Configuring Secure MQTT Communication

MQTT Distributor can be enabled to use TLS to encrypt the communication between MQTT clients. This is useful if MQTT Distributor is used on a public network. Since MQTT communications are not encrypted by default, enabling TLS is highly recommended on a public network.

Reusing Ignition SSL Certificates

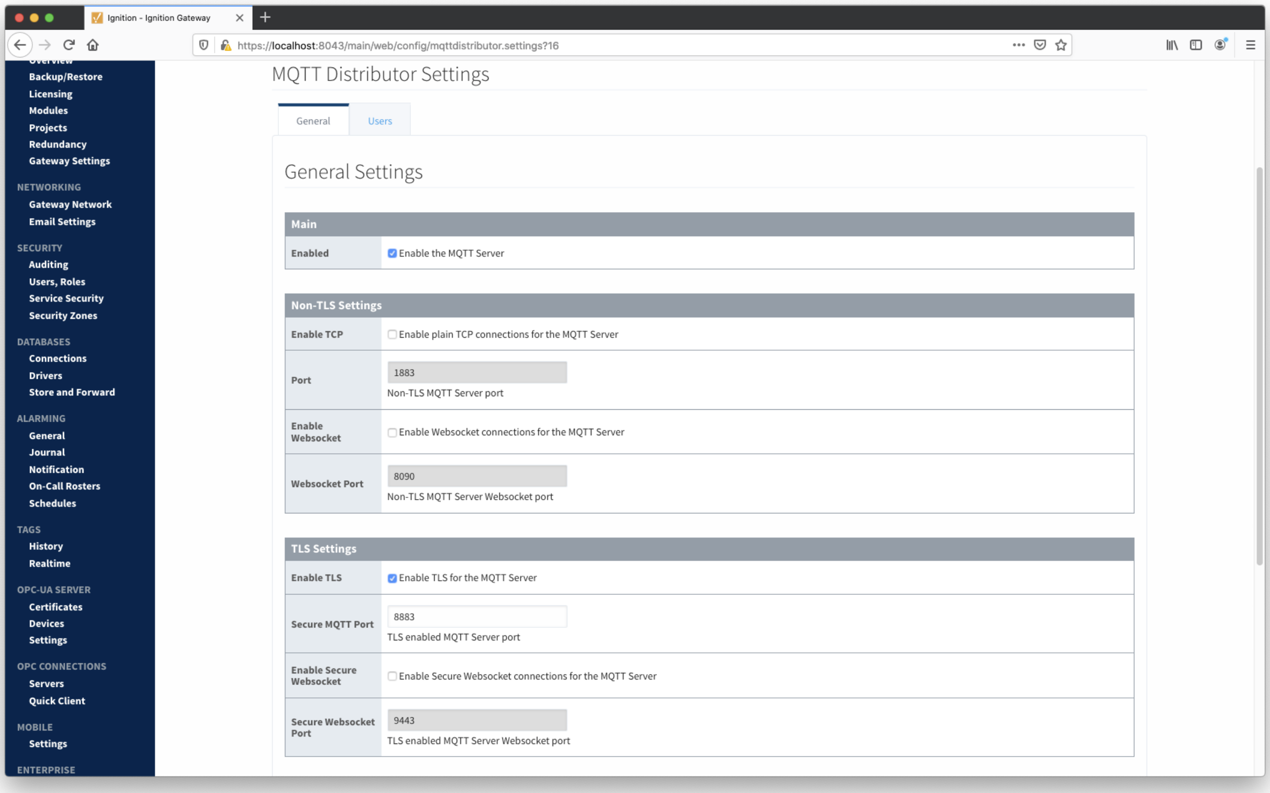
As of module release version **3.4.7**, the Cirrus Link MQTT Distributor module is capable of reusing the existing Ignition web server SSL certificates to secure your MQTT communication. This is the recommended process to secure your MQTT communication using SSL/TLS. If using modules versioned before 3.4.7, the details **here** will help you through this process.

SSL/TLS Enable the Ignition Web Server

Follow the steps outlined **here** to SSL enable the Ignition Web Server.

Configure MQTT Distributor to use SSL/TLS

Once the Ignition Web Server has been SSL enabled, enable SSL/TLS for MQTT Distributor by checking the "Enable TLS" configuration setting under ConfigMQTT DistributorSettingsGeneralTLS Settings. Click Save to confirm the configuration update.



General Settings

Main	
Enabled	☑ Enable the MQTT Server

Non-TLS Settings	
Enable TCP	☐ Enable plain TCP connections for the MQTT Server
Port	1883 Non-TLS MQTT Server port
Enable Websocket	☐ Enable Websocket connections for the MQTT Server
Websocket Port	8090 Non-TLS MQTT Server Websocket port

TLS Settings	
Enable TLS	☑ Enable TLS for the MQTT Server
Secure MQTT Port	8883 TLS enabled MQTT Server port
Enable Secure Websocket	☐ Enable Secure Websocket connections for the MQTT Server
Secure Websocket Port	9443 TLS enabled MQTT Server Websocket port

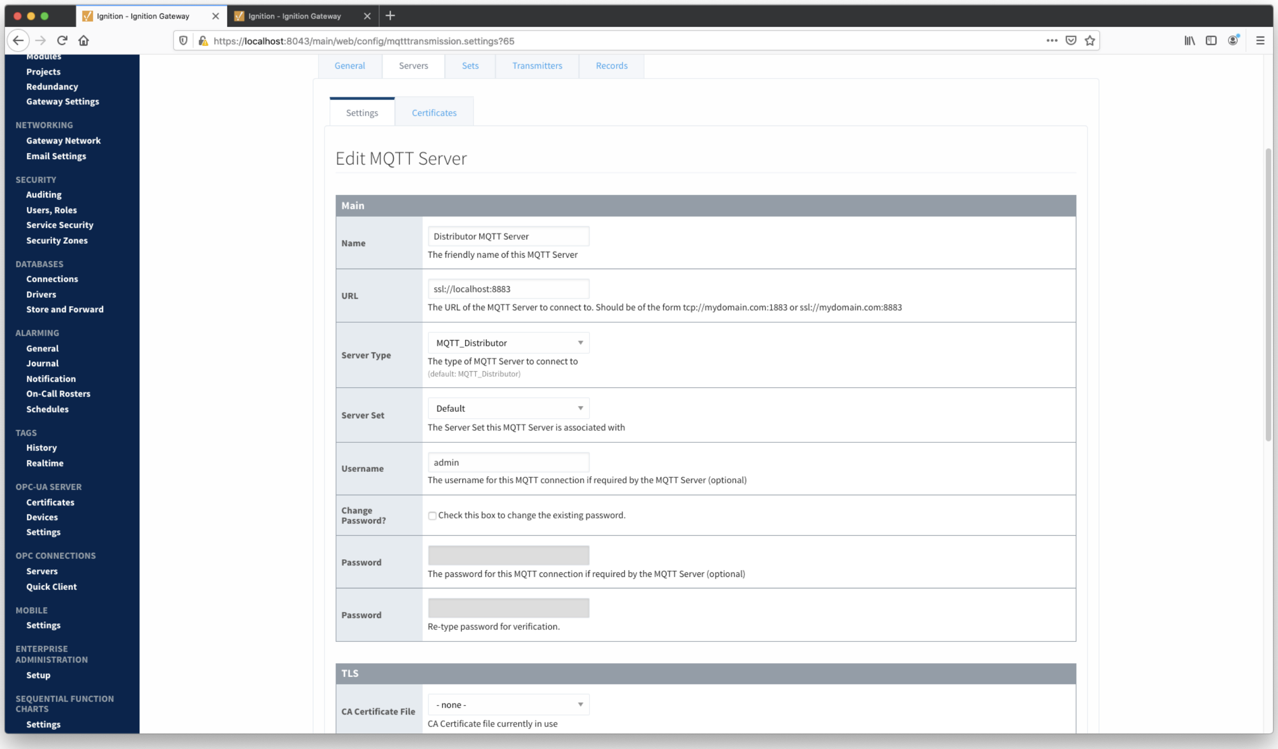
Configure MQTT Engine and Transmission to use SSL/TLS

Once TLS has been enabled for MQTT Distributor, the only change required* for MQTT Engine and MQTT Transmission to connect to Distributor over SSL /TLS, is to update the MQTT Server URL. Update the following MQTT Server URL configuration settings with the appropriate MQTT Server URL for your environment. For example, 'ssl://mqttserver:8883'

SSL/TLS Configuration Settings:

- ConfigMQTT EngineSettingsServers->Settings[Edit:Your MQTT Server]MainURL
- ConfigMQTT TransmissionSettingsServers->Settings[Edit:Your MQTT Server]MainURL

The screenshot below shows MQTT Transmission configured for SSL/TLS. Configure MQTT Engine to use SSL/TLS in the same way.



Edit MQTT Server

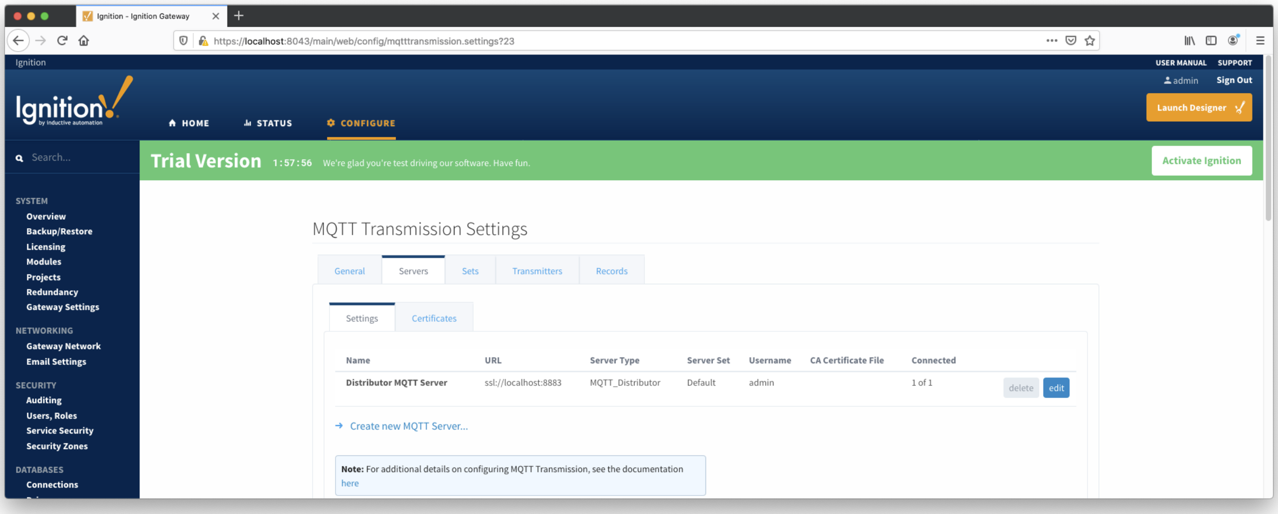
Main

Name	Distributor MQTT Server The friendly name of this MQTT Server
URL	ssl://localhost:8883 The URL of the MQTT Server to connect to. Should be of the form tcp://mydomain.com:1883 or ssl://mydomain.com:8883
Server Type	MQTT_Distributor The type of MQTT Server to connect to (default: MQTT_Distributor)
Server Set	Default The Server Set this MQTT Server is associated with
Username	admin The username for this MQTT connection if required by the MQTT Server (optional)
Change Password?	☐ Check this box to change the existing password.
Password	The password for this MQTT connection if required by the MQTT Server (optional)
Password	Re-type password for verification.

TLS

CA Certificate File	- none - CA Certificate file currently in use

At this point, MQTT Engine and MQTT Transmission should show they're connected to MQTT Distributor over SSL/TLS.

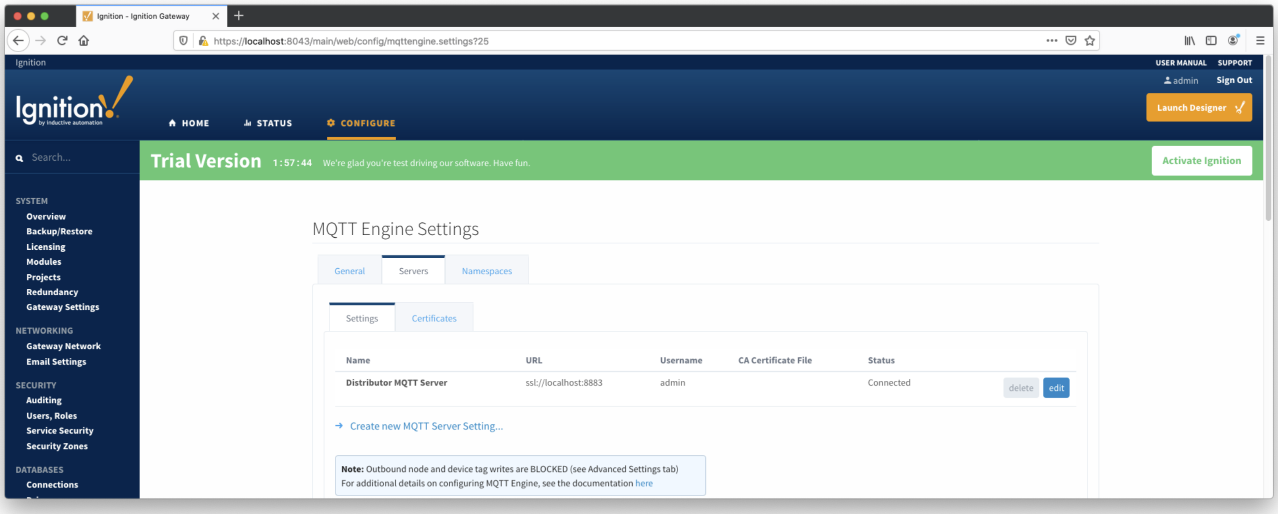


MQTT Transmission Settings

Name	URL	Server Type	Server Set	Username	CA Certificate File	Connected	
Distributor MQTT Server	ssl://localhost:8883	MQTT_Distributor	Default	admin		1 of 1	delete edit

→ Create new MQTT Server...

Note: For additional details on configuring MQTT Transmission, see the documentation here



MQTT Engine Settings

Name	URL	Username	CA Certificate File	Status
Distributor MQTT Server	ssl://localhost:8883	admin		Connected

→ Create new MQTT Server Setting...

Note: Outbound node and device tag writes are BLOCKED (see Advanced Settings tab)
For additional details on configuring MQTT Engine, see the documentation here

If running pre-3.4.7 modules, your Ignition web server is not SSL/TLS enabled, you're using self-signed certificates or the default workflow above did not work as expected, read on to see notes and variations on the standard process for enabling SSL/TLS.

Configuration Variations

Using a Java Keystore File with Distributor

This step should only be necessary if you're running pre-3.4.7 modules. The steps below will show how to create a Java keystore (JKS) containing all appropriate certificates and how to configure MQTT Distributor to use this keystore.

Convert Ignition's Keystore

If running pre-3.4.7 modules and your Ignition web server **is** SSL/TLS enabled, you can create the necessary Java keystore (JKS) file from the existing Ignition keystore (<Ignition_Install>\webserver\ssl.pfx). This can be done easily using the KeyStore Explorer tool to convert the Ignition keystore of type PKCS #12 to a Java keystore of type JKS. The details **here** will help you through this process.

Create a Java Keystore

If running pre-3.4.7 modules and your Ignition web server **is not** SSL/TLS enabled, you will need to create a Java keystore from scratch using the KeyStore Explorer tool. The details **here** will help you through this process.

Using Self-signed Certificates

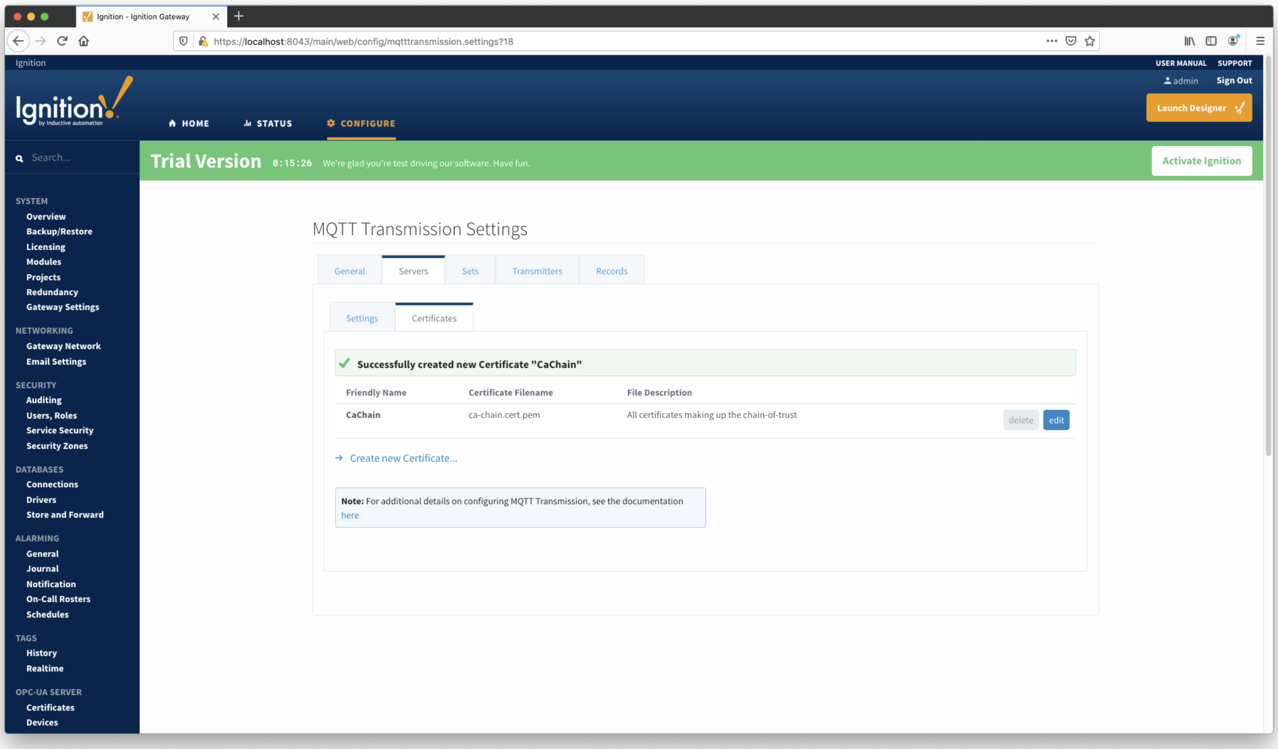
If using self-signed certificates in your environment, the steps to enable SSL/TLS are identical to the default workflow with the additional requirement that the certificate chain (aka. "chain-of-trust") be available to MQTT Engine and Transmission. When using self-signed certificates, the required CA certificates are not known to MQTT clients by default as they would be if the certificate was generated by a real CA and the CA certificate was provided by Java's default keystore. Therefore, MQTT Engine and Transmission must be configured to use the appropriate certificate chain.

Identify the certificate chain

The certificate chain (aka. "chain-of-trust") is a collection of the public root CA (Certificate Authority) certificate and any/all public intermediate CA certificates between the root and the CA that issued the certificate. If there are no intermediate CAs, then the chain is made up of only the public root CA certificate. You will need to configure MQTT Engine and Transmission to trust these CAs by adding their certificates under MQTT Engine and MQTT Transmission configuration. If a single certificate, move to the next step to upload and configure this certificate. If more than one certificate makes up the certificate chain, you will need to copy the contents of each certificate into a single file (in x509 PEM format; give it a name like 'ca-chain.cert.pem') and move to the next step to upload then configure this certificate.

Upload the certificate chain

To upload the certificate chain (aka. "chain-of-trust") to MQTT Engine and MQTT Transmission, launch the Ignition Web Portal, navigate to the "Servers" tab in the module configuration for each module, click on the "Certificates" tab and click 'Create new Certificate' to bring up the creation UI. Next, choose the certificate to upload, give it a friendly name like 'CaChain' and click 'Save'. The two screenshots below show configuration specific to MQTT Transmission. Configure MQTT Engine certificates in the same way.



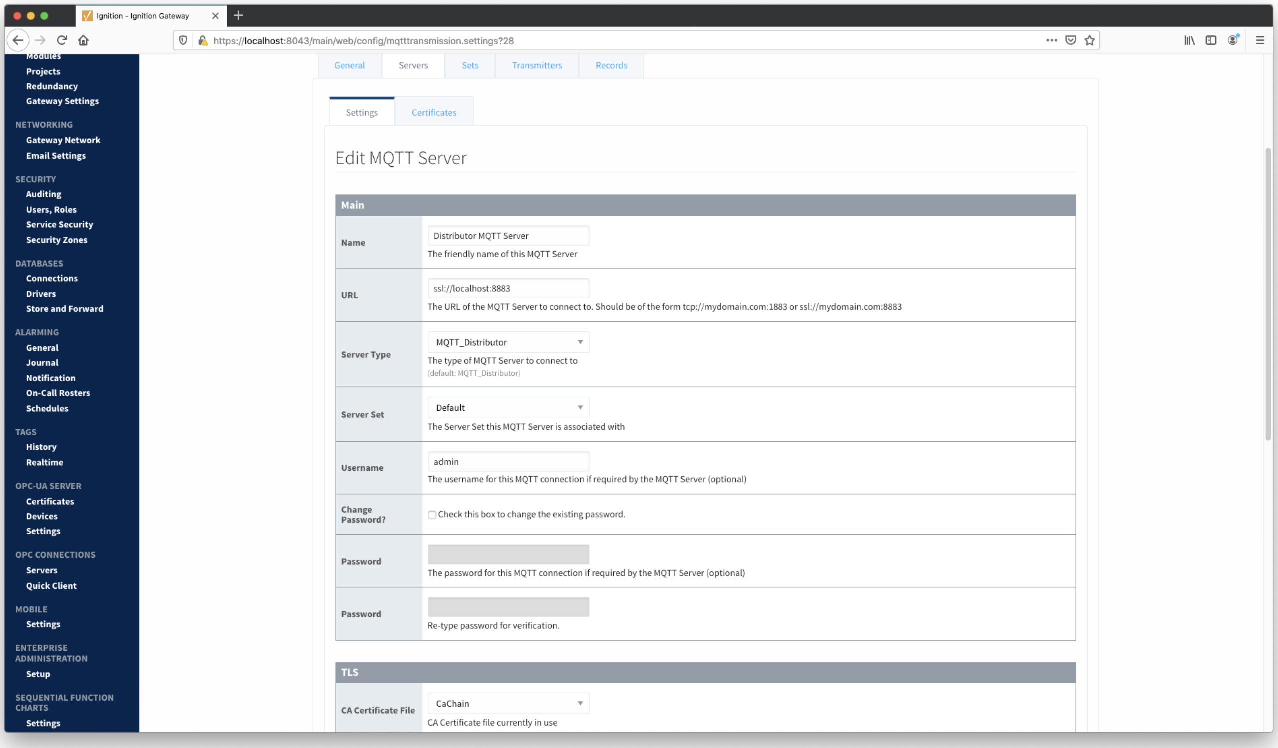
MQTT Transmission Settings

Successfully created new Certificate "CaChain"

Friendly Name	Certificate Filename	File Description
CaChain	ca-chain.cert.pem	All certificates making up the chain-of-trust

Note: For additional details on configuring MQTT Transmission, see the documentation here

Associate the certificate just uploaded to each module by setting the 'CA Certificate File' configuration setting to be equal to the certificate created. Click 'Save'.



Edit MQTT Server

Main	
Name	Distributor MQTT Server The friendly name of this MQTT Server
URL	ssl://localhost:8883 The URL of the MQTT Server to connect to. Should be of the form tcp://mydomain.com:1883 or ssl://mydomain.com:8883
Server Type	MQTT_Distributor The type of MQTT Server to connect to (default: MQTT_Distributor)
Server Set	Default The Server Set this MQTT Server is associated with
Username	admin The username for this MQTT connection if required by the MQTT Server (optional)
Change Password?	☐ Check this box to change the existing password.
Password	The password for this MQTT connection if required by the MQTT Server (optional)
Password	Re-type password for verification.

TLS	
CA Certificate File	CaChain CA Certificate file currently in use

MQTT Engine and Transmission should now show connected to Distributor over SSL/TLS. If the connection is unsuccessful, review the steps in the default workflow to ensure they were completed successfully.

Notes

Getting a Certificate from a Certificate Authority

The first step to securing MQTT communication is to get a certificate from a certificate authority (CA). There are many available such as Verisign, Thawte and RapidSSL. There are also a number of other certificate authorities available. The general process is as follows:

- Generate a RSA key
    - This is the private key and used for encryption/decryption of data. Keep this private and don't share it with anyone including the CA. However, the server (MQTT Distributor) will need it to encrypt/decrypt data.
- Create a Certificate Signing Request (CSR)
    - Generally the CA can provide instructions on how to generate a CSR. Windows, Linux, and OSX all have tools available for generating a CSR and there is lots of documentation online about all of them.
    - Make sure the Common Name specified in the CSR matches the server URL (i.e. example.com). Also, do not include www. because this will used for MQTT.
- Give the CSR to the CA
- The CA will then provide back a public certificate for use with MQTT Distributor
- In some cases depending on the CA an intermediate certificate may also be required. If so, the CA will also provide this.

Creating a Self-Signed Certificate

Creating your own CA, intermediate CA, and generating your own signed certificates can be done following the three steps below using some open source tooling. Note creating an Intermediate CA is not explicitly required, but is recommended if you will be using self-signed certs in a private network in production. If this is simply for development that step can be skipped and the root CA can be used to sign server certificates. Again, using self-signed certs in production over the Internet is not recommended.

- Create the Root Pair
- Create and Sign the Intermediate Pair
- Create and Sign the Server pair
    - Make sure the Common Name specified in the CSR matches the server URL that will be used by the clients (i.e. 192.168.1.100). It could also be the network hostname.