Version: July 2026

# LSEG Workspace

## WebSocket Secure Configuration Guide

LSEG DATA & ANALYTICS

# Contents

# About this document

This document outlines the necessary steps for generating SSL Certificates required to enable WebSocket Secure port on ADS/RTC to accept mounts from Real-Time data consuming applications like LSEG Workspace. For creating self-signed certificate procedure, this guide could be complemented with WSS enablement script which is distributed separately.

**In this document, it is assumed that all operations on the Linux machines running ADS and RTC processes[1] are performed under the radmin user account.**

## Intended readership

This reference guide is intended for LSEG/Refinitiv Customers who would like to enable encryption for Real-Time traffic from ADS/RTC over WebSocket protocol.

## Further information

To:

- Request product assistance, contact Support.
- Access other LSEG Workspace technical content, see the Workspace technical documentation site.
- Provide feedback on Workspace technical content, contact DocFeedback@lseg.com.

[1] ADS and RTC must be on version 3.7.2 or above.

# Generating certificates

Real-Time Advanced Distribution System (ADS) and the Real-Time Connector (RTC) can listen and accept mounts from Real-Time data-consuming applications – such as LSEG Workspace – on both WebSocket (WS) and WSS ports simultaneously. However, to enable WebSocket Secure (WSS) connectivity in ADS and RTC, certificates must be generated.

There are three common approaches for generating certificates:

- Through a Public Root Certificate Authority
- Through your Enterprise Root Certificate Authority
- As a self-signed certificate

The certificate creation procedures require the OpenSSL opensource package to be installed to run the commands shown in the sections below. It can be installed and used for these purposes either on the Linux machine running the ADS and RTC process or on any workstation running Windows or Linux.

# Generating through a Public Root Certificate Authority

To generate a certificate through a Public Root Certificate Authority (or *Root CA*), follow the steps below:

1. **Create a Certificate Signing Request (CSR) configuration file**

   Use a text editor to create a file named `<ADS_hostname>.cnf`. For example:

```
[req]
prompt = no
default_bits = 2048
distinguished_name = req_distinguished_name
req_extensions = req_ext      # Required to add SAN to CSR

[req_distinguished_name]
C = PL                        # Country (2-letter code)
ST = Pomerania# State/Province
L = Gdynia                    # City
O = LSEG                      # Organization
OU = TIS                      # Organizational Unit
CN = lseg.com                 # Primary domain (still required but not used for validation)

[req_ext]
keyUsage = digitalSignature, keyEncipherment
extendedKeyUsage = serverAuth
subjectAltName = @alt_names   # Reference SANs below

[alt_names]
DNS.1 = lseg.com              # Primary domain
DNS.2 = adshost1.lseg.com     # Subdomain 1
DNS.3 = adshost2.lseg.com     # Subdomain 2
IP.1 = 192.168.1.101          # Optional: Add IP addresses
IP.2 = 192.168.1.102
```

   ⚠ To prevent rejections by strict validation servers, defining Common Name (CN) as a first Subject Alternative Name (SAN) is a requirement. SANs must match the ADS hostnames that are configured either in your shared configuration JSON file or defined under the Configuration Manager in LSEG Workspace.

2. **Generate Private Key[2] and CSR**

   The Certificate Signing Request (CSR) file contains information that identifies your organization and domain, including your public key, which is signed by your private key.

```
openssl req -new -newkey rsa:2048 -nodes  -keyout <ADS_hostname>.key  -out
<ADS_hostname>.csr -config <ADS_hostname>.cnf
```

[2] The private key is not protected with a passphrase.

📌 Depending on policies in your organisation, the RSA-2048 asymmetric encryption algorithm that used in the example statement above, could be changed. For example, you can amend the `-newkey` parameter to specify a key pair encryption system, such as `rsa:3072` or `rsa:4096`.

3. **Examine CSR file created**

```
openssl req -noout -text -in <ADS_hostname>.csr
```

4. **Submit CSR to your preferred Public Root Certificate Authority (CA)**

5. **Download and store certificate files**

   It is recommended that certificates and private key files are stored under `/opt/refinitiv/SOFTWARE/globalconfig`, in a separate sub-folder, for example, `certificates`. This subfolder can also be hidden.

6. **Concatenate primary and intermediate files**

   To build a complete trust chain, concatenate the files by using the following statement:

```
cat <ADS_hostname>.crt CA-bundle.crt > <ADS_hostname>_bundle.crt
```

📌 To enable WebSocket Secure protocol in ADS and RTC, refer to Enable WebSocket Secure listening port in ADS/RTC.

# Generating through an Enterprise Root Certificate Authority

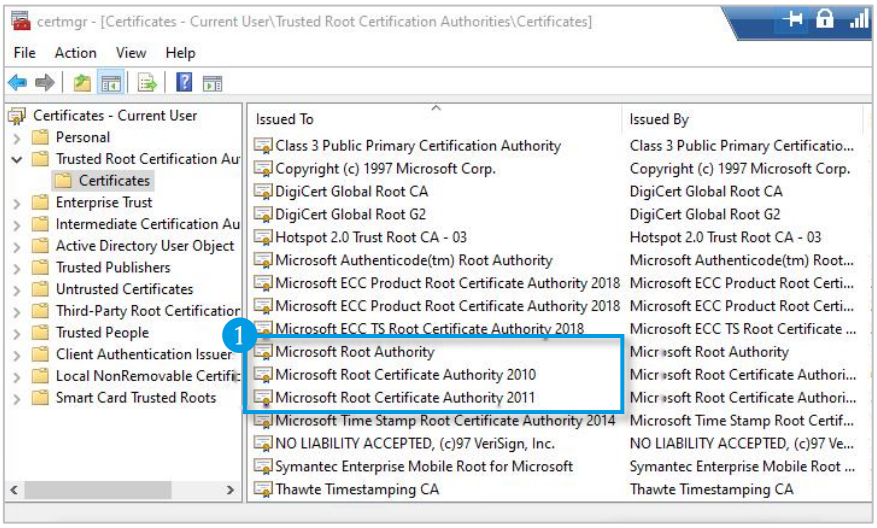
If your organisation has its own Enterprise Root Certificate Authority, certificates for ADS and RTC can be generated using that service. For further information about Active Directory Certificate Services (AD CS), including installation, configuration, and use of group policy certificate distribution, refer to the Microsoft AD CS documentation site.

All examples in this section are for informational purposes only, as the actual procedures can vary by customers. Please consult your local AD CS Administrator for detailed instructions or assistance.

📌 You can use steps from Create Certificate from Public Root Authority to create CSR file to share with your internal Security Team responsible for Enterprise Certificate management. Or refer to the steps described below.

1. **Establish a trust relationship between the PC running LSEG Workspace and the Enterprise Root CA**

   To establish a trust relationship, the Enterprise CA Root Certificate must be installed on each PC that runs LSEG Workspace. This is done through the Trusted Root Certificate Authority store in the Certificate Manager tool, as highlighted (1) in the example, opposite:




2. **Create a Private Key with a custom Request in AD CS**

   Do the following:

   a) Run (Windows key + R) **mmc.exe**.

   b) In the MMC Console, select **File** > **Add/Remove Snap-in…**

   c) In the Available snap-ins section of the Add or Remove Snap-ins dialog, select **Certificates** and click **Add >**.

   d) In the Certificates snap-in panel, select **Computer Account** and click **Next**.

   e) In the Select Computer panel, select **Local Computer** and click **Finish**.

   f) Return to the MMC Console, expand Certificates (Local Computer) Snap-in, to display the logical store names, and right-click **Personal**.

   g) Select **All Tasks** > **Advanced Operations** > **Create Custom Request**.

   h) In the Select Certificate Enrollment Policy panel, select **Proceed without enrolment policy** and click **Next**.

i) In the Custom request panel, for Request format, select **PKCS #10** and click **Next**.

j) In the Certificate Information panel, under the **Details** drop-down section, click **Properties**.

k) Under the **Subject** tab, set the **Subject Name** (CN=**<ADS_hostname>**, O=Org, L=City, S=State, C=Country) and add the necessary SANs (DNS names).

You are advised not to rely solely on the CN definition, as it is often considered obsolete for hostname checking. Use a SAN instead, with the CN explicitly as the first entry. this approach also enables a single signed certificate to be created for multiple ADS and RTC machines.

l) Under the **Private Key** tab, set the **Key size** to 2048 (RSA, if applicable) and select the **Make private key exportable** checkbox.

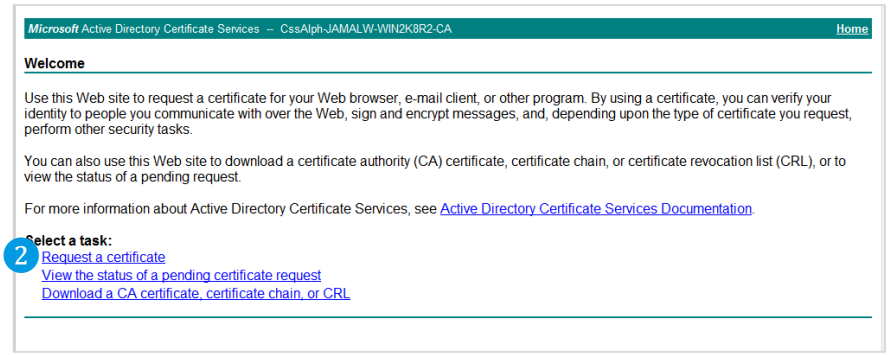
m) Click OK to save your settings and return to the Certificate Enrollment wizard.

n) In the Where do you want to save the offline request? panel, specify the file name (for example, `<ADS_HOSTNAME>.csr`) and location, and click **Finish**.

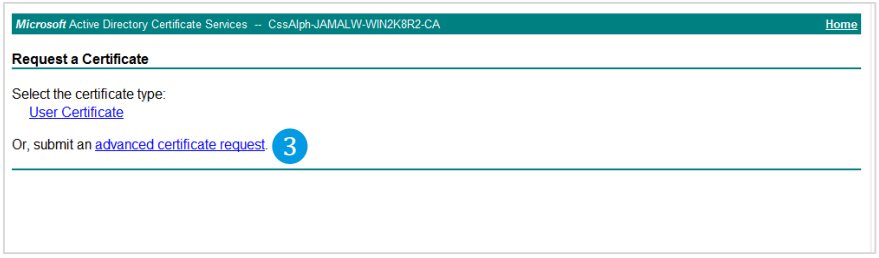
**3. Submit the CSR to AD CS**

a) In a web browser, navigate to `http://<CA-Server>/certsrv`.

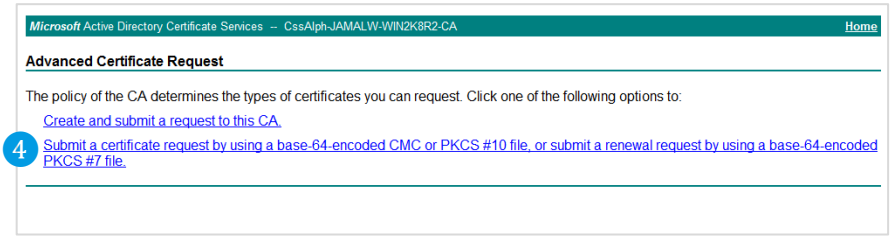
b) Under the Welcome page, select **Request a certificate** 2.




c) Under the Request a Certificate page, select **advanced certificate request** 3.



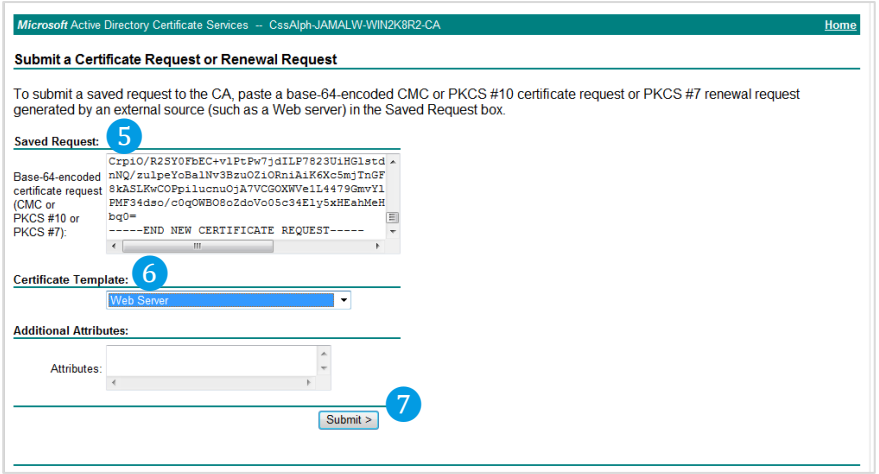

d) Under the Advanced Certificate Request page, select **Submit a certificate request...** 4.




e) Open the `<ADS_HOSTNAME>.csr` file in Notepad, then copy the entire content (including `-----BEGIN`..., and so on), and paste it into the **Saved Request** box 5.

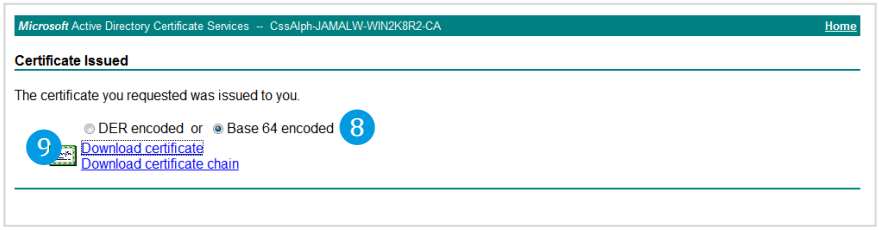
f) Under **Certificate Template** 6, select the appropriate computer.

g) Click **Submit** 7.



h) Under the Certificate Issued page, select **Base 64 encoded** 8 and click **Download certificate** 9.

i) Export the Private Key and copy it together with the file downloaded in previous step to the Linux machine running ADS and RTC processes.




4. **Upload prepared certificate files to ADS machines**

a) [Recommended] Store certificates and private key files under `/opt/refinitiv/SOFTWARE/globalconfig`, in a separate subfolder, `certificates`, for example[3].

b) Copy the certificate and private key files to the sub-folder[4].

📌 To enable WebSocket Secure protocol in ADS and RTC, refer to Enable WebSocket Secure listening port in ADS/RTC.

# Generating a self-signed certificate

This can be done in the following ways:

- Creating a separate certificate for each ADS host
- Creating a single certificate for multiple ADS machines
- Uploading the <ADS_hostname>_CA.crt to user PCs

# Creating a separate certificate for each ADS host

To do this, follow the steps below:

1. **Create a private key for the root CA of your server**

a) Use an openssl statement, similar to the example below, to create the private key. This key is used to sign the root certificate:

```
openssl genrsa -des3 -out <ADS_hostname>_CA.key 2048
```

b) Enter a passphrase and store it in a safe location.

📌 Do not share your Private Keys with anyone outside your organisation.

2. **Create a public certificate for the CA of your server**

The public certificate is used to sign other certificates for ADS and RTC services. It can also be distributed to either:

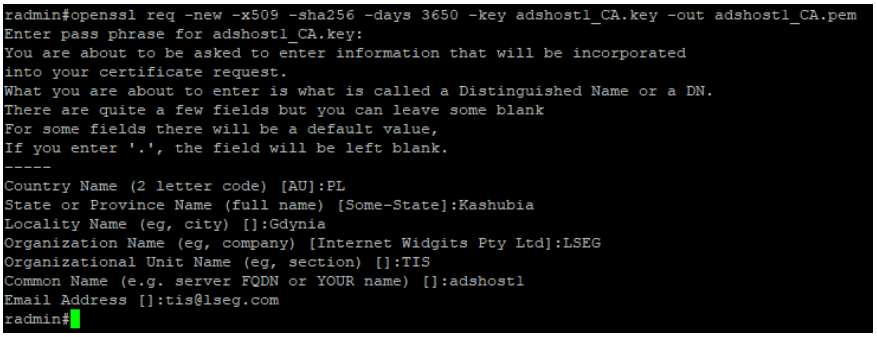
- the computer of every user running LSEG Workspace, or
- Your Enterprise Certificate Authority Trust storage.

To create the public certificate, use an openssl statement, similar to the example below:

```
openssl req -new -x509 -sha256 -days 3650 -key <ADS_hostname>_CA.key -out
<ADS_hostname>_CA.pem
```

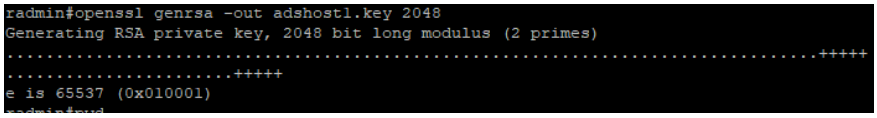
[3] This subfolder can also be hidden.
[4] Each signed certificate and the private key it is based on must be copied to every machine running ADS.

You are prompted to enter the details and email address of your organisation. If you intend to create separate certificate for every ADS, then use the output of `hostname` command of the Linux machine hosting ADS as the Common Name (CN).




3. **Examine the CSR file created**

   To verify the decrypted content of the CSR file, use the following statement




```
openssl req -noout -text -in <ADS_hostname>.csr
```

4. **Create Certificate Signing Request (CSR) for ADS/RTC service:**

```
openssl req -new -key <ADS_hostname>.key -out <ADS_hostname>.csr -sha256
```

You are prompted to enter a Challenge Password. This should be blank.

5. **Manually create an .ext file**

   Use a text editor to manually create an .ext file (`<ADS_hostname>.ext` in the example, below) with the FQDNs of the server(s) running ADS and RTC services:

```
keyUsage = critical, digitalSignature, keyEncipherment
extendedKeyUsage = serverAuth
subjectAltName = @alt_names
[alt_names]
DNS.1 = <ADS_hostname1>
DNS.2 = <ADS_hostname2>
.
.
DNS.#N = <ADS_hostname#N>
```

   If you intend to use ADS and RTC IP addresses to set up streaming using Configuration Manager in LSEG Workspace, add the following parameters to the file:

   - IP.1 = 192.168.1.11
   - IP.2 = 192.168.1.12

6. **Sign the CSR with a CA certificate**

   Use an openssl statement, similar to the example below, to generate self-signed certificate

```
openssl x509 -req -sha256 -days 3650 -in <ADS_hostname>.csr -CA <ADS_hostname>_CA.pem -
CAkey <ADS_hostname>_CA.key -CAcreateserial -out <ADS_hostname>.crt -extfile
<ADS_hostname>.ext
```

You are prompted to enter the passphrase for your private CA key.

7. **Convert CA certificate from PEM to CRT format**

   Windows accepts CA certificates in CRT format only. So, to convert your file, use an openssl state (see the example below):

```
openssl x509 -outform der -in <ADS_hostname>_CA.pem -out <ADS_hostname>_CA.crt
```

# Creating a single certificate for multiple ADS machines

To do this, follow the steps below:

1. **Create a private key for the root CA of your server**

   a) to create the private key, use an openssl statement (see the example below). This key is used to sign the root certificate:

```
openssl genrsa -des3 -out <DOMAIN_NAME>_CA.key 2048
```

b) Enter a passphrase and store it in a safe location.

📌 Do not share your Private Keys with anyone outside your organisation.

**2. Create a public certificate for the CA of your server**

The public certificate is used to sign other certificates for ADS and RTC services. It can also be distributed to either:

- the computer of every user running LSEG Workspace, or
- Your Enterprise Certificate Authority Trust storage.

To create the public certificate, use an openssl statement (see the example below):

```
openssl req -new -x509 -sha256 -days 3650 -key <DOMAIN_NAME>_CA.key -out
<DOMAIN_NAME>_CA.pem
```

📌 You are prompted to enter the details and email address of your organisation. You can use wildcards for the CN, if a single certificate will be used on multiple ADS and RTC servers. Refer to the `DNS.1` entry in step 3, below, for a wildcard example.

**3. Create the CSR configuration file**

Use a text editor to create a CSR configuration file `<DOMAIN_NAME>.cnf`, similar to the example, below:

```
[req]
prompt = no
default_bits = 2048
distinguished_name = req_distinguished_name
req_extensions = req_ext      # Required to add SAN to CSR

[req_distinguished_name]
C = PL                        # Country (2-letter code)
ST = Pomerania# State/Province
L = Gdynia                    # City
O = LSEG                      # Organization
OU = TIS                      # Organizational Unit
CN = lseg.com                 # Primary domain (still required but not used for validation)

[req_ext]
basicConstraints = CA:FALSE
keyUsage = digitalSignature, keyEncipherment
extendedKeyUsage = serverAuth
subjectAltName = @alt_names   # Reference SANs below

[alt_names]
DNS.1 = *.lseg.com            # Primary domain
DNS.2 = adshost1.lseg.com     # Subdomain 1
DNS.3 = adshost2.lseg.com     # Subdomain 2
IP.1 = 192.168.1.101          # Optional: Add IP addresses
IP.2 = 192.168.1.102
```

📌 To prevent rejections by strict validation servers, the Common Name (CN) must be defined as the first SAN. SANs must match ADS hostnames that are configured either in your shared configuration JSON file or under the Configuration Manager option in LSEG Workspace.

**4. Generate the private key and CSR files for your ADS service**

Use an OpenSSL command, similar to the example below, to generate the files:

```
openssl req -new -newkey rsa:2048 -nodes -keyout <DOMAIN_NAME>.key -out <DOMAIN_NAME>.csr -
config <DOMAIN_NAME>.cnf
```

📌 Depending on policies in your organisation, the RSA-2048 asymmetric encryption algorithm that used in the example statement above, could be changed. For example, you can amend the `-newkey` parameter to specify a key pair encryption system, such as `rsa:3072` or `rsa:4096`.

**5. Sign CSR with CA certificate**

```
openssl x509 -req -sha256 -days 3650 -in < DOMAIN_NAME>.csr -CA <DOMAIN_NAME>_CA.pem -CAkey
<DOMAIN_NAME>_CA.key -CAcreateserial -out <DOMAIN_NAME>.crt -extfile <DOMAIN_NAME>.ext
```

You are prompted to enter the passphrase for your private CA key.

6. **Convert CA certificate from PEM to CRT format**

   Windows accepts CA certificates in CRT format only. So, use an openssl state, similar to the example below, to convert your file:

```
openssl x509 -outform der -in <DOMAIN_NAME>_CA.pem -out <DOMAIN_NAME>_CA.crt
```

# Uploading the <ADS_hostname>_CA.crt to user PCs

You can upload an issued `<ADS_hostname>_CA.crt` or `<DOMAIN_NAME>_CA.pem` to every computer running LSEG Workspace and import it into Trusted Root Certificate Authorities.

1. **Import the certificate**

   You can import the certificate .crt file either by using the Certificate Import Wizard, or by running a PowerShell cmdlet.

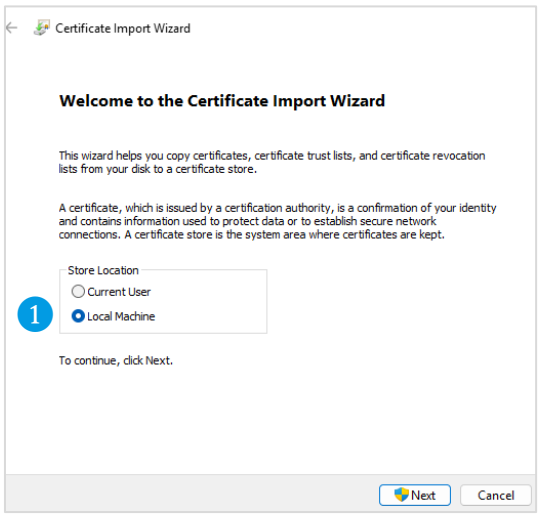
   **Using the Certificate Import Wizard**

   a) To run the Certificate Import Wizard, in Windows File Explorer, right-click the `<ADS_hostname>_CA.crt` file and select **Install Certificate**.

      The Welcome to the Certificate Import Wizard dialog is displayed.

   b) In **Store Location**, select the **Local Machine** radio button.

   c) Click **Next**.

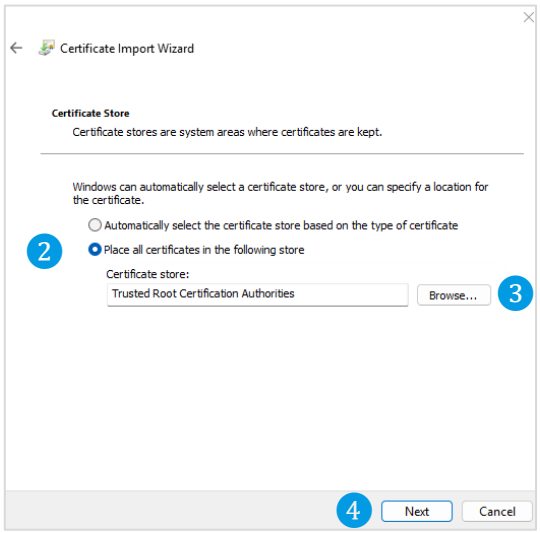



   d) In the Certificate Store dialog, select the **Place all certificates in the following store** radio button 2.

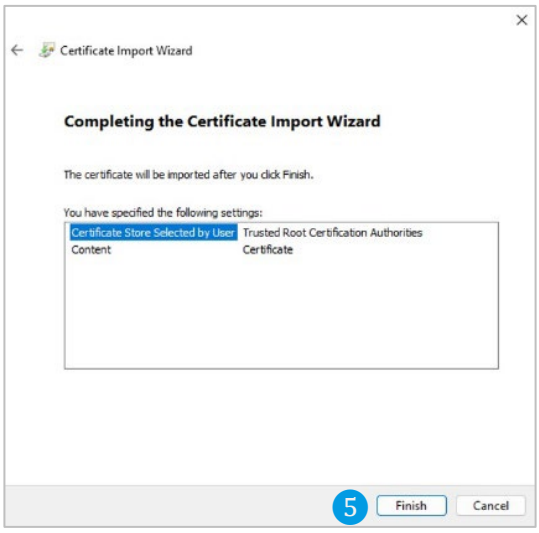
   e) In **Certificate store**, click **Browse…** 3 and navigate to Trusted Root Certification Authorities.

   f) Click **Next** 4.

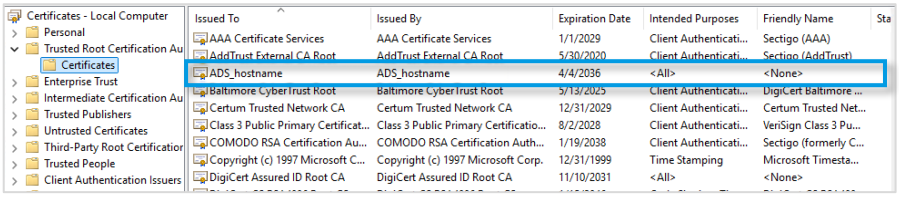
      The Completing the Certificate Import Wizard dialog is displayed.



g) At the bottom of the Completing the Certificate Import Wizard panel, click **Finish** 5.




In Certmgr, a certificate is created under the **Trusted Root Certificates Authority > Certificates** folder:




**By running a PowerShell cmdlet**

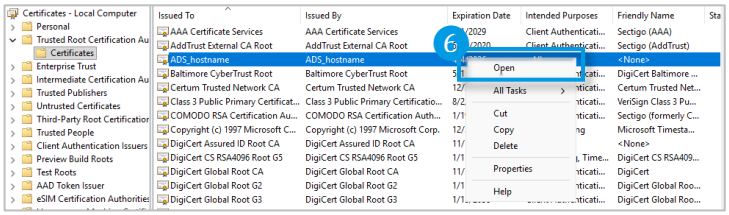
- You can also import a certificate by running a cmdlet from PowerShell, launched with administrative privileges:

```
Import-Certificate -FilePath "<PATH_TO_CERTIFICATE\<ADS_HOSTNAME>.crt" -
CertStoreLocation Cert:\LocalMachine\Root
```

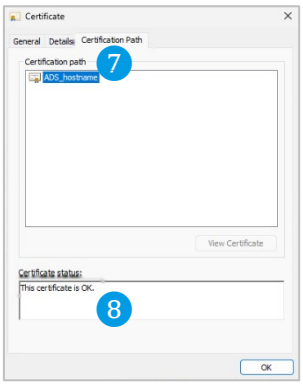
**2. Verify certificate generation**

To verify if self-signed certificate has been generated and installed properly:

a) In Certmgr, select **Trusted Root Certification Authorities** > **Certificates**, right-click the <ADS_hostname> file and select **Open** 6.




b) In the Certificate dialog, opposite, under the **General** tab, verify the certificate expiration date and, under the **Certification Path** tab 7, confirm that the certification path reports a valid trust chain 8.




**3. Download and prepare Certificate files**

a) [Recommended] Store certificates and private key files under `/opt/refinitiv/SOFTWARE/globalconfig`, in a separate subfolder, `certificates`, for example [5].

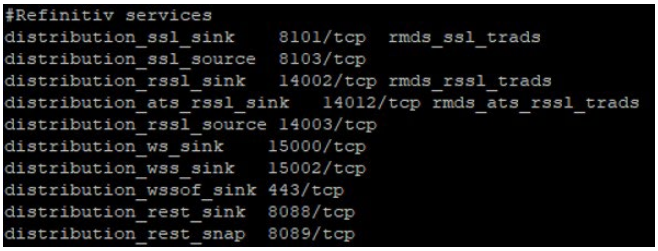
b) Copy the certificate and private key files to the sub-folder[6].

To enable WebSocket Secure protocol in ADS and RTC, refer to Enable WebSocket Secure listening port in ADS/RTC.

[5] This subfolder can also be hidden.
[6] Each signed certificate and the private key it is based on must be copied to every machine running ADS.

# Enabling the WSS listening port

This section describes the processes that are used to enable the WSS listening port on ADS and RTC.




## Configuring the port on ADS

To configure the WSS listening port on ADS, perform the steps below. For line placement guidance, refer to the illustration, opposite:

1. Create a backup copy of your current `/opt/refinitiv/SOFTWARE/globalconfig/distribution.cnf` file.
2. Open `$DISTRIBUTION_CONFIG` in a text editor, then do the following:
   a) After the `*ads*wsPort` parameter(s), insert:

   ```
   *ads*wsPortList : distribution_ws_sink,distribution_wss_sink
   ```

   b) After the `*ads*distribution_ws_sink*zlibCompressionLevel` parameter line, insert the following (WSS) parameters:

   ```
   *ads*distribution_wss_sink*applicationIdName : applicationId
   *ads*distribution_wss_sink*authTokenName : AuthToken
   *ads*distribution_wss_sink*cipherSuite :
   *ads*distribution_wss_sink*clientToServerPings : True
   *ads*distribution_wss_sink*compressionType : 0
   *ads*distribution_wss_sink*connectionType : 1
   *ads*distribution_wss_sink*dhParams :
   *ads*distribution_wss_sink*flushInterval : 1
   *ads*distribution_wss_sink*guaranteedOutputBuffers : 200
   *ads*distribution_wss_sink*inputBias : 10
   *ads*distribution_wss_sink*interfaceName :
   *ads*distribution_wss_sink*maxMounts : 256
   *ads*distribution_wss_sink*maxOutputBuffers : 4000
   *ads*distribution_wss_sink*numInputBuffers : 100
   *ads*distribution_wss_sink*outputThresholdBreach : 60
   *ads*distribution_wss_sink*outputThresholdOK : 30
   *ads*distribution_wss_sink*pingTimeout : 30
   *ads*distribution_wss_sink*poolSize : 300000
   *ads*distribution_wss_sink*positionName : AuthPosition
   *ads*distribution_wss_sink*reusePort : False
   *ads*distribution_wss_sink*serverCert :
   /opt/refinitiv/SOFTWARE/globalconfig/Certs/adshost1.crt
   *ads*distribution_wss_sink*serverPrivateKey : /opt/refinitiv/SOFTWARE/globalconfig
   *ads*distribution_wss_sink*serverToClientPings : True
   *ads*distribution_wss_sink*tcpRecvBufSize : 64240
   *ads*distribution_wss_sink*tcpSendBufSize : 64240
   *ads*distribution_wss_sink*timedWrites : True
   *ads*distribution_wss_sink*zlibCompressionLevel : 3
   ```

3. Update the Certificate and Private Key path to match your deployment environment, for example:

   ```
   *ads*distribution_wss_sink*serverCert : /opt/refinitiv/SOFTWARE/globalconfig
   /certificates/adshost1.crt
   *ads*distribution_wss_sink*serverPrivateKey : /opt/refinitiv/SOFTWARE/globalconfig
   /certificates/adshost1.key
   ```

4. Open the `/etc/services` file in a text editor, with **sudo** rights, and, after the `distribution_ws_sink` line, insert the following line:

   ```
   distribution_wss_sink    15002/tcp
   ```

   Ensure that suggested port number, `15002/tcp`, is unique across that system file. Otherwise, use next available port number in this range (that is, 15003,15004, and so on).

📌 Note that non-root processes like ADS should be enabled to use privileged port numbers below 1024 on Linux OS level.

5. Restart the ADS process with the following commands:

```
mdsStop ads
mdsStart ads
```

   Or using the equivalent **sudo** commands:

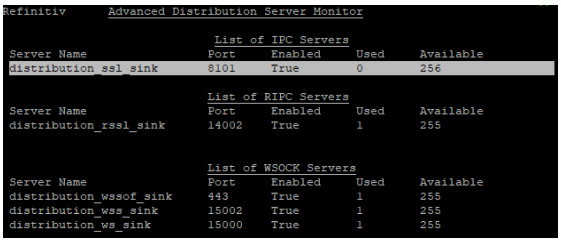
```
sudo systemctl stop ads
sudo systemctl start ads
```

6. To validate WSS has been properly configured, do the following in a terminal connected to your ADS or RTC machine:

   a) Run the `adsmon` command.

   b) In the main menu, navigate to the **Network Statistics** page, under the **Sink IPC Statistics** menu.

   c) Check whether the `Enabled` value for `distribution_wss_sink` is **True**, confirming ADS is listening on the WebSocket Secure port.




# Configuring the port on RTC

To configure the WSS listening port on RTC, do the following:

1. Create a backup copy of the current `/opt/refinitiv/SOFTWARE/globalconfig/RTC_client.cnf` file.
2. Open `RTC_client.cnf` in a text editor, then do the following:

   a) After the `*rtc*wsPort` parameter statement, insert the following:

```
*rtc*wsPortList : distribution_ws_sink,distribution_wss_sink
```

   b) At the end of `WebSocket Config` section, insert the following:

```
*rtc*distribution_wss_sink*compressionType : 1
*rtc*distribution_wss_sink*serverToClientPings : True
*rtc*distribution_wss_sink*clientToServerPings : True
*rtc*distribution_wss_sink*pingTimeout : 30
*rtc*distribution_wss_sink*minPingTimeout : 6
*rtc*distribution_wss_sink*interfaceName :
*rtc*distribution_wss_sink*reusePort : False
*rtc*distribution_wss_sink*connectionType : 1
*rtc*distribution_wss_sink*serverCert : <PATH_TO_CERT>
*rtc*distribution_wss_sink*serverPrivateKey : <PATH_TO_PRIV_KEY>
*rtc*distribution_wss_sink*maxOutputBuffers : 4000
*rtc*distribution_wss_sink*poolSize : 300000
```

   c) In the Automatic Login for Authentication section, insert the following:

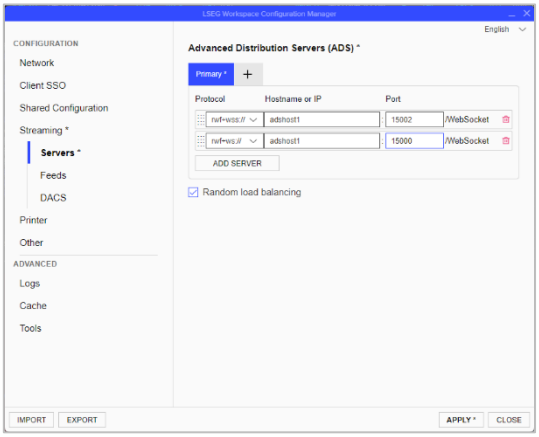
```
*rtc*distribution_wss_sink*positionName : AuthPosition
*rtc*distribution_wss_sink*applicationIdName : applicationId
*rtc*distribution_wss_sink*authTokenName : AuthToken
```

3. WSS enablement post-checks for RTC are performed using the same approach as described for ADS.

# Configuring WSS in LSEG Workspace

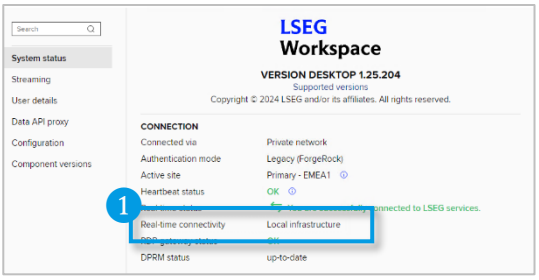
To configure LSEG Workspace to use the WSS connection protocol, follow the steps below:

1. In LSEG Workspace, select **Help > Configuration Manager**.
2. Select **Streaming** and choose the **From a local LSEG Real-Time Advanced Distribution Server (ADS)** radio button.
3. From the sub-menu that is now shown, select **Servers**.
4. In the Advanced Distribution Servers (ADS) panel, click **ADD SERVER**.
5. Configure the servers as they were defined when creating the signed certificate.
6. Click **APPLY**.

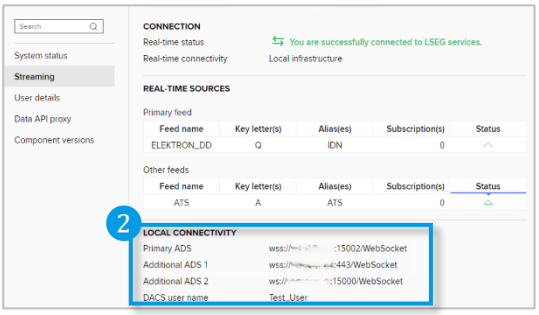



To verify your settings:

1. Restart LSEG Workspace and select Help > About LSEG Workspace.
2. Under the **System Status** tab, check whether LSEG Workspace is connected to **Local infrastructure** 1.




3. Under the **Streaming** tab, in the Local Connectivity section, check that:
   a) the connection to ADS is using the WSS protocol 2.
   b) The FQDNs or IP addresses must match the SANs used when creating certificates.



# Troubleshooting

Should you experience issues with the configuration of WSS, to verify your connection, follow the steps below:

1. Verify that the WSS port is accessible from the workstation running LSEG Workspace, using telnet or Windows PowerShell utilities, by running the following statements:
   - `C:\>telnet adshost1 15002`, or
   - `PS C:\>tnc adshost1 -Port 15002`
2. Run the `testclient` utility to ensure that ADS accepts mounts on WSS port:

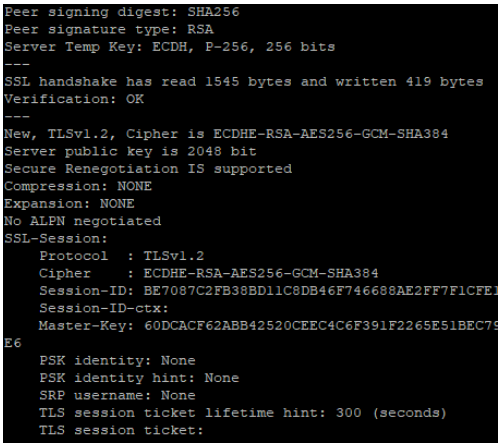
   ```
   ./testclient -h <HOSTNAME> -S <SERVICE_NAME> -ct ws_rwf -il EUR= -t 0 -T -p 15002 -sslCAStore
   <PATH_TO_ROOT_CA_CERT> -encPro 4 -tunnel ssl
   ```

   To run the test above, the Root CA certificate must be copied to the machine running ADS machine. For:
   - Certificates obtained from either Public or Enterprise CA, the Root CA certificate can be exported from the Trusted Certificate store in the Windows Certificate Manager.
   - Self-signed Certificates, the Root CA certificate must be created locally using the OpenSSL tool. For details, refer to the Creating a single certificate for multiple ADS machines section.
3. Verify certificate details, using the following command:

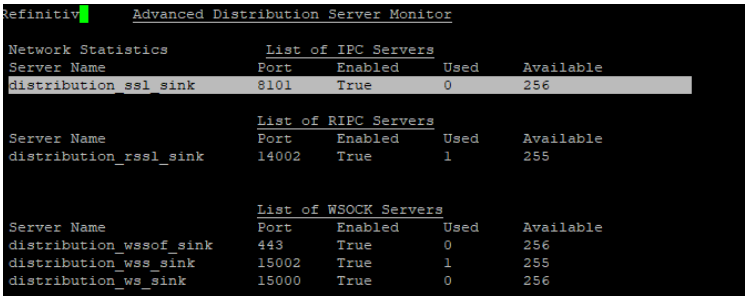
   ```
   openssl x509 -in adshost1.crt -text -noout
   ```
4. Attempt to connect to the ADS host, using the following command:

   ```
   openssl s_client -connect <ADS_hostname>:15002 -cert <ADS_hostname>.crt -CAfile <ADS_hostname>_CA.pem
   ```

   If the connection is successful, the command, above, produces similar output to the screenshot, opposite:




Following successful connection, using the adsmon utility, you will find confirmation of the new WSS mount on the **Network Statistics** page, under the **Sink IPC Statistics** menu:



# Glossary of terms and acronyms

| Term | Definition |
|---|---|
| Active Directory Certificate Services (AD CS) | Windows Server role for issuing and managing public key infrastructure (PKI) certificates used in secure communication and authentication protocols. |
| ADS | Advanced Distribution Server |
| Certificate | A file that attests to the identity of an organisation or web browser user and is used to verify that data being exchanged over a network is from the intended source. The certificate is digitally signed either by a Certificate Authority (CA) or is self-signed. There are CA certificates, client CA certificates, client certificates, and server certificates. |
| Certification Authority (CA) | A third-party organisation, which is used to confirm the relationship between a party to the https transaction and the public key of that party. |
| Certificate Signing Request (CSR) | An encoded text file generated on a server, containing public key infrastructure (PKI) information like common name (domain), organization, and location, along with a public key. |
| Common Name (CN) | The Fully Qualified Domain Name (FQDN) you want to secure, such as www.example.com or example.com. |
| Digital certificate files | The following file formats and extensions are used:<br>• KEY, generally the private key.<br>• CRT, contains a single signed certificate.<br>• PEM, generally the public key, but it can also contain one or more certificates and, optionally, a private key. |
| Encryption | Encryption is the process of changing data into a form that can be read only by the intended receiver. |
| FQDN | Fully Qualified Domain Name |
| OpenSSL | Open-source software library and command-line toolkit that provides a robust set of cryptographic functions. |
| Private Key | A private key is a highly secure, secret alphanumeric code used in asymmetric cryptography to decrypt data and generate digital signatures. |
| Public Key | A public key is a freely distributable cryptographic key used in asymmetric cryptography to encrypt data and verify digital signatures. It forms a pair with a corresponding private key, which must be kept secret to decrypt messages or sign documents. |
| Root Certificate | A self-signed certificate issued from a root level Certificate Authority (CA). |
| RTC | Real-Time Connector (formerly, ADS PoP) |
| RTDS | Real-Time Distribution System |
| RSA | Rivest-Shamir-Adleman (RSA) Algorithm is an asymmetric or public-key cryptography algorithm which means it works on two different keys: Public Key and Private Key.<br>• The Public Key is used for encryption and is known to everyone.<br>• The Private Key is used for decryption and must be kept secret by the receiver. |
| Subject Alternative Name (SAN) | Flexible extension that allows a single certificate to secure multiple hostnames, domains, and subdomains. |