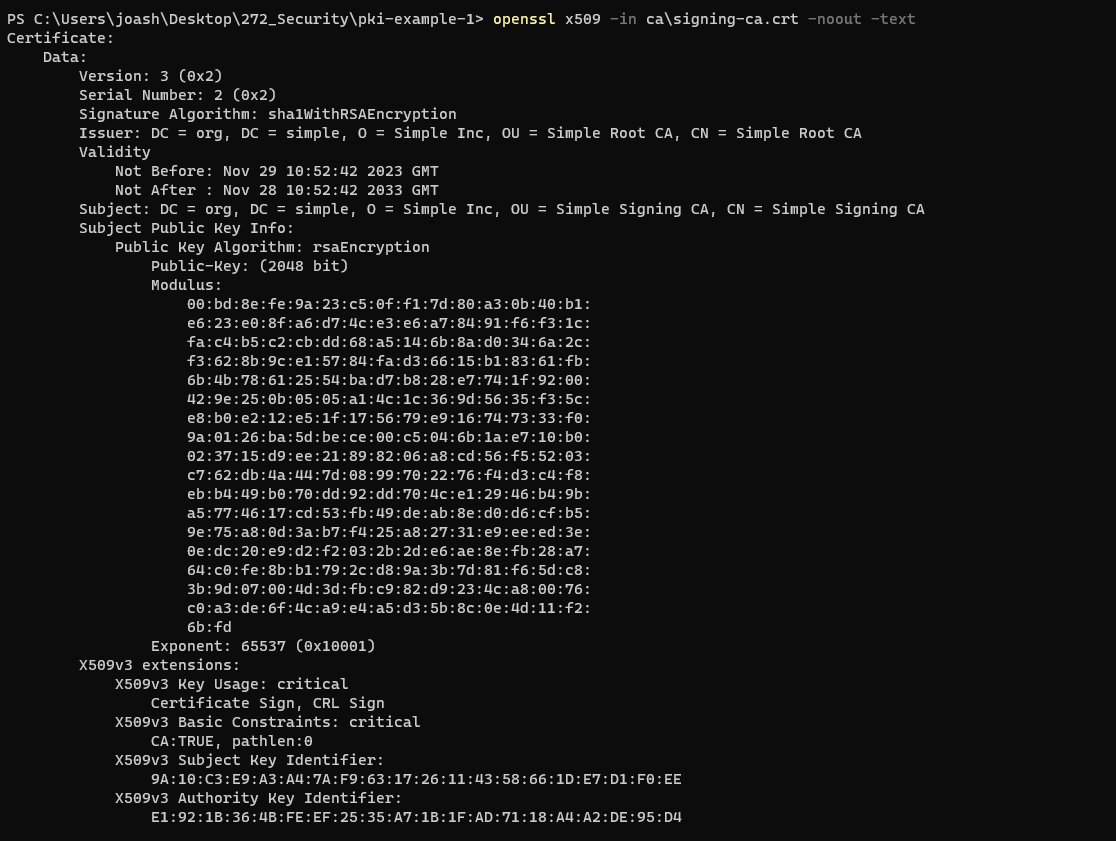
**Verify Signing CA's Private Key and CSR: Confirm the presence of the Signing CA's private key and CSR.**

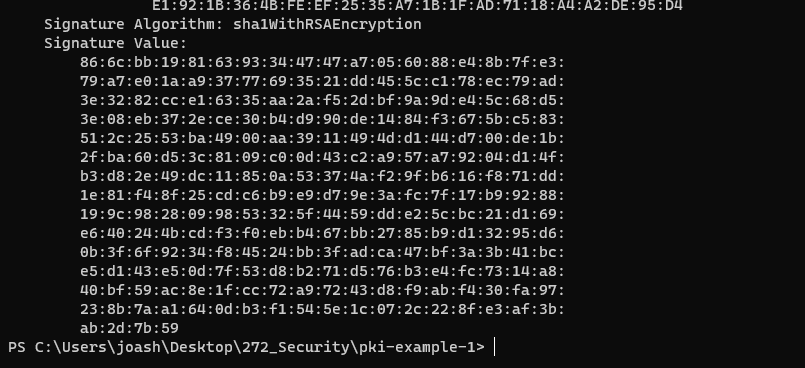
****

**Check the Signing CA Certificate: Confirm the existence of the Signing CA certificate.**



**Inspect the Signing CA Certificate: view the details of the Signing CA certificate.**





**Step 3: Creating the TLS certificate**

**Generating the Web Server's Private Key and CSR:**

openssl req -new -nodes -out certs/webserver.csr -keyout certs/webserver.key -config etc/server.conf

**Signing the Web Server's CSR with the Signing CA:**

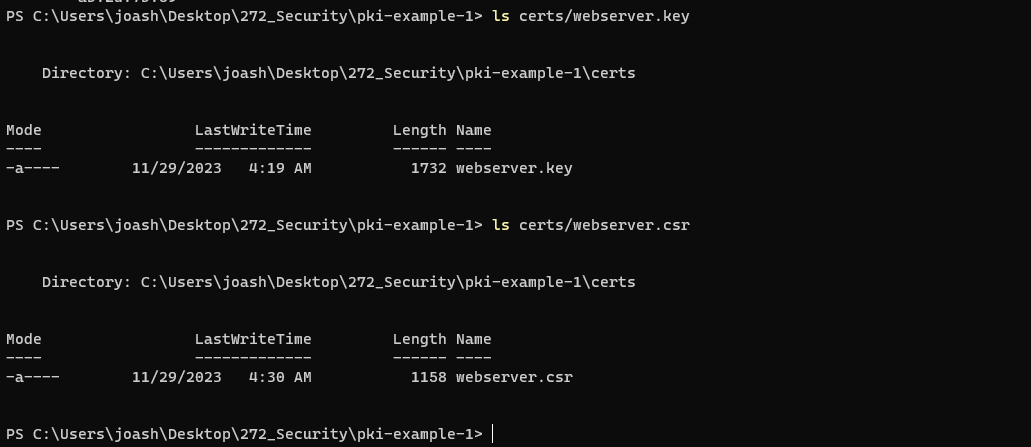
openssl ca -config etc/signing-ca.conf -in certs/webserver.csr -out certs/webserver.crt -extensions server\_ext

**Converting the TLS Certificate to PKCS#12 Format:**

openssl pkcs12 -export -out certs/keystore.p12 -inkey certs/webserver.key -in certs/webserver.crt -certfile ca/signing-ca.crt

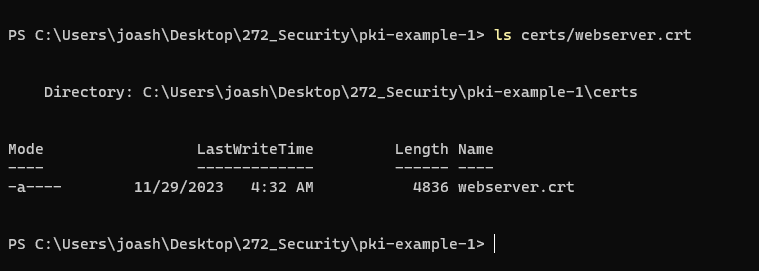
**Confirm that the respective files for each step were successfully created:**

**Confirm Generating the Web Server's Private Key and CSR:**To check if the private key (webserver.key) and CSR (webserver.csr) exist:

****

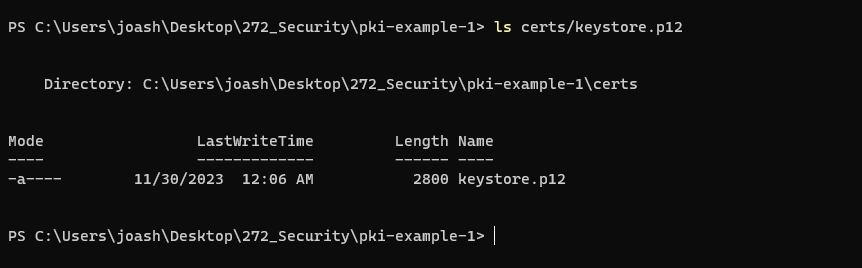
**Confirm Signing the Web Server's CSR with the Signing CA:**

* To check if the signed TLS certificate (webserver.crt) exists:



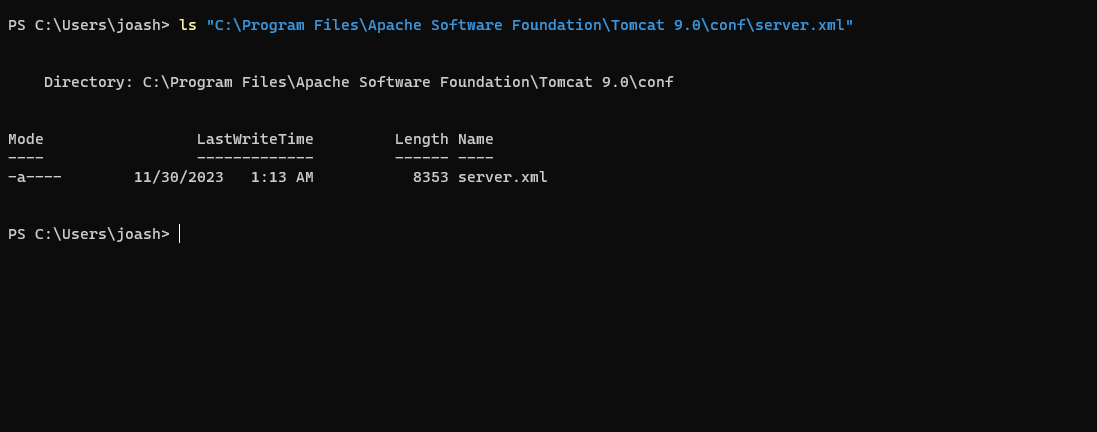
**Confirm Converting the TLS Certificate to PKCS#12 Format:**

* To check if the PKCS#12 keystore file (keystore.p12) exists:

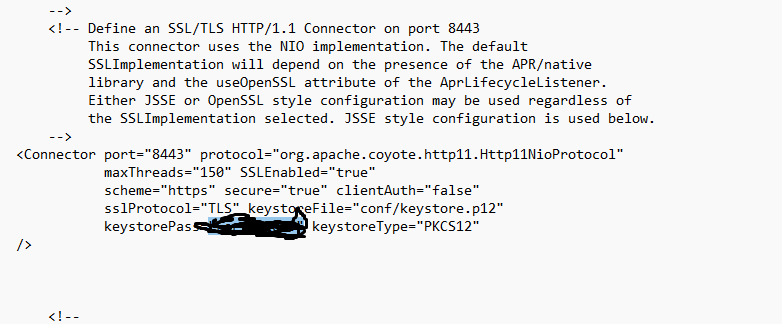


**Step 4: onfiguring Apache Tomcat to use the generated TLS certificate for enabling HTTPS connections**.

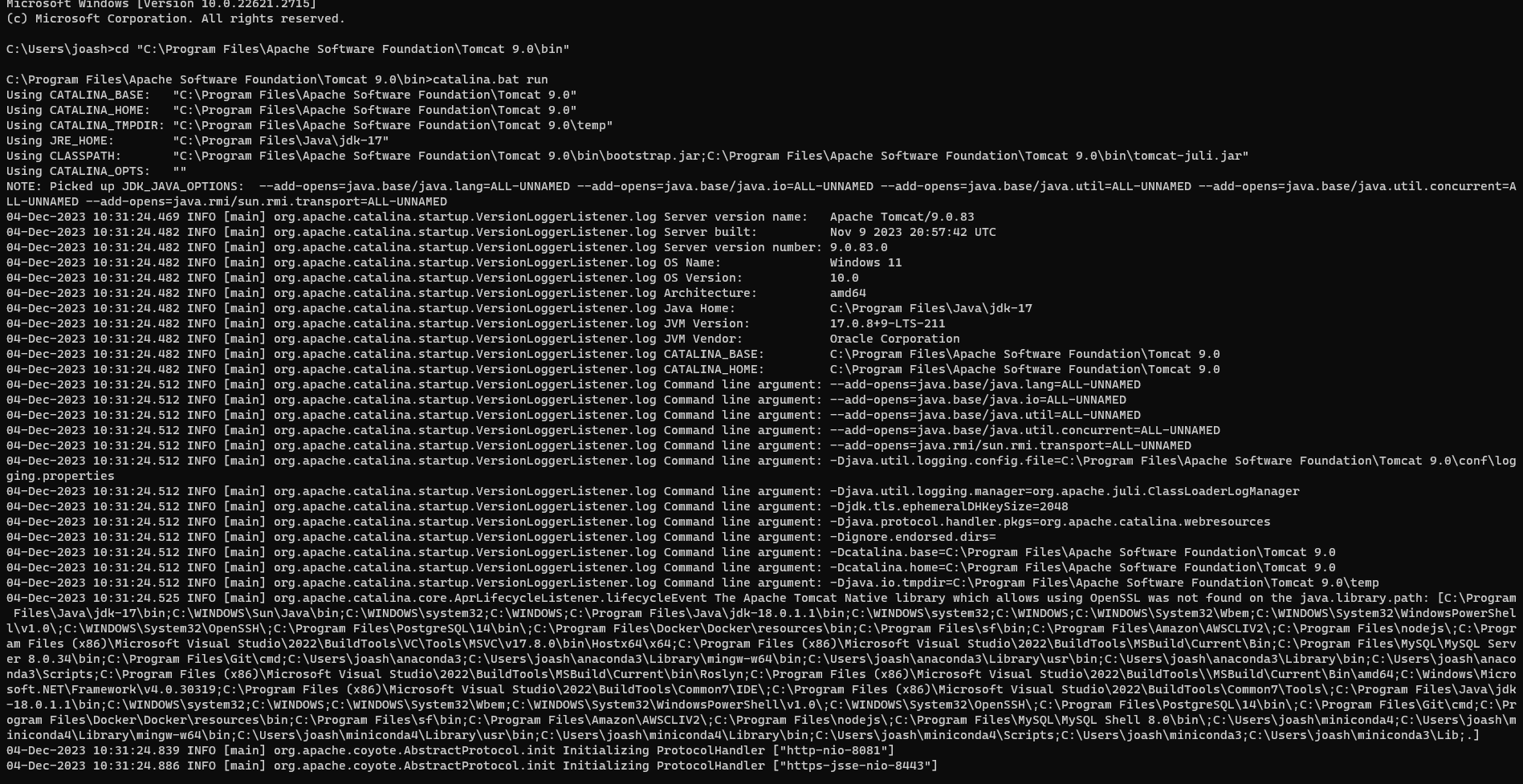
Locate and Open Tomcat's server.xml Configuration File:

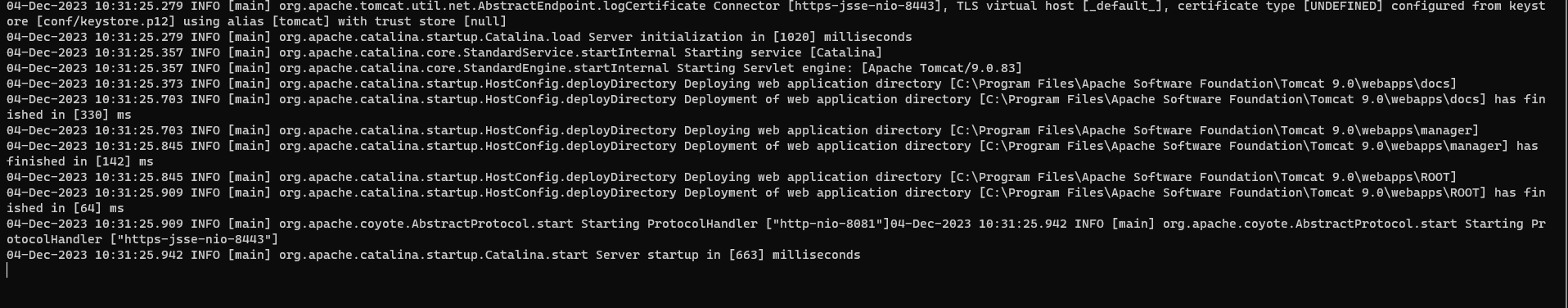


Confirm SSL/TLS Connector Configuration:



**Running Tomcat:**

****

****

**Test HTTPS Connection:**

* To verify that Tomcat is correctly configured with SSL/TLS, open a web browser and navigate to https://localhost:8443.
* If the Tomcat homepage is displayed without any security warnings (or with an expected security warning due to the use of a self-signed certificate), it confirms that SSL/TLS has been successfully configured.

