



This PDF is generated from authoritative online content, and is provided for convenience only. This PDF cannot be used for legal purposes. For authoritative understanding of what is and is not supported, always use the online content. To copy code samples, always use the online content.

# Web Services and Applications Deployment Guide

Appendix: How-to Create SSL Certificate

---

## Contents

- 1 Appendix: How-to Create SSL Certificate
  - 1.1 Prerequisites
  - 1.2 Create Root Certificate
  - 1.3 Verify Root Certificate
  - 1.4 GWS Key and Certificate Generation
  - 1.5 Converting Procedures
  - 1.6 Import Missing Certs and Create Truststore
  - 1.7 GWS Configuration (application.yaml)
  - 1.8 On Client Desktop

# Appendix: How-to Create SSL Certificate

## Prerequisites

- Create the root pair (rootCA key & rootCA cert).
- Prepare the `mkdir /root/ca` directory.
- Create the directory structure:

```
# cd /root/ca
# mkdir certs crt newcerts private
# chmod 700 private
# touch index.txt
# echo 1000 > serial
```

- Copy the root CA configuration (openssl.cnf) to `/root/ca/openssl.cnf`
- Create the root key:

```
# cd /root/ca
# openssl genrsa -aes256 -out private/<rootCA>.key.pem 4096
```

- Enter pass phrase for `<rootCA>.key.pem`: <Enter password>
- Verifying - Enter pass phrase for `<rootCA>.key.pem`: <Enter password>

```
# chmod 400 private/<rootCA>.key.pem
```

## Create Root Certificate

- Use the `<rootCA>.key.pem` root key to create the `<rootCA>.cert.pem` root certificate.

```
# cd /root/ca
# openssl req -config openssl.cnf -key private/<rootCA>.key.pem -new -x509 -days 7300
-sha256 -extensions v3_ca -out certs/<rootCA>.cert.pem
```

- Enter the pass phrase for `<rootCA>.key.pem`: <password for "rootCA.key.pem">

You are about to be asked to enter information that will be incorporated into your certificate request.

```
-----
Country Name (2 letter code) [XX]: <Enter country code>
State or Province Name []: <Enter state or province>
Locality Name []: <Enter city>
Organization Name []: <Enter company name>
Organizational Unit Name []: <Enter company OU>
Common Name []: <Enter some value>
Email Address []: <Enter admin mail account>
```

```
# chmod 444 certs/<rootCA>.cert.pem
```

## Verify Root Certificate

```
# cd /<rootCA>.cert.pem
```

The output shows:

- The Signature Algorithm used
- The dates of certificate Validity
- The Public-Key bit length
- The Issuer, which is the entity that signed the certificate
- The Subject, which refers to the certificate itself

The Issuer and Subject are identical as the certificate is self-signed. Note that all root certificates are self-signed.

```
Signature Algorithm: sha256WithRSAEncryption
Issuer: C=GB, ST=England,
       O=Alice Ltd, OU=Alice Ltd Certificate Authority,
       CN=Alice Ltd Root CA
Validity
  Not Before: Apr 11 12:22:58 2015 GMT
  Not After : Apr  6 12:22:58 2035 GMT
Subject: C=GB, ST=England,
       O=Alice Ltd, OU=Alice Ltd Certificate Authority,
       CN=Alice Ltd Root CA
Subject Public Key Info:
  Public Key Algorithm: rsaEncryption
  Public-Key: (4096 bit)
```

The output also shows the X509v3 extensions. We applied the v3\_ca extension, so the options from [ v3\_ca ] should be reflected in the output.

```
X509v3 extensions:
X509v3 Subject Key Identifier:
  38:58:29:2F:6B:57:79:4F:39:FD:32:35:60:74:92:60:6E:E8:2A:31
X509v3 Authority Key Identifier:
  keyid:38:58:29:2F:6B:57:79:4F:39:FD:32:35:60:74:92:60:6E:E8:2A:31

X509v3 Basic Constraints: critical
  CA:TRUE
X509v3 Key Usage: critical
  Digital Signature, Certificate Sign, CRL Sign
```

## GWS Key and Certificate Generation

- Make a directory for GWS files:  

```
# cd /root/ca # mkdir gwsCerts
```
- Create a key:  

```
# cd /root/ca
```

```
# openssl genrsa -aes256 -out gwsCerts/<gwsKey>.key.pem 2048
# chmod 400 gwsCerts/<gwsKey>.key.pem
```

- Create a certificate (CSR):

**Requirement**

the Common Name must be a fully qualified domain name.

**Copy san.cnf and v3.ext to /root/ca and modify the following parameters in these files**

```
commonName = <Enter FQDN of your GWS host>
DNS.1      = commonName
DNS.2      = *.<part of FQDN>
```

```
# cd /root/ca
# openssl req -out gwsCerts/<gwsCSR>.csr -newkey rsa:2048 -nodes -keyout
gwsCerts/<gwsKey>.key.pem -config san.cnf
```

- **Enter pass phrase for <gwsKey>.key.pem**  
    <password for gws Key>

**You are about to be asked to enter information that will be incorporated into your certificate request.**

```
Country Name (2 letter code) [XX]: <Enter country code>
State or Province Name []: <Enter state>
Locality Name []: <Enter city>
Organization Name []: <Enter company>
Organizational Unit Name []: <Enter company OU>
Common Name []: <Enter FQDN of your GWS host>
Email Address []: <Enter email address>
```

- Sign the GWS CSR file:  
**Use rootCA authority to sign up GWS csr file.**

```
# cd /root/ca
# openssl x509 -req -sha256 -days 367 -in gwsCerts/<gwsCSR>.csr -CA <full path to
'rootCA.cert.pem'> -CAkey <full path to 'rootCA.key.pem'> -CAcreateserial -out
gwsCerts/<gwsSignedCert>.pem -extfile v3.ext -extensions v3_req
# chmod 444 gwsCerts/<gwsSignedCert>.cert.pem
```

**Example:**

```
openssl x509 -req -sha256 -days 367 -in gwsCerts/gwsCSR.csr -CA /root/ca/certs/
rootCA.cert.pem -CAkey /root/ca/private/rootCA.key.pem -CAcreateserial -out gwsCerts/
gwsSignedCert.pem -extfile v3.ext -extensions v3_req
# chmod 444 gwsCerts/gwsSignedCert.pem
```

- Verify the certificate:

```
# openssl x509 -noout -text -in gwsCerts/<gwsSignedCert>.pem
Check for x509v3 extensions (SAN & v3 extensions).
```

## Converting Procedures

- Convert the existing cert to a PKCS12 using OpenSSL.

### Important

A password is required.

```
# cd /root/ca
# openssl pkcs12 -export -in <gwsSignedCert>.pem -inkey <gwsKey>.key.pem -out
<keystore.p12> -name <certAlias> -CAfile <full path to 'rootCA.cert.pem'> -caname rootCA
```

#### Example:

```
# openssl pkcs12 -export -in /root/ca/gwsCerts/gwsSignedCert.pem -inkey /root/ca/gwsCerts/
gwsKey.key.pem -out keystore.p12 -name firstcert -CAfile /root/ca/certs/rootCA.cert.pem
-caname rootCA
```

- Convert the PKCS12 to a Java Keystore File.

```
# cd /root/ca
# keytool -importkeystore -deststorepass <new_keystore_pass> -destkeypass <new_key_pass>
-destkeystore <gwsKeystore.jks> -srckeystore <keystore.p12> -srcstoretype PKCS12
-srcstorepass <pass_used_in_p12_keystore> -alias <alias_used_in_p12_keystore>
```

#### Example:

```
keytool -importkeystore -deststorepass password -destkeypass password -destkeystore
gwsKeystore.jks -srckeystore keystore.p12 -srcstoretype PKCS12 -srcstorepass password
-alias firstcert
* System will automatically tell to change the format:<source lang="text">
keytool -importkeystore -srckeystore gwsKeystore.jks -destkeystore gwsKeystore.jks
-deststoretype pkcs12
```

## Import Missing Certs and Create Truststore

- Import rootCa certificate to gwsKeystore.jks:
- Use keytool -importcert to import the rootCa certificate into each node keystore:

```
# cd /root/ca
# keytool -importcert -keystore <gwsKeystore>.jks -alias rootCA -file <path to
'rootCA.cert.pem'> -noprompt -keypass <keystore password> -storepass <password>
```

#### Example:

```
# keytool -importcert -keystore gwsKeystore.jks -alias rootCA -file /root/ca/certs/
rootCA.cert.pem -noprompt -keypass password -storepass password
```

- Create a server truststore:

```
# cd /root/ca
# keytool -importcert -keystore <gwsTruststore>.jks -alias rootCA -file
<rootCA>.cert.pem -noprompt -keypass <key password> -storepass <password>
```

#### Example:

```
# keytool -importcert -keystore gwsTruststore.jks -alias rootCA -file /root/ca/certs/
rootCA.cert.pem -noprompt -keypass password -storepass password
```

## GWS Configuration (application.yaml)

- Configuration example (**jetty** section):

```
enableSsl: true
ssl:
  port: 8443
  securePort: 443
  idleTimeout: 30000
  soLingerTime: -1
  trustAll: true
  keyStorePath: /root/ca/<gwsKeystore>.jks
  keyStorePassword: <keystore password>
  keyStoreType: JKS
  trustStorePath: /root/ca/<gwsTruststore>.jks
  trustStorePassword: <truststore password>
```

### Example:

```
# caCertificate: /root/ca/myKeystore.jks
# jksPassword: Manila@1234
port: 8443
securePort: 443
idleTimeout: 30000
soLingerTime: -1
trustAll: true
keyStorePath: /root/ca/gwsKeystore.jks
keyStorePassword: password
keyStoreType: JKS
trustStorePath: /root/ca/gwsTruststore.jks
trustStorePassword: password
```

## On Client Desktop

- Add your host (hostname should be specified as FQDN) in <%system\_drive%>\Windows\System32\drivers\etc\hosts.

### Example in the file

```
192.168.100.26      gws-centos7.genesys.com
```

- Convert <rootCA>.cert.pem to PFX format:

```
# cd /root/ca
# openssl pkcs12 -inkey <rootCA>.key.pem -in <rootCA>.cert.pem -export -out <rootCA>.pfx
```

### Example:

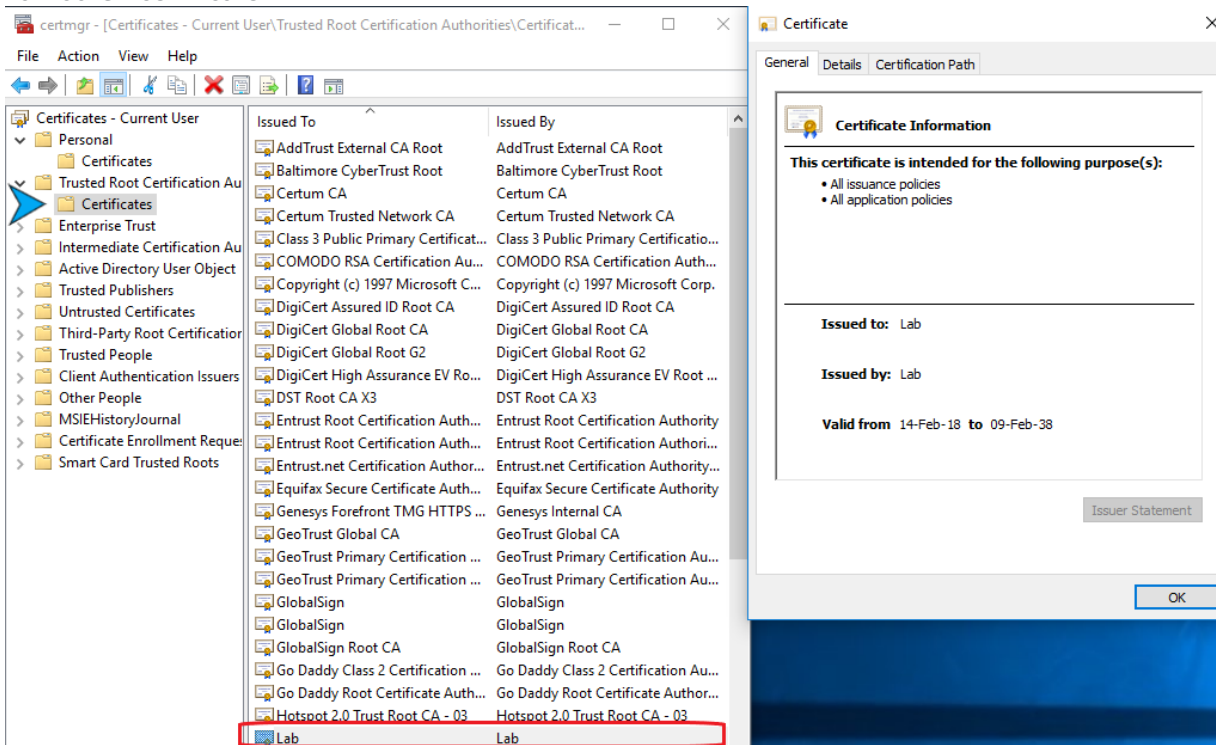
```
openssl pkcs12 -inkey /root/ca/private/rootCA.key.pem -in /root/ca/certs/rootCA.cert.pem
-export -out rootCA.pfx
```

- Copy <rootCA>.pfx and <keystore>.p12 to Windows host.
- Import the <keystore>.p12 file by double-clicking on it. Use default configuration and specify the password.
- Import the <rootCA>.pfx file, make sure to select “Place all certificates in the following store”. Browse “Trusted Root Certification Authorities”

- Verify that certificates are present using `certmgr.msc`.

### Example:

- For rootCA certificate:



- For GWS certificate:

