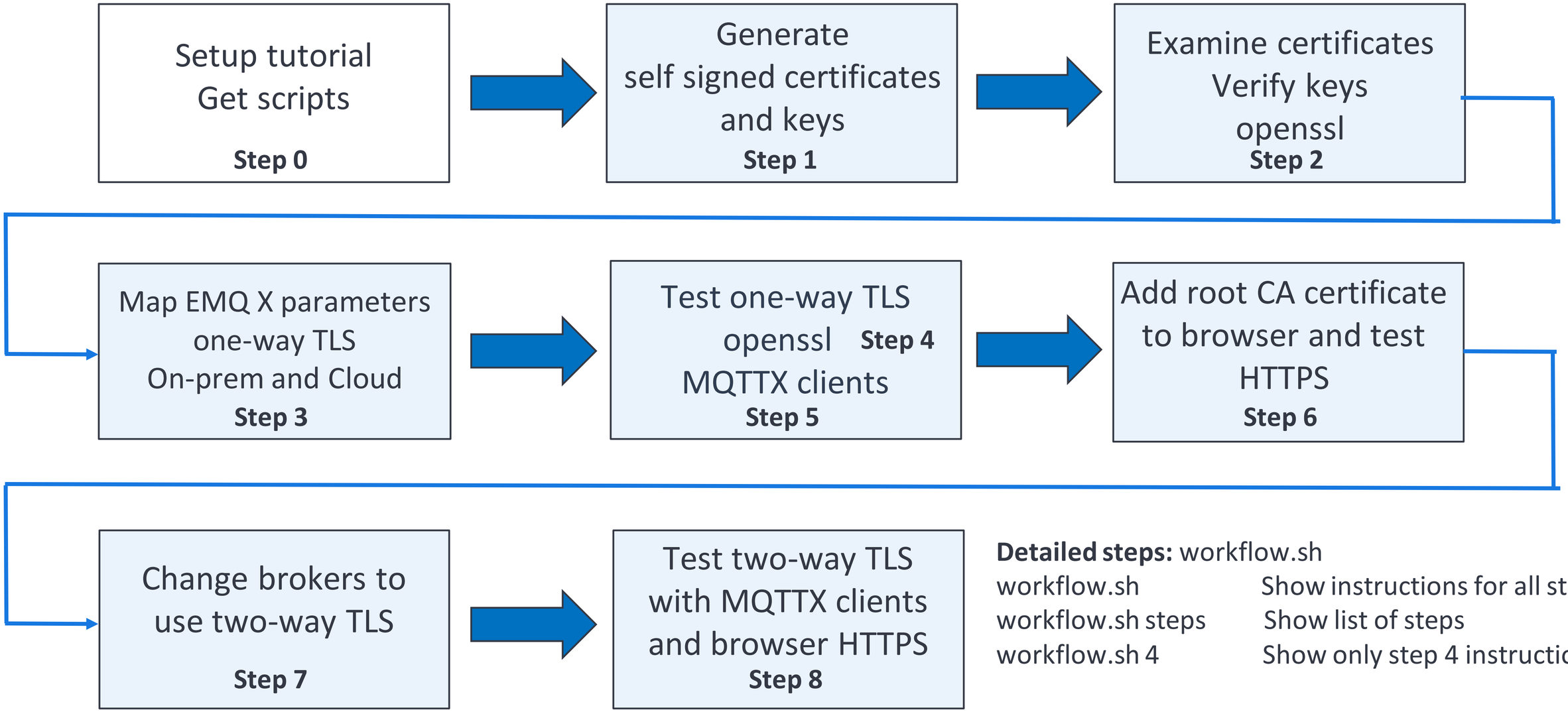
EMQ X Cloud Webinar

Ensure MQTT Security with TLS/SSL in EMQ X
Hands-on Tutorial

February 9th
9:00am EST / 3:00pm CET / 2:00pm UTC

Speaker:
Kary Ware, Sales Engineer @EMQ

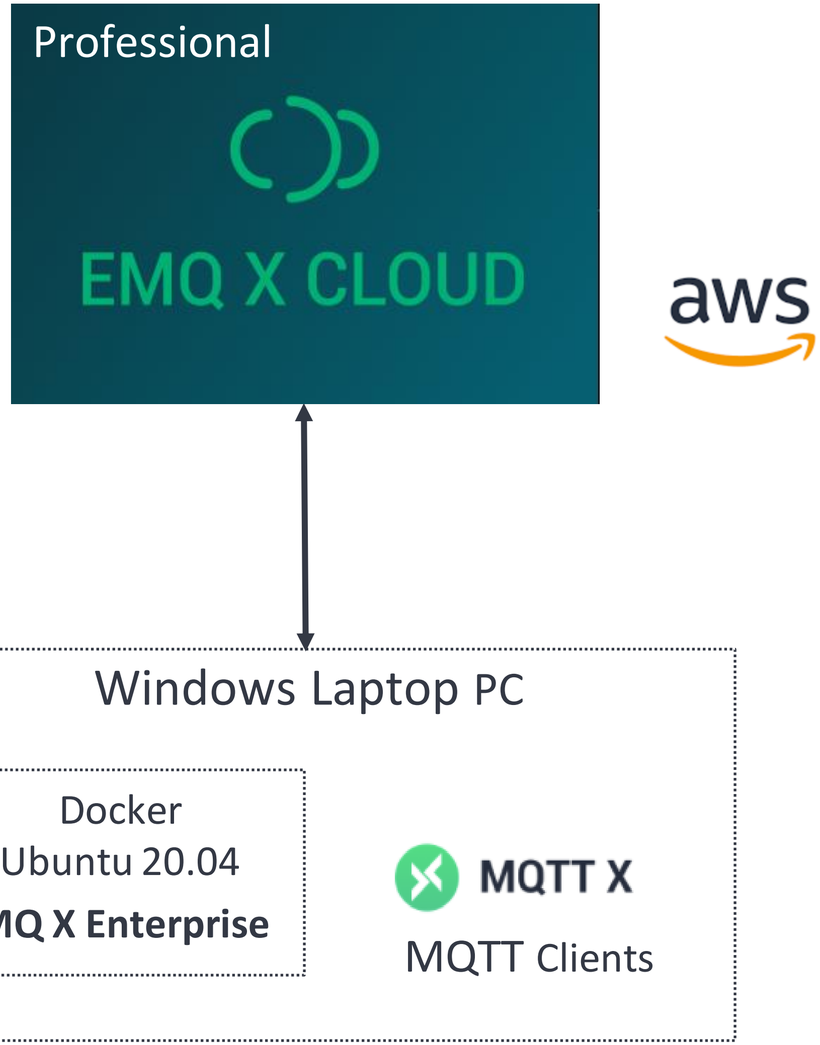
Tutorial Workflow



Detailed steps: workflow.sh

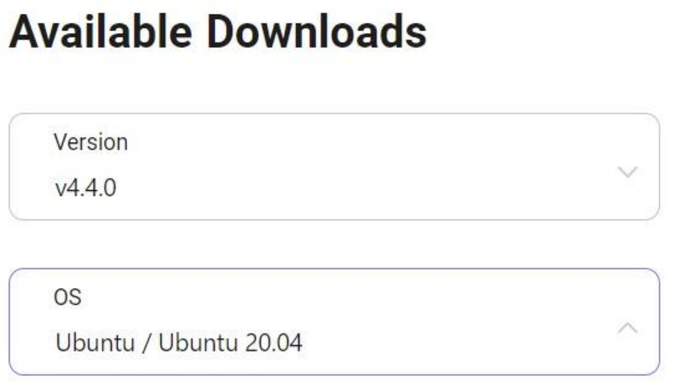
workflow.sh	Show instructions for all steps
workflow.sh steps	Show list of steps
workflow.sh 4	Show only step 4 instructions

Step 0 – Tutorial setup overview



You can also install EMQ X directly on Linux

Docker Ubuntu image
+
EMQ X Ubuntu install



≠

EMQ X Docker Install

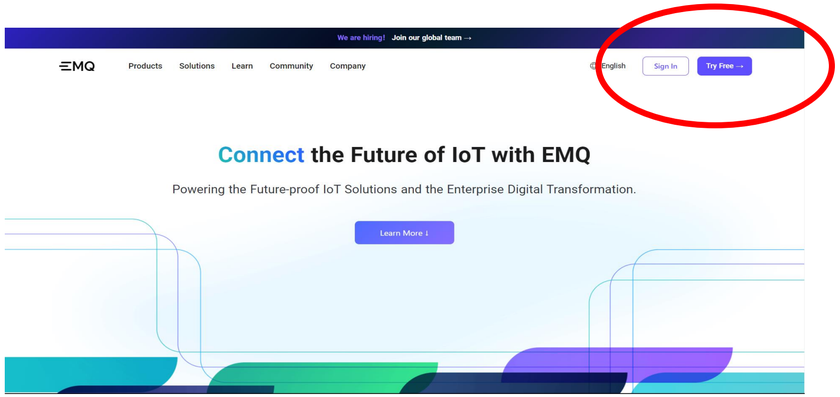
Alpine Linux
Smaller

Get a free trial

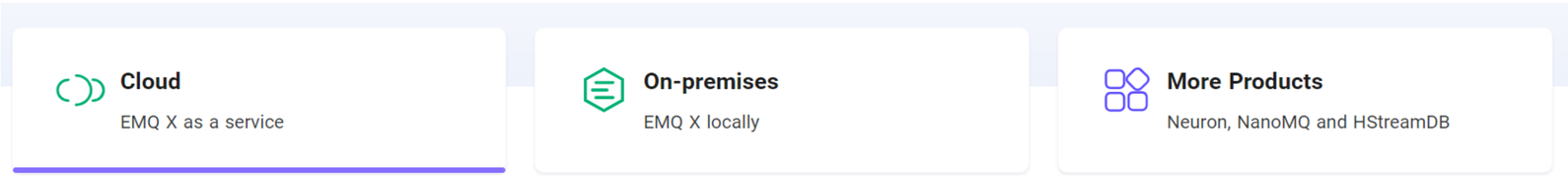
Try Free →

Go to www.emqx.com

and click **Try Free**



Choose **Cloud** or On-Premises



Follow the instructions

You don't need a credit card!

Downloads

To match the setup in the tutorial

Download docker ubuntu 20.04 image

https://hub.docker.com/_/ubuntu

Install EMQ X on-prem and cloud & MQTT X

Install EMQ X Enterprise Ubuntu 20.04

www.emqx.com/en/try?product=enterprise

EMQ X Cloud free trial

https://www.emqx.com/en/try?product=cloud

Install MQTT X https://mqttx.app/

Download tutorial scripts

Download certificate generating scripts and put them in: emqx/etc/certs2

https://github.com/emqx/emqx-webinars/tree/main/2022-02-09-emqx-tls/scripts

Step 1 - Generate self-signed certificates and keys

These are the scripts used to generate the certificates.
generate-certs.sh is the main file that calls the others.

generate-certs.sh
00-generate-root-ca.sh
01-issue-inter-ca.sh (x2)
02-issue-server-cert.sh
03-issue-client-cert.sh

In the demo, generate-certs.sh is run **twice**
Once to generate **local** certificates, and once for **cloud**

The scripts read from environment variables to set the certificate information.

The certificate files are put into separate directories.

local

```
export TLS_CLIENT_COMMON_NAME="172.17.0.1"
export TLS_SERVER_DNS=localhost
```

```
ca.pem  client-fullchain.pem  client.key  client.pem
server-fullchain.pem  server.key  server.pem
```

cloud

```
export TLS_CLIENT_COMMON_NAME="82.181.15.24"
export TLS_SERVER_DNS=s1.....amazonaws.com
```

```
ca.pem  client-fullchain.pem  client.key  client.pem
server-fullchain.pem  server.key  server.pem
```

The demo uses a helper script to set the environment variables

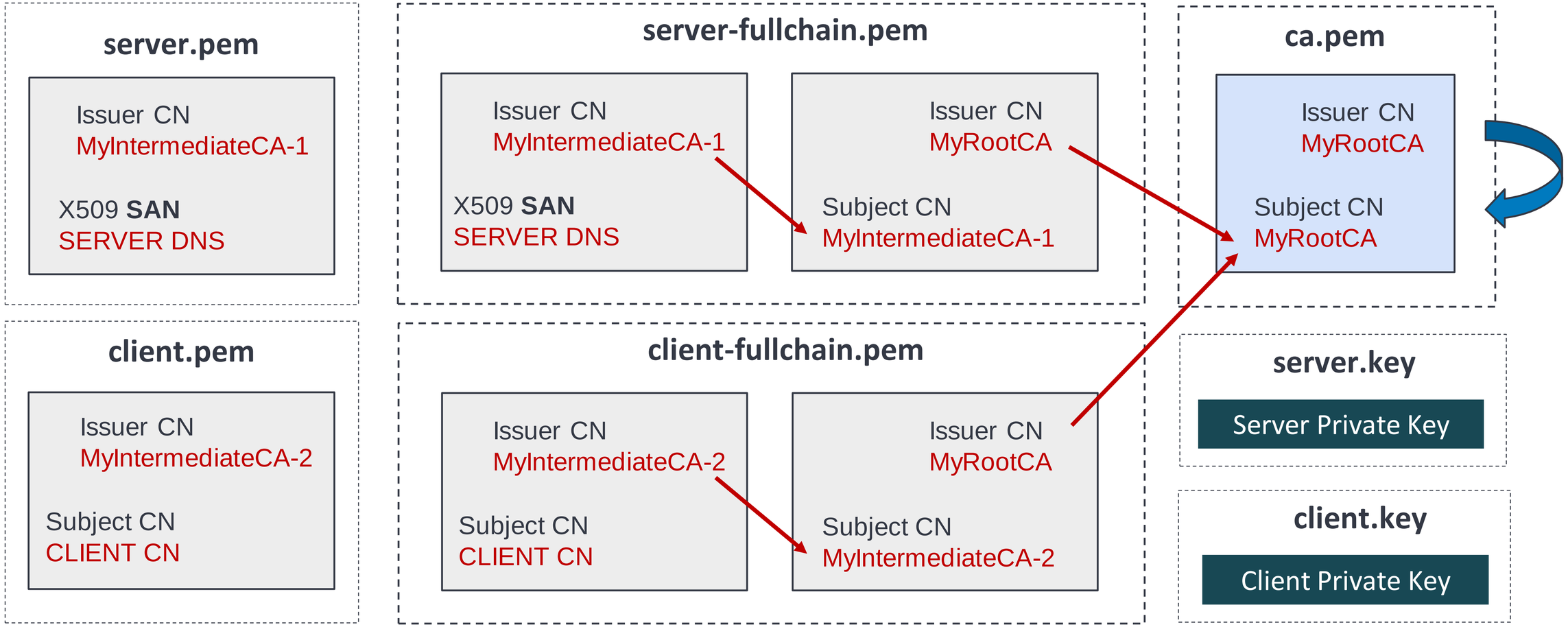
```
source env-config.sh
```

```
source env-config.sh cloud
```

Note: The TLS_SERVER_DNS is put in the X509 extensions Subject Alternate Name and not the subject name

Generated certificate files

generate-certs.sh will generate the following certification files: **local** and **cloud**



	local	cloud
Server DNS	localhost	...amazonaws.com
Client CN	client IP-1	client IP-2

SAN = Subject Alternative Name

Step 2 - Examine certificates and verify keys

These scripts can be used to view and verify the files

```
show.sh    trace.sh   cacerts_show.sh
verify.sh   key-cert-verify.sh
```

Examine the root CA file and verify that EMQ X does not contain MyRootCA

```
../trace.sh ca.pem
../cacerts_show.sh | grep "MyRootCA"
```

Verify the subject, issuer chain is correct including the X509 extensions

```
../trace.sh server-fullchain.pem
../show.sh server-fullchain.pem | grep -iA 4 X509  --color
```

Verify the dates are valid

```
../show.sh server-fullchain.pem | grep -iA 4 X509  --color
```

Verify the keys are valid

```
../verify.sh server
```

Step 3 - Map EMQ X parameters for one-way TLS

On-prem

Edit the **listeners.conf** file for **SSL and WSS**

```
listeners.ssl.external.keyfile = /opt/emqx/etc/certs2/local/server.key
listeners.ssl.external.certfile = /opt/emqx/etc/certs2/local/server-fullchain.pem
listeners.ssl.external.verify = verify_none

listeners.wss.external.keyfile = /opt/emqx/etc/certs2/local/server.key
listeners.wss.external.certfile = /opt/emqx/etc/certs2/local/server-fullchain.pem
listeners.wss.external.verify = verify_none
```

*.verify = **verify_none** means the broker will use one-way TLS

Edit the plugins/**emqx_dashboard.conf** file for **HTTPS**

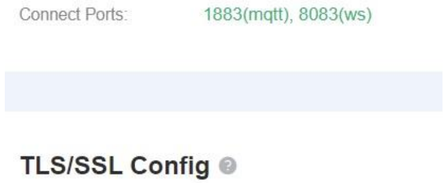
```
dashboard.listener.https.keyfile = /opt/emqx/etc/certs2/local/server.key
dashboard.listener.https.certfile = /opt/emqx/etc/certs2/local/server-fullchain.pem
dashboard.listener.https.verify = verify_none
```

The broker will automatically update the configuration in a few minutes.
You do not need to restart the broker, but you can restart the broker if you want it to update immediately.

```
emqx/bin/emq stop
emqx/bin/emqx start
```

Map EMQ X parameters or one-way TLS

Cloud



The certificate files are mapped in the TLS/SSL Config section as shown.

Make sure to use the certificates in the cloud directory.

Should now see all the ports

Step 4 - Test one-way TLS (using openssl s_client)

These scripts can be used to test the certificates on the running brokers.

```
sclient-ssl.sh
sclient-wss.sh
sclient-https.sh
```

To test **local** certificates
Run in the **local** directory

```
../sclient-ssl.sh
../sclient-wss.sh
../sclient-https.sh
```

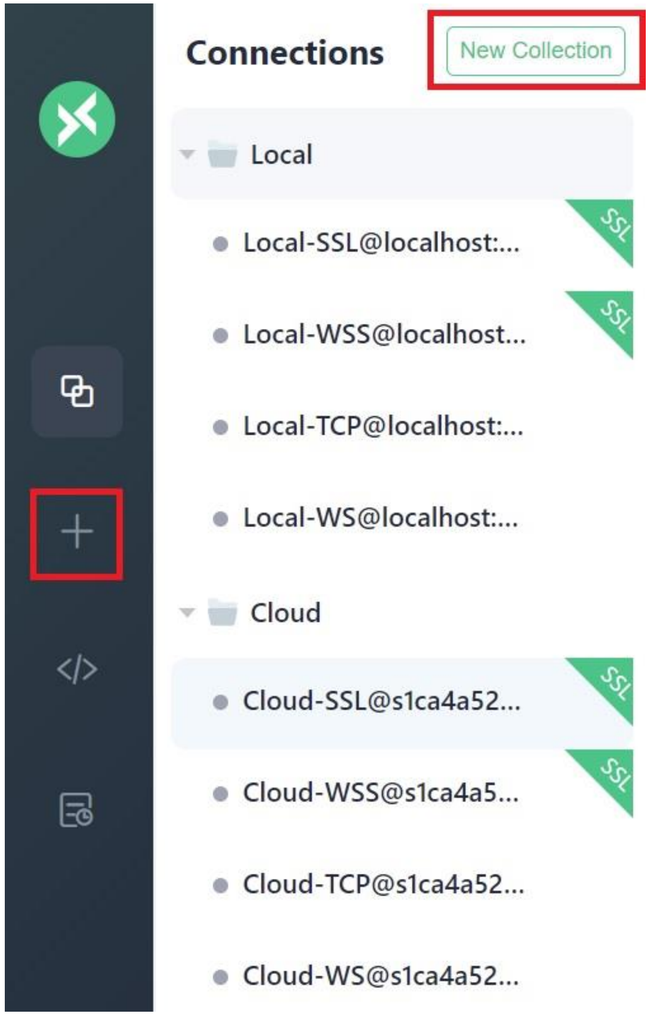
To test cloud certificates
Run in the **cloud** directory and use the cloud parameter

```
../sclient-ssl.sh cloud
../sclient-wss.sh cloud
```

EMQ X Cloud does not use port 18084 for HTTPS

Step 5 - Test one-way TLS using MQTTX

Create Local and Cloud collections.
Create four **Local** clients and four **Cloud** clients. One for each connection type



Connection	Port
mqtt:// (TCP)	1883
mqtts:// (SSL)	8883
ws://	8083
wss://	8084

Connect

Click **Connect** to connect the clients to the broker.

Defaults to EMQ's public broker

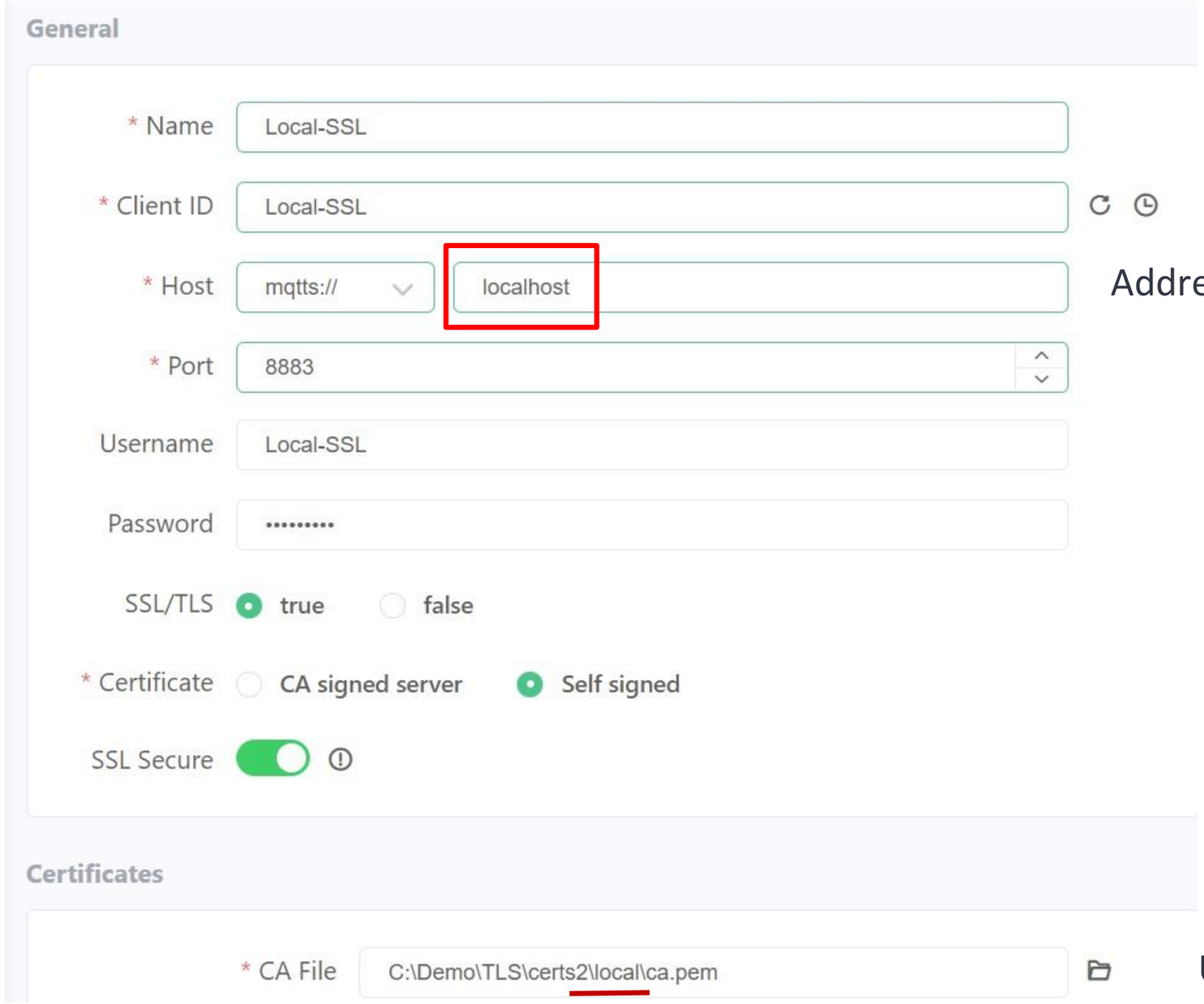
If client connects, but you don't see it in EMQ X, check that it is connected to the correct broker.

mqtts (SSL) and WSS

mqtt (TCP) and WS

Make sure to use the correct ca.pem file

MQTTX local client example

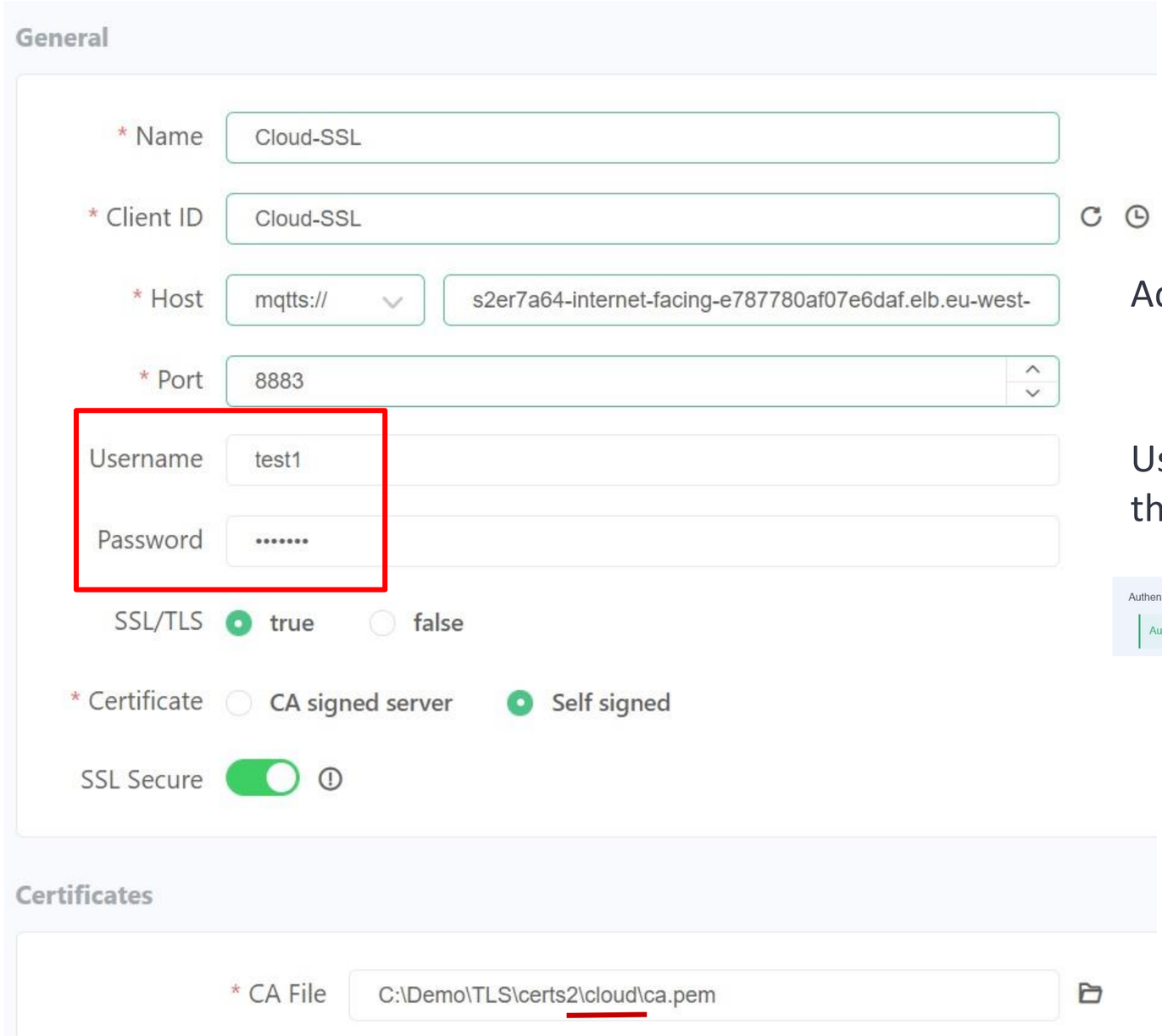


Connect

Address to local on-prem EMQ X broker

Use the local ca.pem file

MQTTX cloud client example



Connect

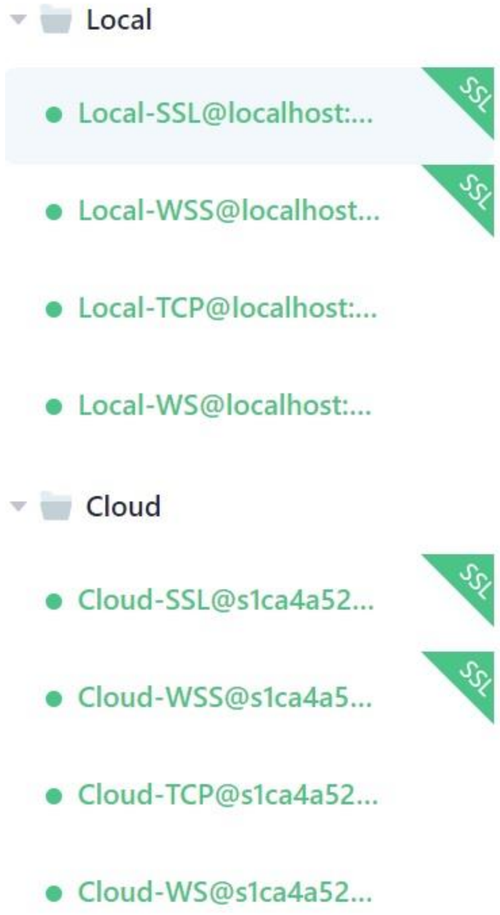
Address to cloud EMQ X broker

Username / password needs to be in the authentication list in EMQ X cloud

Use the cloud ca.pem file

All clients subscribe to testtopic/#

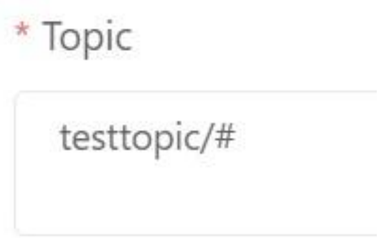
All clients should now be connected and listed in green.



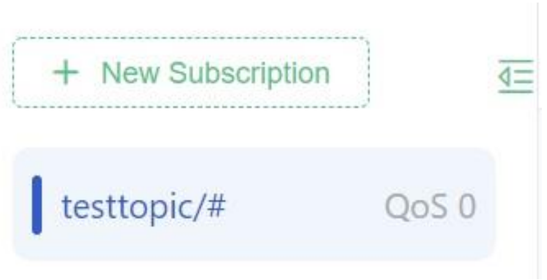
For each client

Click **New Subscription**

Subscribe to the default topic **testtopic** and all subtopics



Each client should now be subscribed to topic **testtopic** and all subtopics



Clients listed in their brokers

EMQ X Cloud

Client connections and subscriptions

Client ID	Topic
Cloud-TCP	testtopic/#
Cloud-WS	testtopic/#
Cloud-SSL	testtopic/#
Cloud4-WSS	testtopic/#

EMQ X on-prem

Client connections and subscriptions

Client ID	Topic
Local-WS	testtopic/#
Local-WSS	testtopic/#
Local-TCP	testtopic/#
Local-SSL	testtopic/#

Publish message

Choose one client in **Local** or **Cloud** group.

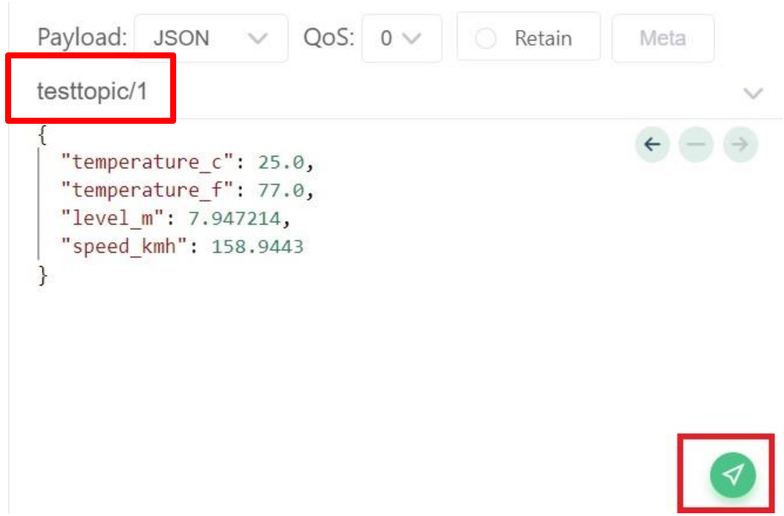
Enter Payload message in JSON format.

Click green button to publish message to testtopic/1

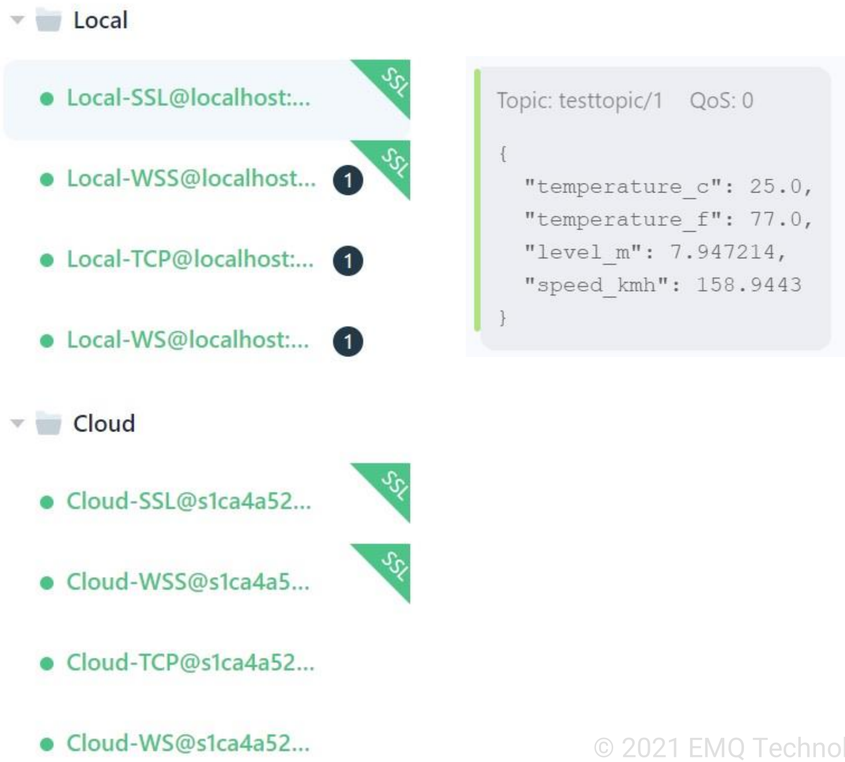
Verify all client**s in the group** received the message

Repeat for one client in the other group

To publishing a message

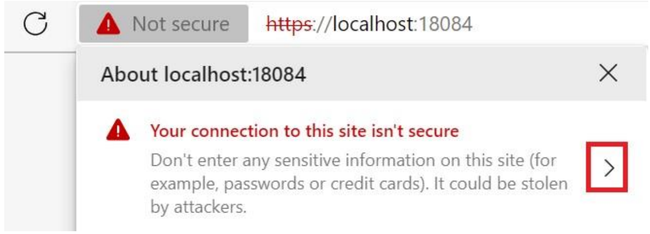


Received message is shown to the right of the selected client

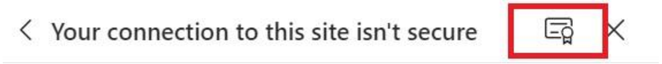


Step 6 - Add server root certificate to browser

Browser: https://localhost:18084
Click **Not Secure**



Edge



Chrome: Certificate is not valid

Click **Certification Path**
Select MyIntermediateCA-a
Click **View Certificate**

Problem is that browser does not have MyRootCA

Issued to: MyIntermediateCA-1

Issued by: MyRootCA

Valid from 2022-01-22 **to** 2032-01-20

Next step is to add the MyRootCA as a trusted certificate...

Add MyRoocCA certificate to Trusted Root CAs

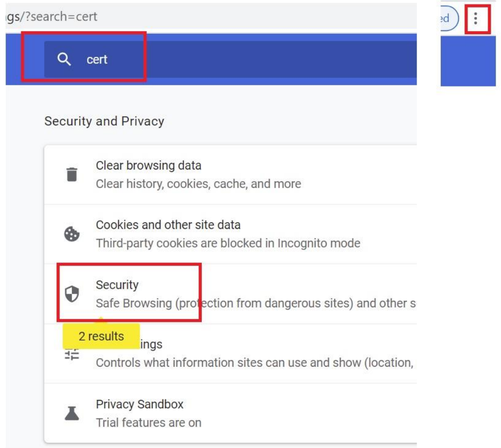
Chrome

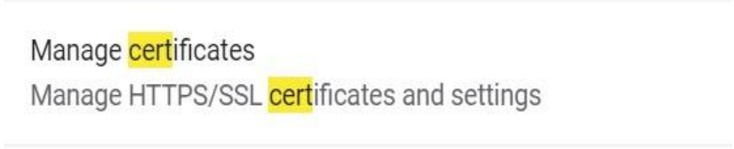
Click three dots

Search "**cert**"

Click **Security**

Scroll down and cli**ck Manage Certificates**





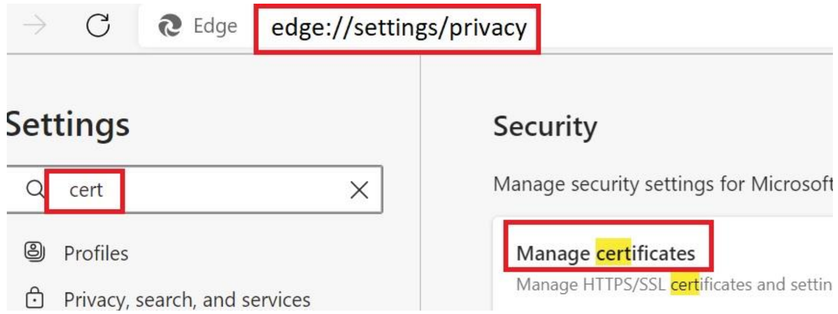
Edge

Address: edge://settings/privacy

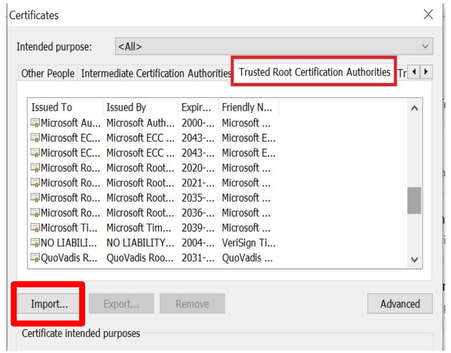
Search "**cert**"

Click **Manage certificates**

Scroll down and click **Manage Certificates**

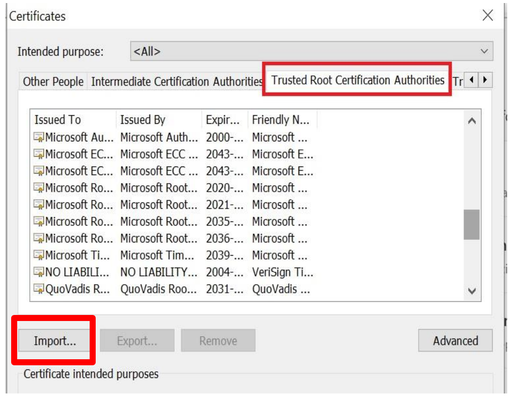


Both browser will bring you to this dialog box

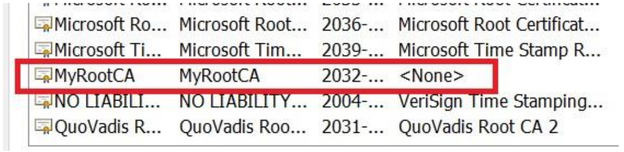


Continued...

Add certificate to Trust CA and test HTTPS



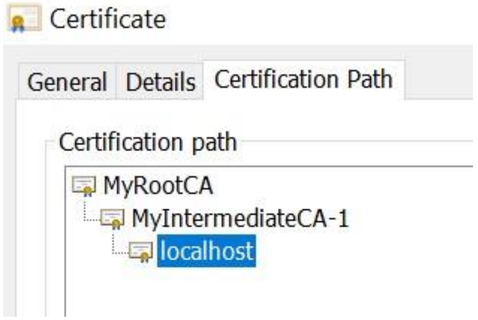
Click **Trusted Root Certificate Authorities**
Verify MyRootCA not included (delete it if it is)

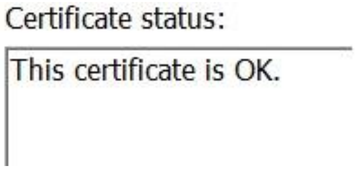


Select Import
Browse to **local/ca.pem**
May need to set to **All Files**

Restart browser
Brows to https://localhost:18084
Verify lock symbol is shown

Verify Certificate Path now includes MyRootCA and that the certificate is valid





Step 7 - Change brokers to use two-way TLS

EMQ X on-prem

Change verify_none to verify_peer

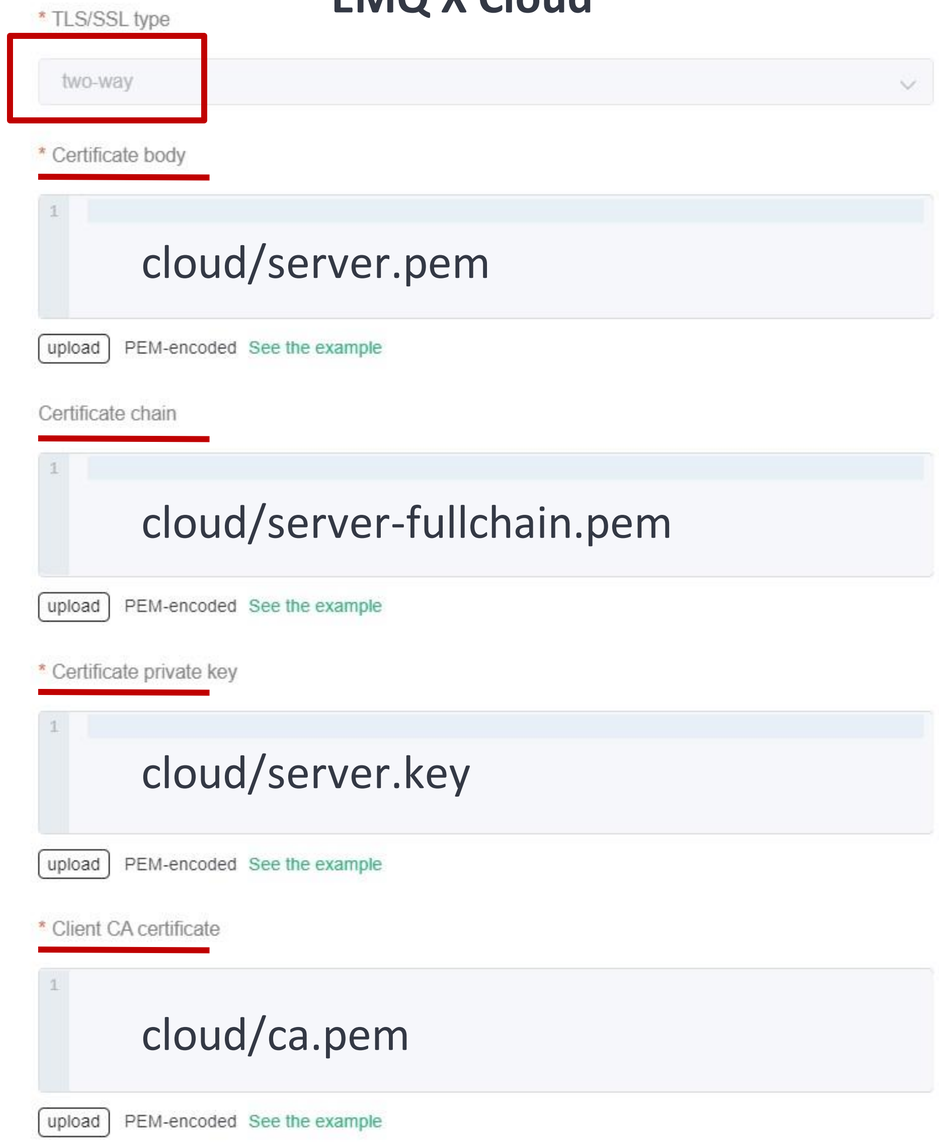
This forces two-way TLS

listeners.conf

```
listener.ssl.exernal.cacertfile = /opt/emqx/etc/certs2/local/ca.pem
listener.ssl.external.verify = verify_peer
listener.ssl.external.fail_if_no_peer_cert = true

listener.wss.exernal.cacertfile = /opt/emqx/etc/certs2/local/ca.pem
listener.wss.external.verify = verify_peer
listener.wss.external.fail_if_no_peer_cert = true
```

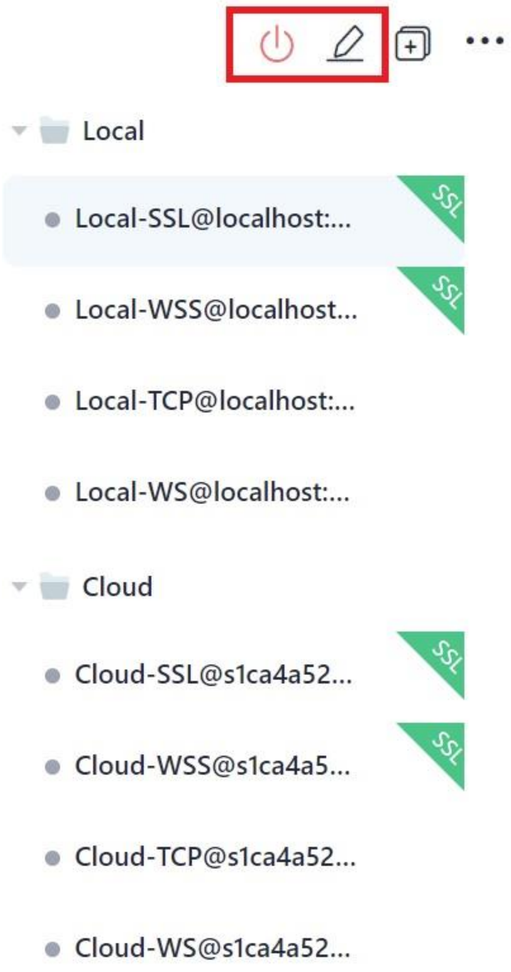
EMQ X Cloud



Client root CA file
needed for two-way TLS

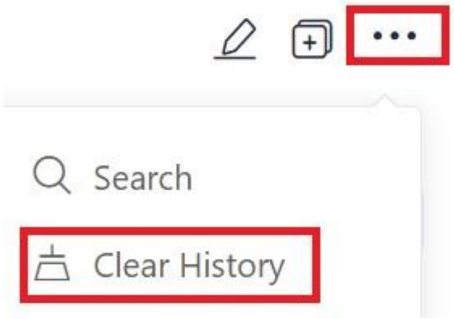
Step 8 - Test two-way TLS with MQTTX

Disconnect all the clients

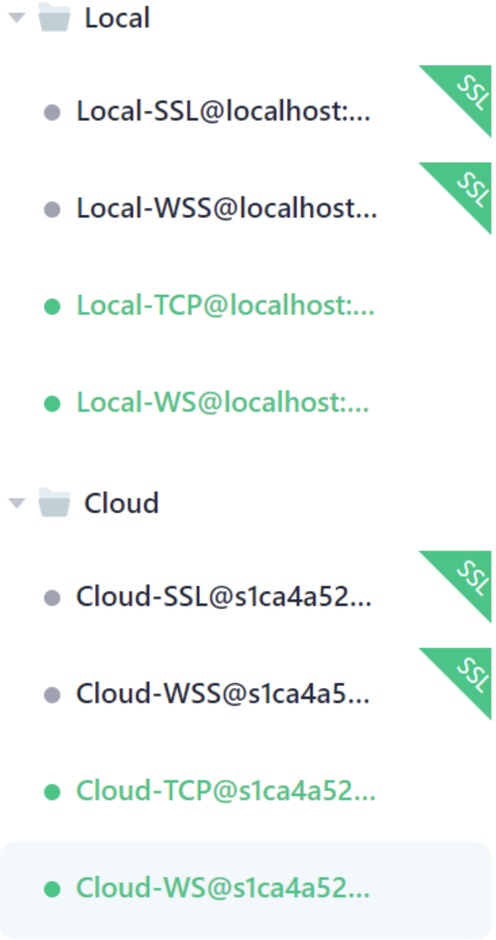


Disconnect and edit buttons

Can use Clear History to clear old messages



The TLS clients cannot connect

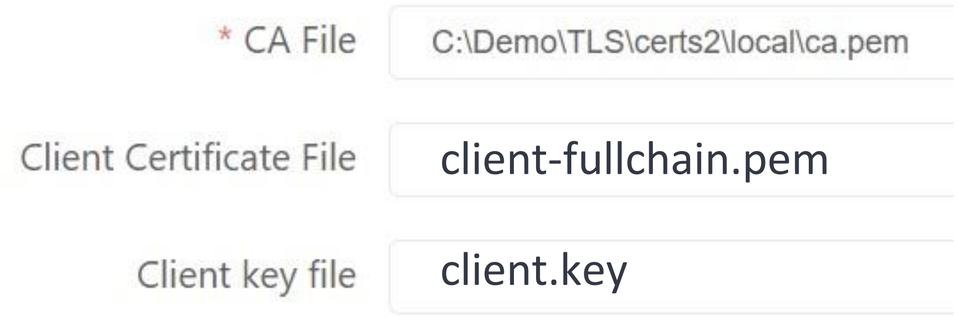


Need to add client certificate files

For two-way TLS

Need to add client-full-chain.pem and client.key files

for the SSL clients

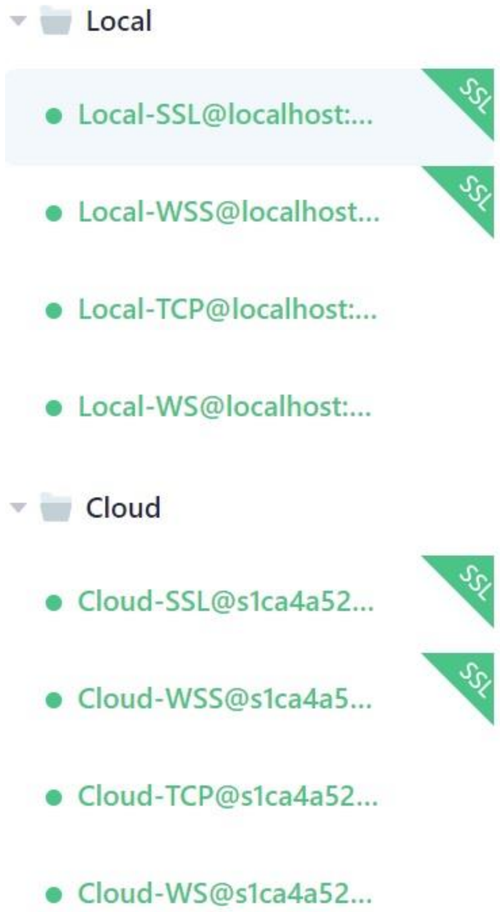


Make sure to use the correct **local** or **cloud** directory

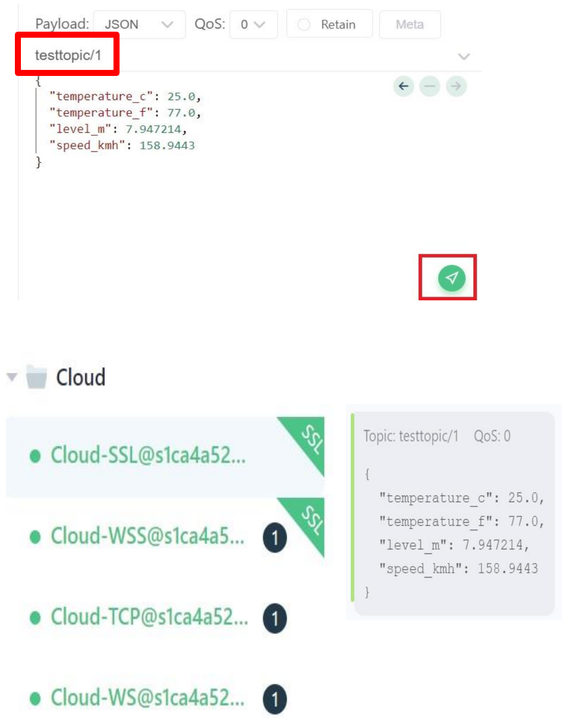
Connect

Click Connect to reconnect the clients.
Note: MQTTX will automatically subscribe clients to their subscribed topics

All clients should now be connected and listed in green.



Publish message as before and verify that the message has been received correctly.

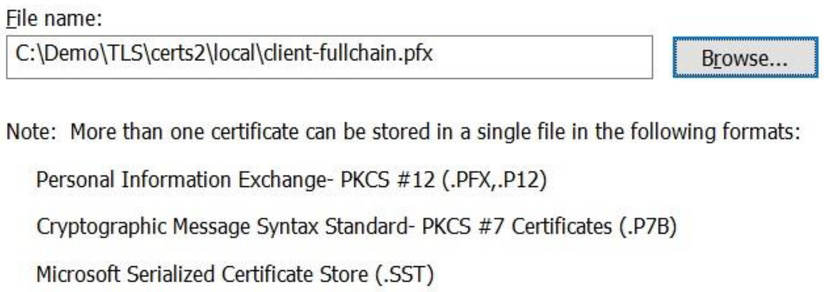


Test two-way HTTPS with browser

For Windows

Need to add client-fullchain certificate

But need to convert it to pxf format since it contains multiple certificates



Script **convertClientToPxf.bat**

```
openssl pkcs12 -export -keypbe NONE -certpbe NONE -in client-fullchain.pem -inkey client.key -out client-fullchain.pfx
```

Follow the previous steps to add the client-fullchain.**pxf** file to the **Personal** certificates

Browser will ask you to choose the client certificate