MQTT Broker Service for Tridium Niagara 4

Technical Guide

Date	07/07/2025
Revision	1.2

INTRODUCTION

The MQTT Broker Service provides the ability to make any Niagara based controller behave as a local MQTT broker:

- Designed to solve the problem where you need to integrate MQTT but have no local MQTT broker available.
- Intended for use in JACE8000 / JACE9000 or any Portability Niagara Controller.
- Designed with scalability in mind. Suitable for smaller scale integrations (Controllers powered by Niagara 4) up to multi-site deployments (Web Supervisor)
- Allows the MQTT Clients to publish and / or subscribe to the MQTT Broker Service.
- Can be used with any MQTT Client:
  - Tridium Niagara Abstract MQTT Driver
    NOTE: This may require JSON Tool Kit license (linked to an active SMA)
  - Tyrrell MQTT Service
    Perpetual licensing - no SMA termination.
  - External MQTT Clients
    This is would be any IoT enabled device(s) you wish to Subscribe from and / or Publish to.
  - MQTT Explorer
    For testing and diagnosis.

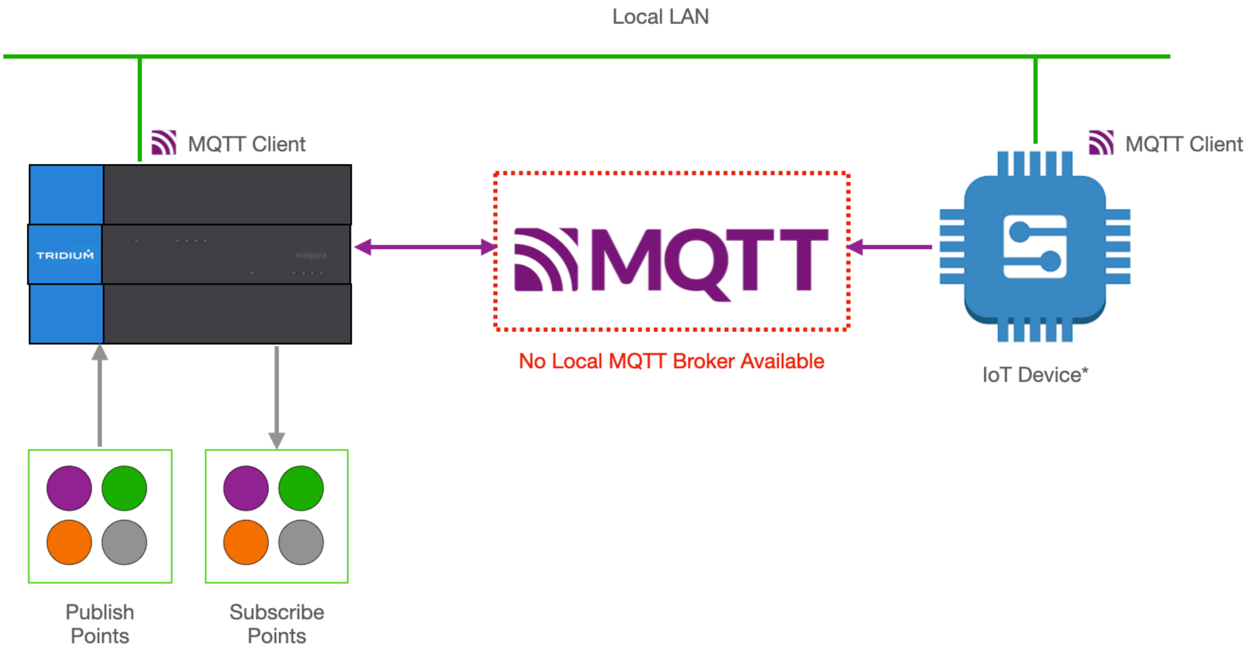
The Niagara MQTT Broker has some strict limitations that you should be familiar with before starting:

- The MQTT Broker Service with lots of connections and lots of activity may impact the performance of the hardware you are utilising.
- An already very busy JACE8000 will have a very limited performance ceiling. Add the Niagara MQTT Broker Service on top could have a performance impact. This will depend on the volume and frequency of traffic going through the Broker.
- You need to fully understand the number of topics and payloads you intend to publish to the Broker before starting.
- Refer to the licensing section to understand any further limitations.
- Retained Messages (Topics)

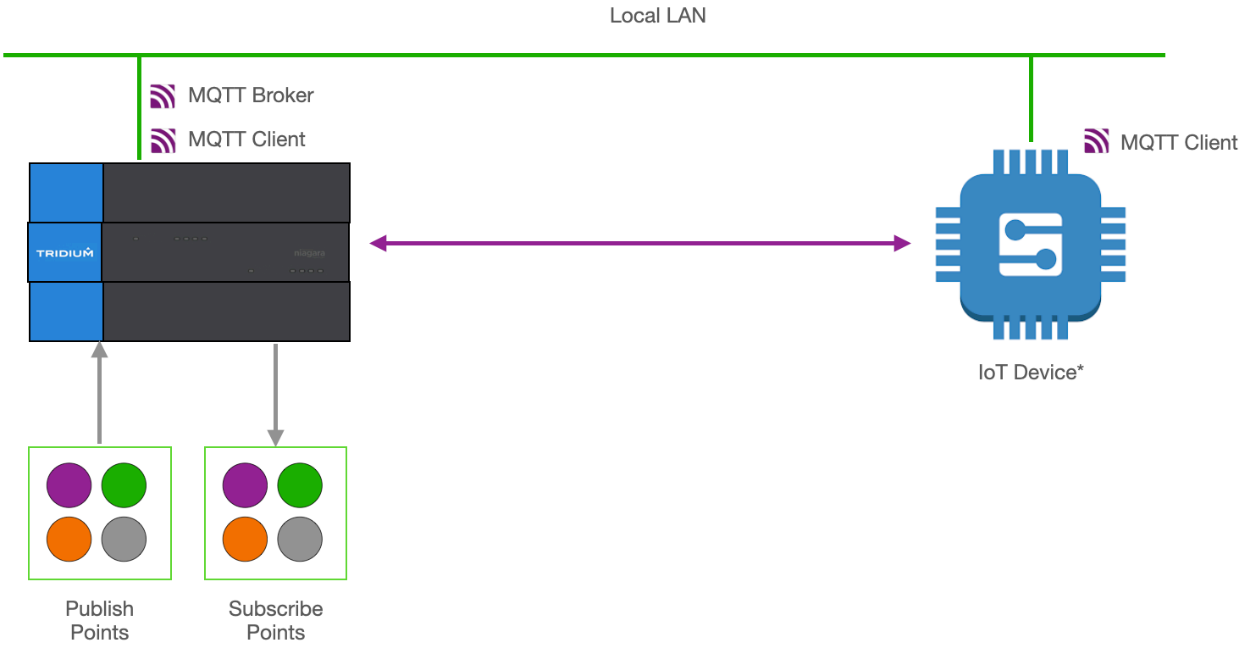
Although not a license 'feature' the service does have a maximum limit of 50 retained messages (Topics)

Example Problem Scenario:

Niagara enabled controller (MQTT Client), IoT enabled device (MQTT Client), no local MQTT broker to connect the two together.



Example Solution Using Niagara MQTT Broker Service:
Niagara enabled controller (MQTT Broker AND MQTT Client), IoT enabled device (MQTT Client), the two can be directly connected to each other.



LICENSING & SOFTWARE MAINTENANCE

The MQTT Broker is licensed based on:

- MQTT Client Connections (2x Client Connections in all base packs)
- Maximum of 5x MQTT Client Connections per Service

You will need to provide your Niagara 4 Host ID as part of your purchase. If you are expanding your system in the future you will need to ensure that your MQTT Service has been expanded to cover the number of new points being added.

Once the license has been generated you can re-import your niagara license files from the Platform > License Manager providing you have an internet connection, alternatively you can be emailed a copy of the new license files.

The MQTT Broker includes a software maintenance feature. Every new purchase of the driver will support the current release of Niagara 4 and the next release of Niagara 4, any subsequent upgrades will require a software maintenance license to be purchased.

As an example the current release of Niagara 4 is N4.15, a new driver purchase will cover you for N4.15 and a future upgrade to N4.16. Any further upgrades, for example to N4.17 or above, will require a software maintenance license to be updated. You can upgrade from any previous release with a single software maintenance purchase.

Any future upgrade to Niagara 5 will require a new software maintenance pack to be purchased, regardless of when the Niagara 4 license was purchased.

Ensure the target Host License Manager is up to date with a Tyrrell.license and Tyrrell.certifcate containing the required license features.

Any questions or queries in relation to this item should be sent to sales@tyrrellproducts.com

MQTT BROKER LICENSE PACKS

Product Code	Description
MQTT-Broker	MQTT Broker Service 2x Client Connections Maximum of 10x Client Connections
MQTT-Broker Connection	Add 1x Client Connections

DRIVER INSTALLATION

The MQTT Broker supports Niagara 4.10 and above.

NOTE:
If your installation is running an an older version of the Niagara software then it must be upgraded to meet the above requirements to run this service.

Any future updates to the MQTT Broker Service will be available for the long term maintenance (LTS) release of Niagara 4 and above. All other releases will become legacy and unsupported.

Niagara 4 Installation:
You will need the version specific JAR files for your Niagara 4 installation.

These can be downloaded from the Driver Hub: https://driverhub.tyrrellproducts.com

To install the Service copy the below JARS to c:\niagara\niagara 4.x.xx\modules

- mqttBroker-rt.jar
- mqttBroker-wb.jar

Once the files have been put into the correct directory close your workbench, and relaunch.

Additionally the **Tyrrell Code Signing Certificate** will need to be imported to:

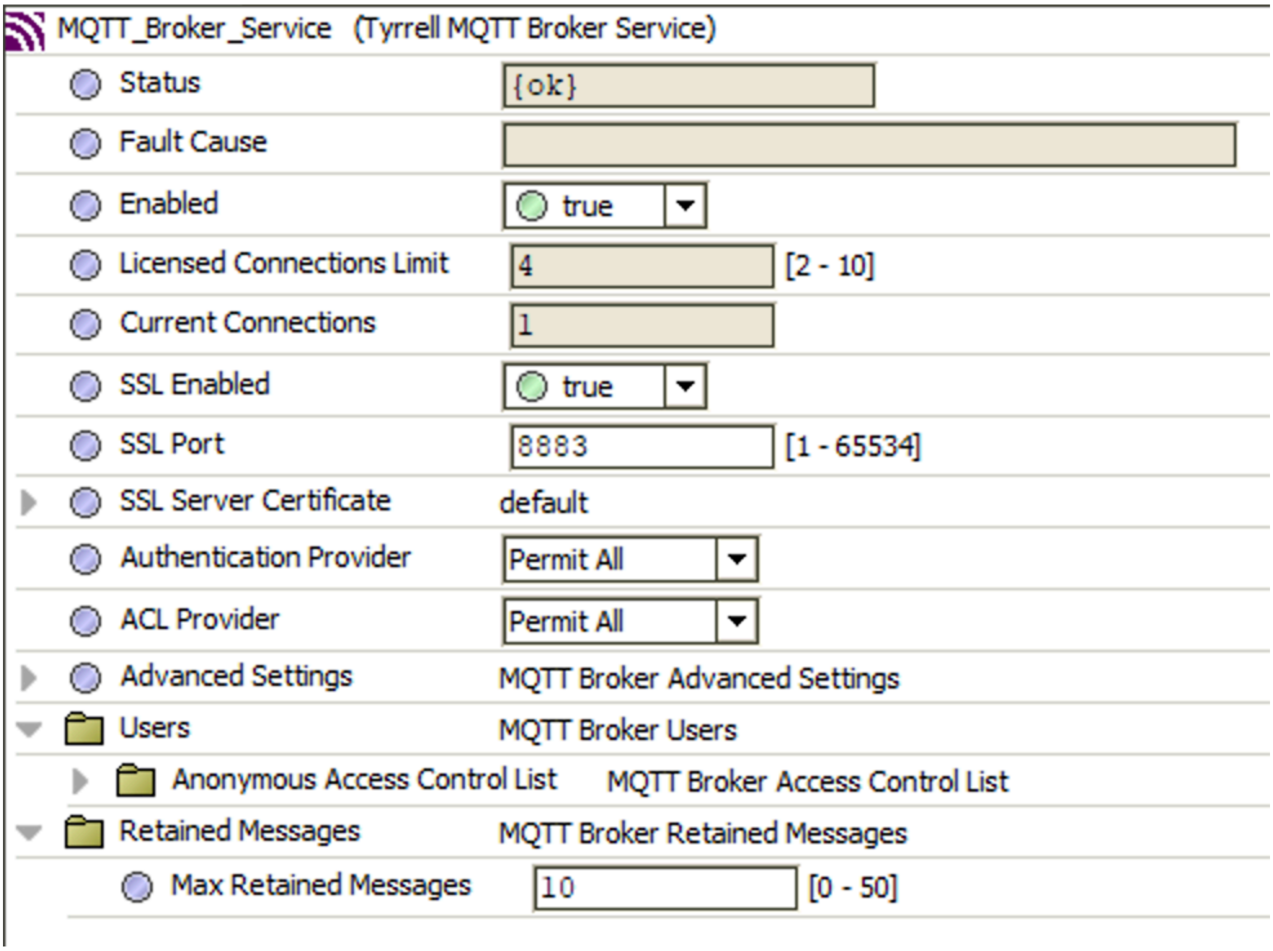
- Workbench > Tools > Certificate Management > User Trust Store
- Platform > Certificate Management > User Trust Store

Refer to the specific User Guide on installing the **Code Signing Certificate.**

The MQTT Broker is now ready to commission / update a Niagara 4 based controller. To install the driver on a JACE use the Commissioning Wizard on the platform of the target device.

SERVICE CONFIGURATION

Connect to the Niagara station where you intend to configure the MQTT Broker.
Expand **Config > Services** container and add the **MQTT Broker Service**, this can be dragged in from the **mqttBroker Palette.**
Navigate to the AX Property Sheet view of the MQTT Broker.



MQTT_Broker_Service (Tyrrell MQTT Broker Service)

Status	{ok}	
Fault Cause		
Enabled	true	
Licensed Connections Limit	4	[2 - 10]
Current Connections	1	
SSL Enabled	true	
SSL Port	8883	[1 - 65534]
SSL Server Certificate	default	
Authentication Provider	Permit All	
ACL Provider	Permit All	
Advanced Settings	MQTT Broker Advanced Settings	
Users	MQTT Broker Users	
Anonymous Access Control List	MQTT Broker Access Control List	
Retained Messages	MQTT Broker Retained Messages	
Max Retained Messages	10	[0 - 50]

The MQTT Broker provides several configuration options all based around security. All the settings are explained in detail on the following pages.
By default the MQTT Broker Service will be configured with:

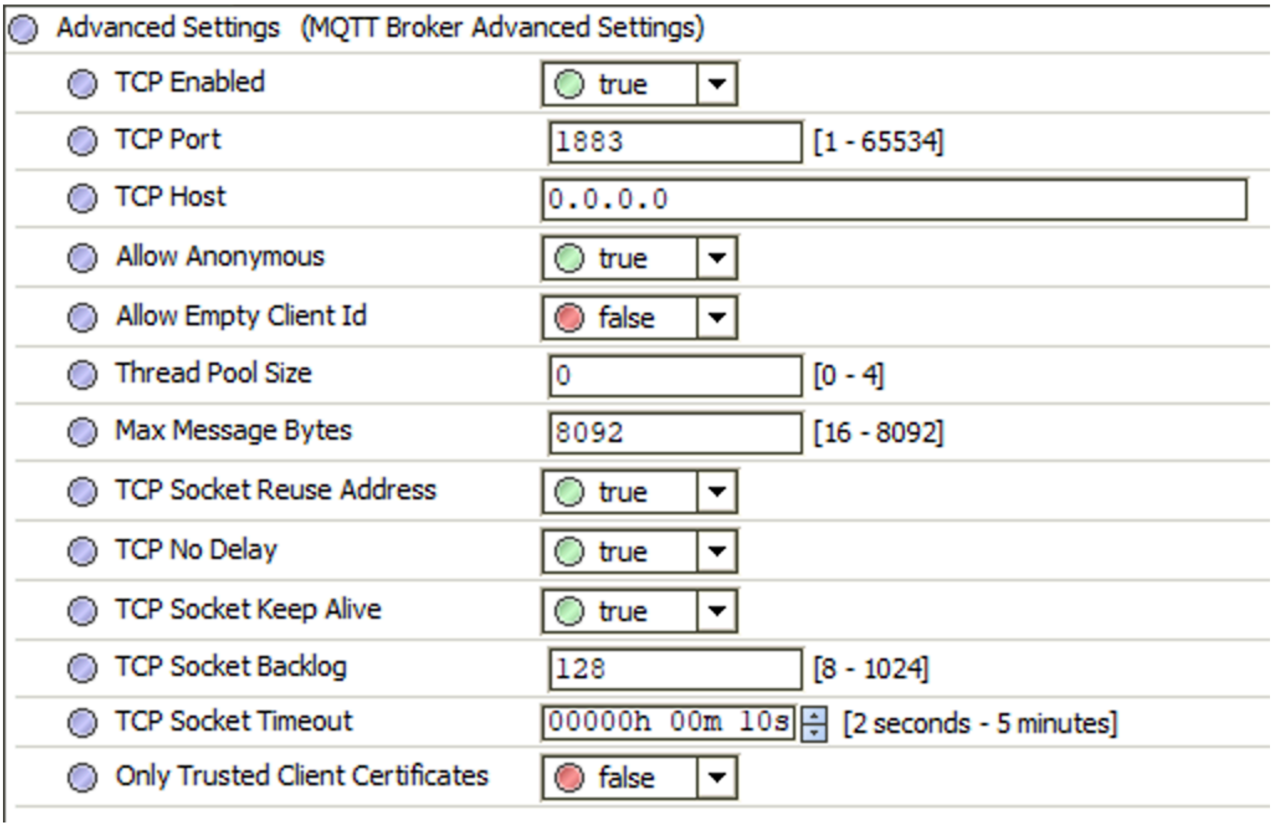
- TCP Disabled (Non Secure 1883)
- SSL Enabled (Secure 8883)
- Requires certificate selection for secure connection to be completed
- Anonymous connections not permitted by default.
  While an anonymous connection is possible its not a good practice implementation. At least one user account should be configured for a secure connection.

MQTT BROKER SETTINGS

Setting	Description
Status	MQTT Broker Status
Fault Cause	Description of the current Fault Cause
Enabled	True / False Enable / disable the entire MQTT Broker Service
Licensed Connections	Base Packs will include 2x Connections. Maximum of 10x Client Connections.
Current Connections	Number of current connections
SSL Enabled	True / False Enable / disable the secure connection to the Broker Default is True
SSL Port	8883 (default secure port for MQTT)
SSL Server Certificate	SSL Certificate Selection (from Niagara User Key Store) Select the Certificate to use.
Authentication Provider	Permit All / Deny All / Broker Users Broker Users have to be configured (Default Permit All)
ACL Provider	Access Control List Provider - used to enforce Publish / Subscribe permissions to specific Topics. Permit All / Deny All / Broker Users (Default Permit All)
Advanced Settings	See Advanced Settings Section
Users	User Account Configuration
Retained Messages	Shows all the retained messages received. Has a limit between 0 - 50 (Default is 10)

ADVANCED SETTINGS

The following advanced settings are available:



Advanced Settings (MQTT Broker Advanced Settings)

TCP Enabled	true	
TCP Port	1883	[1 - 65534]
TCP Host	0.0.0.0	
Allow Anonymous	true	
Allow Empty Client Id	false	
Thread Pool Size	0	[0 - 4]
Max Message Bytes	8092	[16 - 8092]
TCP Socket Reuse Address	true	
TCP No Delay	true	
TCP Socket Keep Alive	true	
TCP Socket Backlog	128	[8 - 1024]
TCP Socket Timeout	00000h 00m 10s	[2 seconds - 5 minutes]
Only Trusted Client Certificates	false	

Setting	Description
TCP Enabled	True / False Enables non-secure connections Default is False
TCP Port	Default MQTT Port 1883
Allow Anonymous	Allow Anonymous Connections Default is False
Allow Empty Client ID	Allows Clients without an ID to connect Default is False

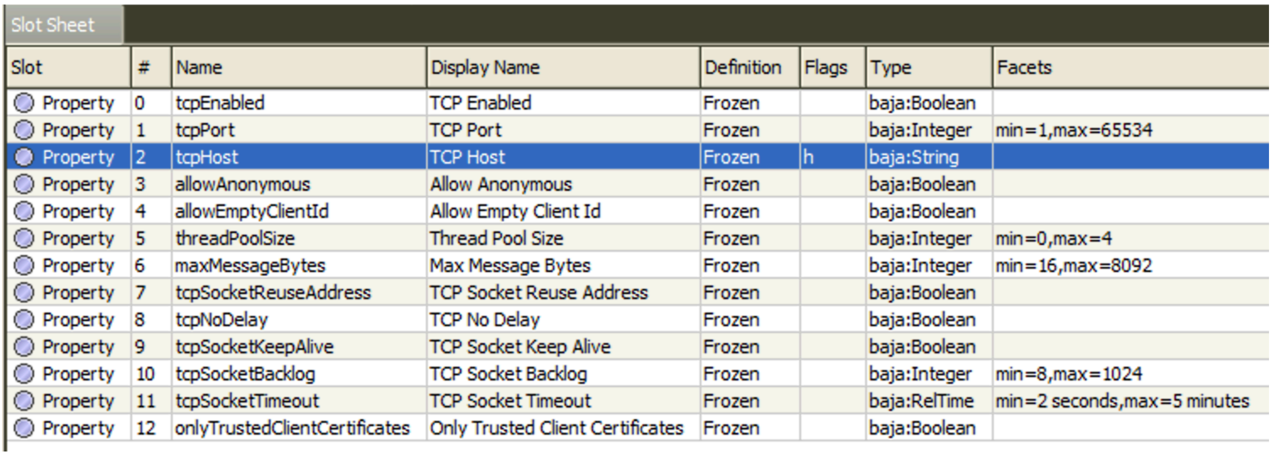
All remaining settings are not intended for general use and should not be adjusted unless explicitly instructed to do so.

NON SECURE MQTT CONNECTIONS

The MQTT Broker Service will support a non-secure MQTT configuration. However this type of setup is not recommended. Site installations should enforce a minimum security level to avoid any potential future security issues.

Non-secure connections have to be manually enabled. Enabling non-secure connections means that you fully understand the security implications of using such a connection.

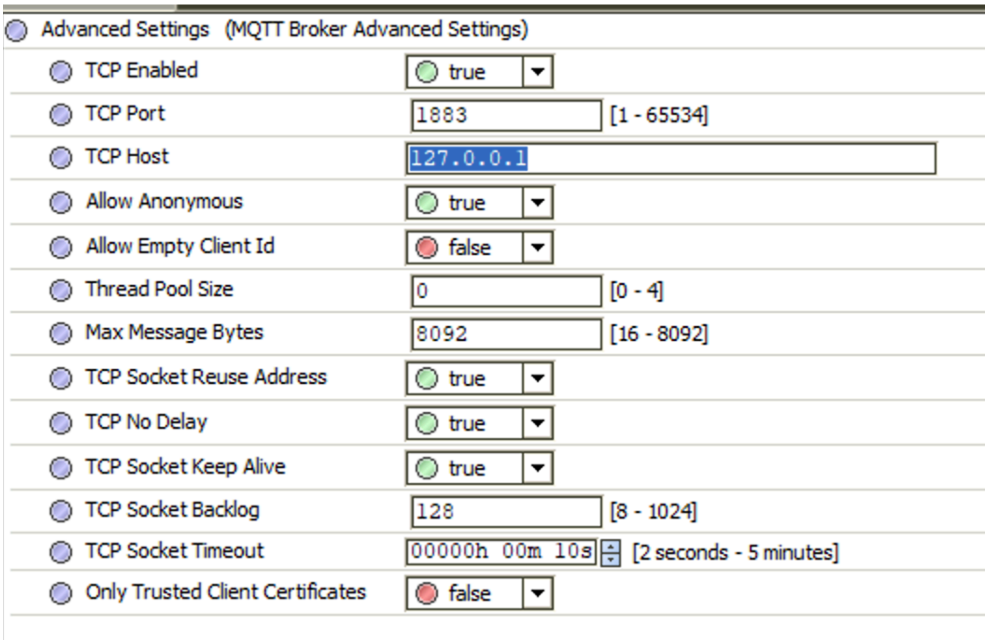
To enable non secure connections navigate to the **AX Slot Sheet** View of the Advanced settings.



Slot Sheet

Slot	#	Name	Display Name	Definition	Flags	Type	Facets
Property	0	tcpEnabled	TCP Enabled	Frozen		baja:Boolean	
Property	1	tcpPort	TCP Port	Frozen		baja:Integer	min=1,max=65534
Property	2	tcpHost	TCP Host	Frozen	h	baja:String	
Property	3	allowAnonymous	Allow Anonymous	Frozen		baja:Boolean	
Property	4	allowEmptyClientId	Allow Empty Client Id	Frozen		baja:Boolean	
Property	5	threadPoolSize	Thread Pool Size	Frozen		baja:Integer	min=0,max=4
Property	6	maxMessageBytes	Max Message Bytes	Frozen		baja:Integer	min=16,max=8092
Property	7	tcpSocketReuseAddress	TCP Socket Reuse Address	Frozen		baja:Boolean	
Property	8	tcpNoDelay	TCP No Delay	Frozen		baja:Boolean	
Property	9	tcpSocketKeepAlive	TCP Socket Keep Alive	Frozen		baja:Boolean	
Property	10	tcpSocketBacklog	TCP Socket Backlog	Frozen		baja:Integer	min=8,max=1024
Property	11	tcpSocketTimeout	TCP Socket Timeout	Frozen		baja:RelTime	min=2 seconds,max=5 minutes
Property	12	onlyTrustedClientCertificates	Only Trusted Client Certificates	Frozen		baja:Boolean	

The TCP Host setting is **hidden** and needs to be made visible via the **Config Flags**. Once the hidden flag has been set to False return to the AX Property Sheet view.



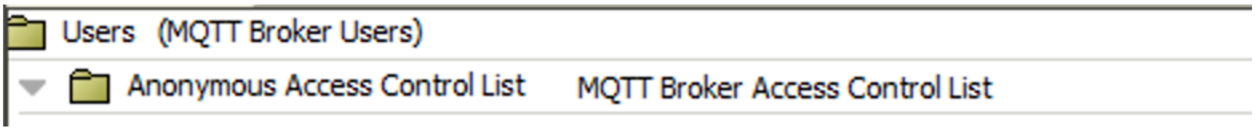
Advanced Settings (MQTT Broker Advanced Settings)

TCP Enabled	true	
TCP Port	1883	[1 - 65534]
TCP Host	127.0.0.1	
Allow Anonymous	true	
Allow Empty Client Id	false	
Thread Pool Size	0	[0 - 4]
Max Message Bytes	8092	[16 - 8092]
TCP Socket Reuse Address	true	
TCP No Delay	true	
TCP Socket Keep Alive	true	
TCP Socket Backlog	128	[8 - 1024]
TCP Socket Timeout	00000h 00m 10s	[2 seconds - 5 minutes]
Only Trusted Client Certificates	false	

The **TCP Host** setting will be set to 127.0.0.1 by default. This needs to be changed to **0.0.0.0**, allowing any remote connection.

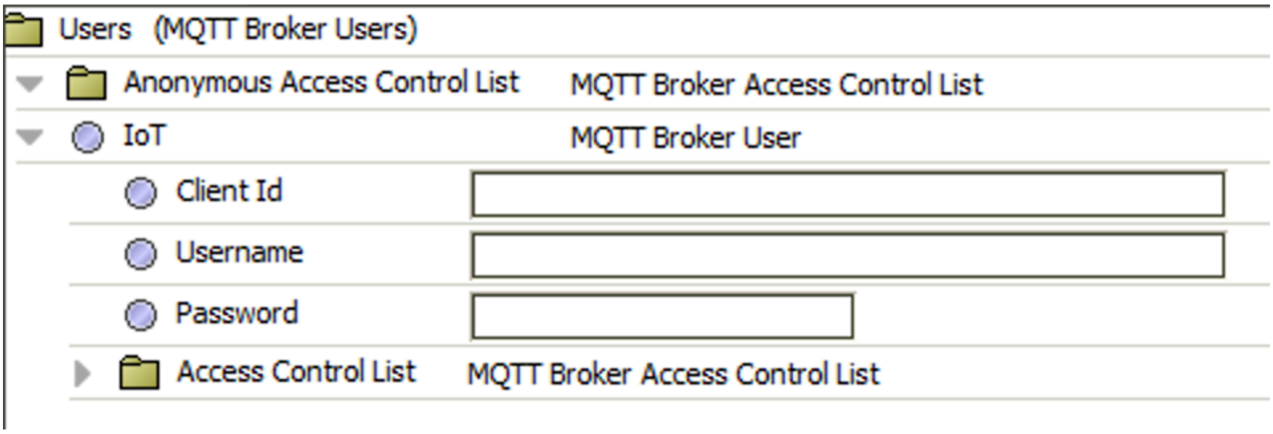
USER ACCOUNTS & ACCESS CONTROL LIST

The user account container has no configuration to start with. The use of this feature requires the settings **Authentication Provider & ACL Provider** to be set to **Broker Users**.



Users (MQTT Broker Users)
Anonymous Access Control List MQTT Broker Access Control List

Note that by default anonymous connections are **not** enabled, so at least one account should be created for a secure connection.
From the **mqttBroker palette** add a **MQTT_Broker_User** to the **Users Container**. Give the account an appropriate name.

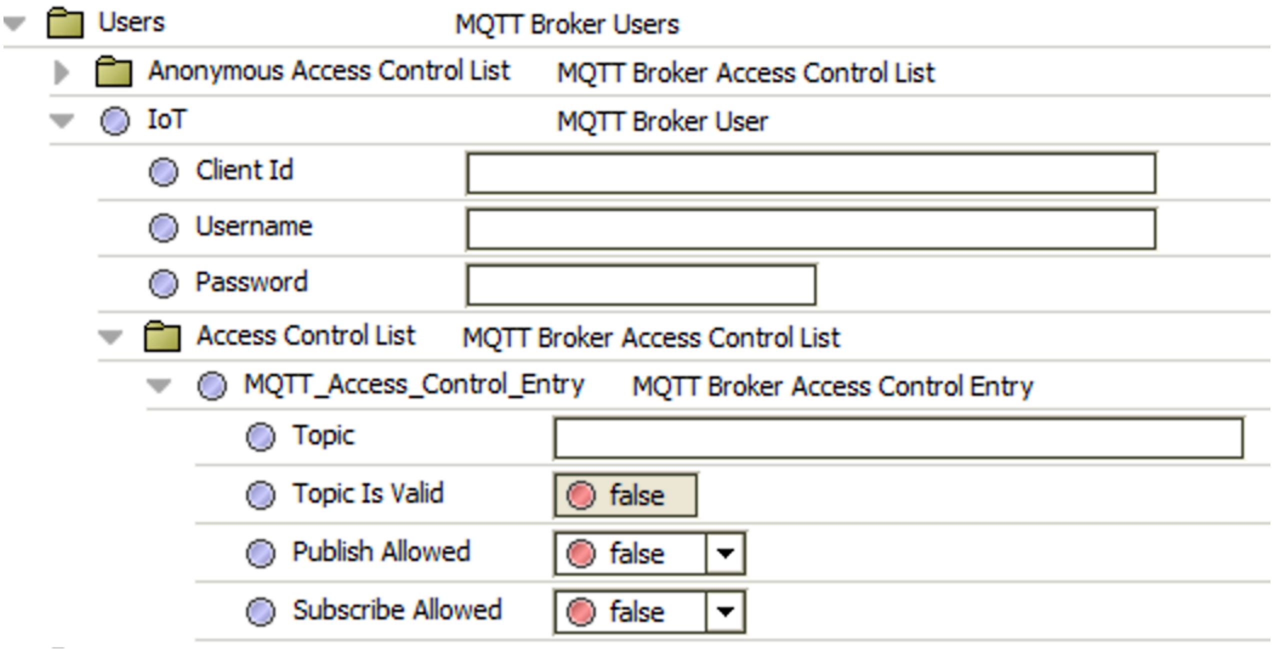


Users (MQTT Broker Users)
Anonymous Access Control List MQTT Broker Access Control List
IoT MQTT Broker User
Client Id
Username
Password
Access Control List MQTT Broker Access Control List

Setting	Description
Client ID	If left blank ignores the Client ID If specified the connecting client must have this ID
Username	Account Username
Password	Account Password
Access Control List	Allows further configuration

ACCESS CONTROL LIST

From the mqttBroker palette add a **MQTT_Access_Control_Entry** to the **Access Control List**.



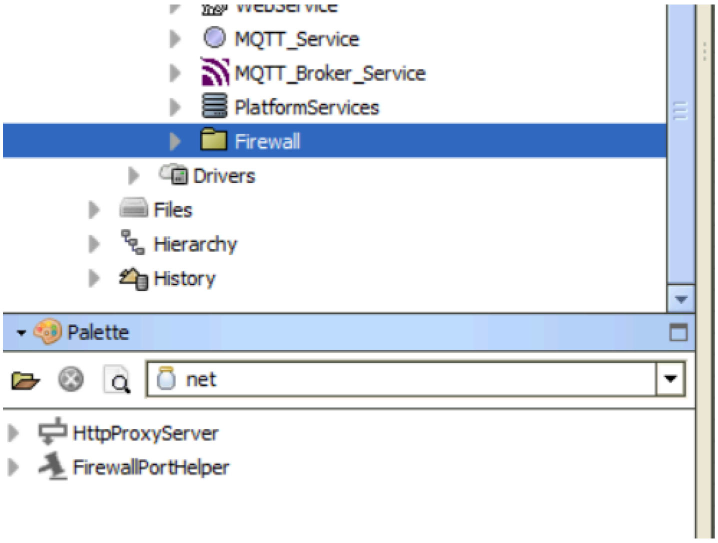
Users	MQTT Broker Users
Anonymous Access Control List	MQTT Broker Access Control List
IoT	MQTT Broker User
Client Id	
Username	
Password	
Access Control List	MQTT Broker Access Control List
MQTT_Access_Control_Entry	MQTT Broker Access Control Entry
Topic	
Topic Is Valid	false
Publish Allowed	false
Subscribe Allowed	false

Setting	Description
Topic	Topic Name To Filter
Topic Valid	Indication that Topic Name is valid A blank name is not valid
Publish Allowed	User is allowed to Publish to Topics
Subscribe Allowed	User is allowed to Subscribe to Topics

JACE9000 FIREWALL

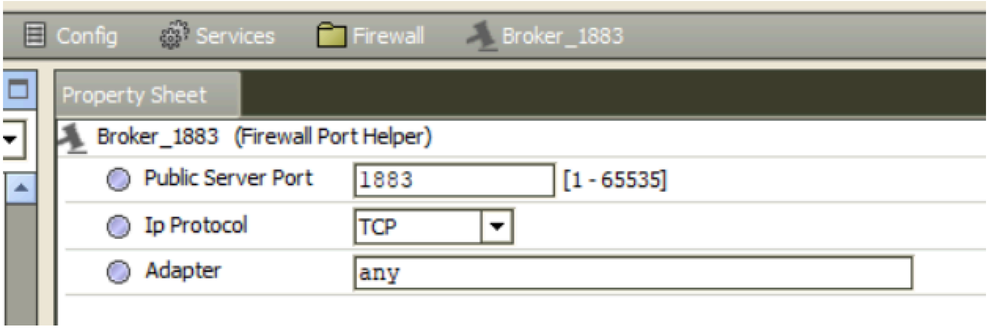
The JACE9000 has an inbuilt firewall. The firewall will prevent any connections to the MQTT Broker from working until some additional firewall rules are added.

To configure the firewall create a new Folder under the **Config > Services** container. The folder name must not contain any special characters (_ / & etc).
Open the **Net** palette



Add a **FirewallPortHelper** component to the **Folder.** You will need to add one of these components per port, to add **1883** and **8883** to the firewall 2x components will need to be added.

Navigate to the **AX Property Sheet** view of the component and configure the Port address.



Broker_1883 (Firewall Port Helper)

Public Server Port	1883	[1 - 65535]
Ip Protocol	TCP	
Adapter	any	

The port will now be added to the firewall and internal / external MQTT Clients can connect.

MQTT CLIENT - ABSTRACT MQTT DRIVER

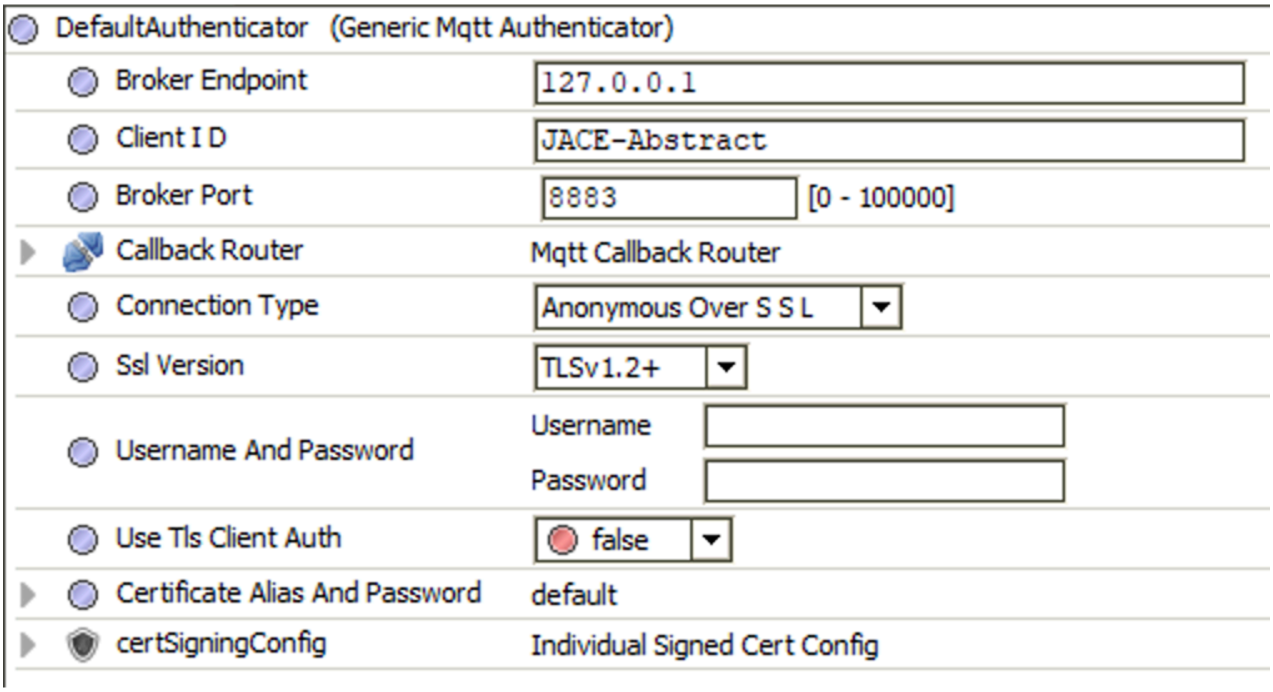
This section will explain how to make a local MQTT Client connection to the Broker within the same same Station. This will be based on the Abstract MQTT Driver. Note that if you intend to use the Abstract MQTT Driver you will most likely need license the JSON Tool Kit, this feature is intrinsically linked to an ACTIVE Software Maintenance Agreement (SMA).

Alternatively the Tyrrell MQTT Service provides automatic dissection of JSON payloads and is a perpetual license.

Ensure the Niagara based controller has the Abstract MQTT driver installed before proceeding.

Navigate to Station > Config > Drivers and add an Abstract MQTT Broker connection.

Navigate to the AX Property Sheet view of the Abstract Driver and add a Default Authenticator from the palette.



DefaultAuthenticator (Generic Mqtt Authenticator)

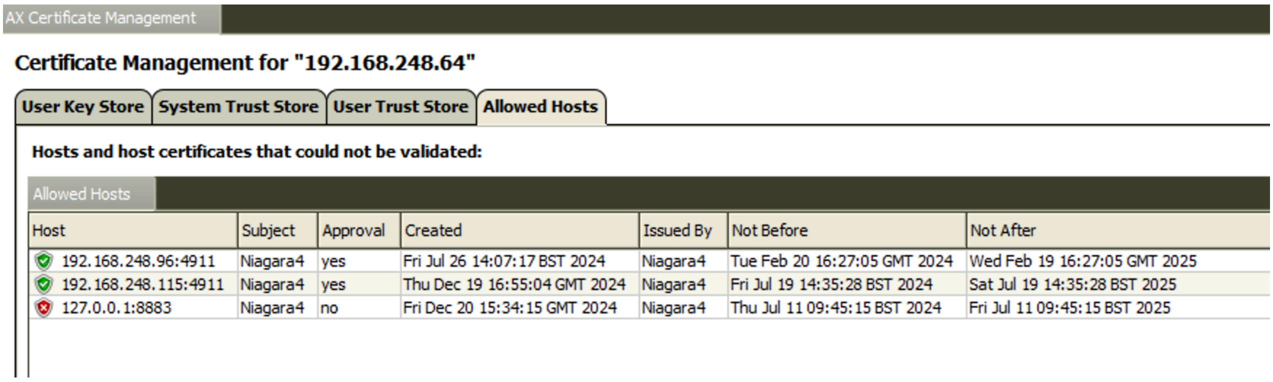
Broker Endpoint	127.0.0.1
Client I D	JACE-Abstract
Broker Port	8883 [0 - 100000]
Callback Router	Mqtt Callback Router
Connection Type	Anonymous Over S S L
Ssl Version	TLSv1.2+
Username And Password	Username Password
Use Tls Client Auth	false
Certificate Alias And Password	default
certSigningConfig	Individual Signed Cert Config

This example is:

- Anonymous Connection over SSL
- Localhost (same JACE Station as the Broker)

The Broker connection will initially fail to connect as this is a secure connection. You will need to go to the JACEs Certificate Management tool.

Navigate to Platform > Certificate Management > Allowed Hosts and approve the secure connection.

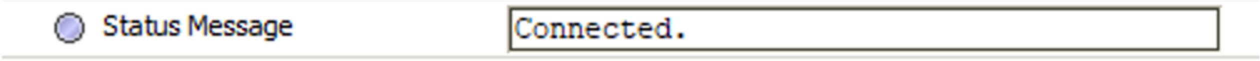


Certificate Management for "192.168.248.64"

Hosts and host certificates that could not be validated:

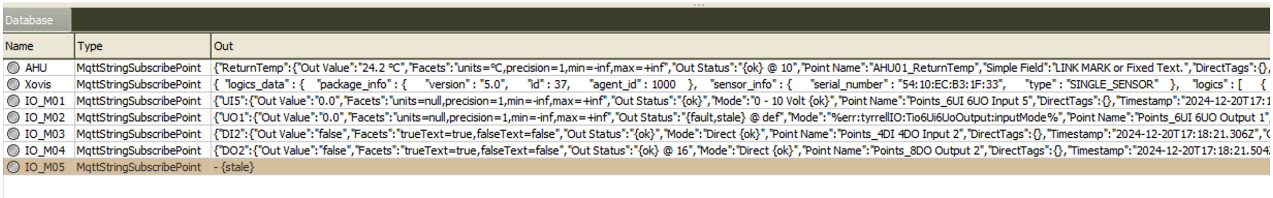
Host	Subject	Approval	Created	Issued By	Not Before	Not After
192.168.248.96:4911	Niagara4	yes	Fri Jul 26 14:07:17 BST 2024	Niagara4	Tue Feb 20 16:27:05 GMT 2024	Wed Feb 19 16:27:05 GMT 2025
192.168.248.115:4911	Niagara4	yes	Thu Dec 19 16:55:04 GMT 2024	Niagara4	Fri Jul 19 14:35:28 BST 2024	Sat Jul 19 14:35:28 BST 2025
127.0.0.1:8883	Niagara4	no	Fri Dec 20 15:34:15 GMT 2024	Niagara4	Thu Jul 11 09:45:15 BST 2024	Fri Jul 11 09:45:15 BST 2025

Once approved return to the Abstract MQTT Driver > Broker Connection and force a Connect action. The Client should then connect to the Broker as indicated by the Status Message



Status Message: Connected.

You can then add Subscribe and / or Publish points to the Abstract driver. Note that if the Subscription point is a JSON formatted payload you will need to use JSON Tool Kit to decode.



Name	Type	Out
AHU	MqttStringSubscribePoint	{"ReturnTemp":{"Out Value":"24.2 °C","Facets":"units=°C,precision=1,min=-inf,max=+inf","Out Status":"{ok} @ 10","Point Name":"AHU01_ReturnTemp","Simple Field":"LINK MARK or Fixed Text.","DirectTags":{},
Xovis	MqttStringSubscribePoint	{ "logics_data" : { "package_info" : { "version" : "5.0", "id" : 37, "agent_id" : 1000 }, "sensor_info" : { "serial_number" : "54:10:EC:B3:1F:33", "type" : "SINGLE_SENSOR" }, "logics" : [{
IO_M01	MqttStringSubscribePoint	{"UI5":{"Out Value":"0.0","Facets":"units=null,precision=1,min=-inf,max=+inf","Out Status":"{ok}","Mode":"0 - 10 Volt {ok}","Point Name":"Points_6UI 6UO Input 5","DirectTags":{},"Timestamp":"2024-12-20T17:1
IO_M02	MqttStringSubscribePoint	{"UO1":{"Out Value":"0.0","Facets":"units=null,precision=1,min=-inf,max=+inf","Out Status":"{fault,stale} @ def","Mode":"%err:tyrrellIO:Tio6Ui6UoOutput:inputMode%","Point Name":"Points_6UI 6UO Output 1",
IO_M03	MqttStringSubscribePoint	{"DI2":{"Out Value":"false","Facets":"trueText=true,falseText=false","Out Status":"{ok}","Mode":"Direct {ok}","Point Name":"Points_4DI 4DO Input 2","DirectTags":{},"Timestamp":"2024-12-20T17:18:21.306Z","
IO_M04	MqttStringSubscribePoint	{"DO2":{"Out Value":"false","Facets":"trueText=true,falseText=false","Out Status":"{ok} @ 16","Mode":"Direct {ok}","Point Name":"Points_8DO Output 2","DirectTags":{},"Timestamp":"2024-12-20T17:18:21.504
IO_M05	MqttStringSubscribePoint	- {stale}

MQTT CLIENT - TYRRELL MQTT SERVICE

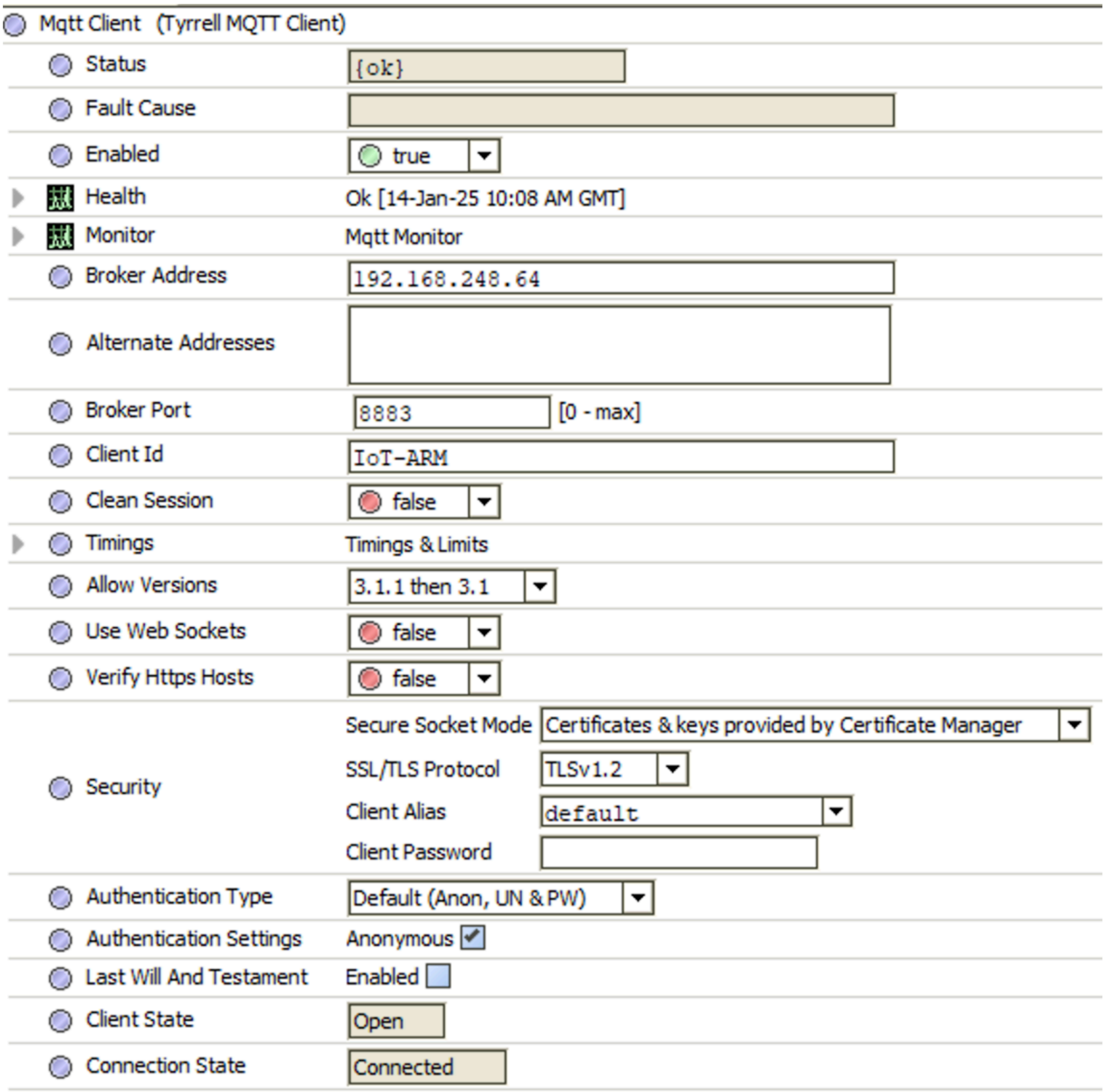
This section will explain how to make a MQTT Client connection to the Broker from a remote Station. This will be based on the Tyrrell MQTT Service.

From the MQTT Service palette, drag a new **MQTT Service** onto the **Services** container (Config

> Services).

From the Broker Connection Manager create a new connection and configure to match the settings at the MQTT Broker Service.

As an example:

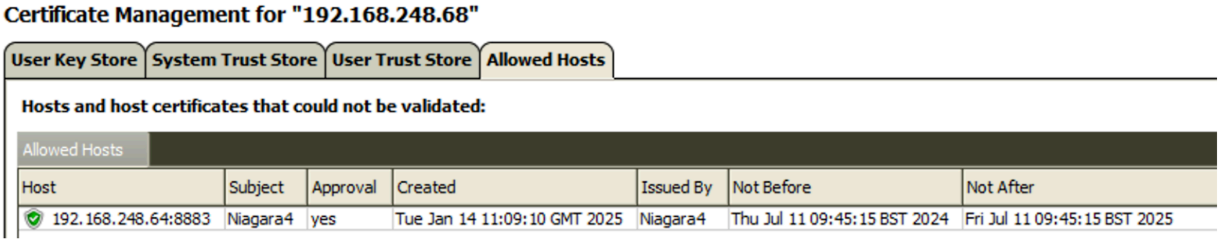


Mqtt Client (Tyrrell MQTT Client)

Status		{ok}
Fault Cause		
Enabled		true
Health		Ok [14-Jan-25 10:08 AM GMT]
Monitor		Mqtt Monitor
Broker Address		192.168.248.64
Alternate Addresses		
Broker Port		8883 [0 - max]
Client Id		IoT-ARM
Clean Session		false
Timings		Timings & Limits
Allow Versions		3.1.1 then 3.1
Use Web Sockets		false
Verify Https Hosts		false
Security	Secure Socket Mode	Certificates & keys provided by Certificate Manager
	SSL/TLS Protocol	TLSv1.2
	Client Alias	default
	Client Password	
Authentication Type		Default (Anon, UN & PW)
Authentication Settings		Anonymous ☑
Last Will And Testament		Enabled □
Client State		Open
Connection State		Connected

The Broker connection will initially fail to connect as this is a secure connection. You will need to go to the local Niagara Stations Certificate Management tool.

Navigate to **Platform > Certificate Management > Allowed Hosts** and approve the secure connection.



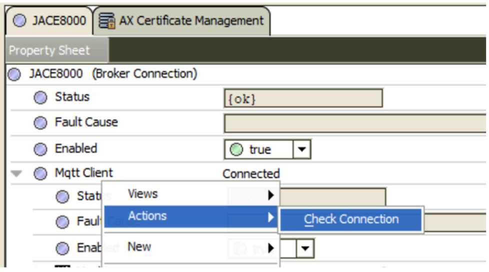
Certificate Management for "192.168.248.68"

Hosts and host certificates that could not be validated:

Allowed Hosts

Host	Subject	Approval	Created	Issued By	Not Before	Not After
192.168.248.64:8883	Niagara4	yes	Tue Jan 14 11:09:10 GMT 2025	Niagara4	Thu Jul 11 09:45:15 BST 2024	Fri Jul 11 09:45:15 BST 2025

Then confirm the connection is made. You may need to right click on the **MQTT Client > Actions > Check Connection**



JACE8000 (Broker Connection)

Status {ok}

Fault Cause

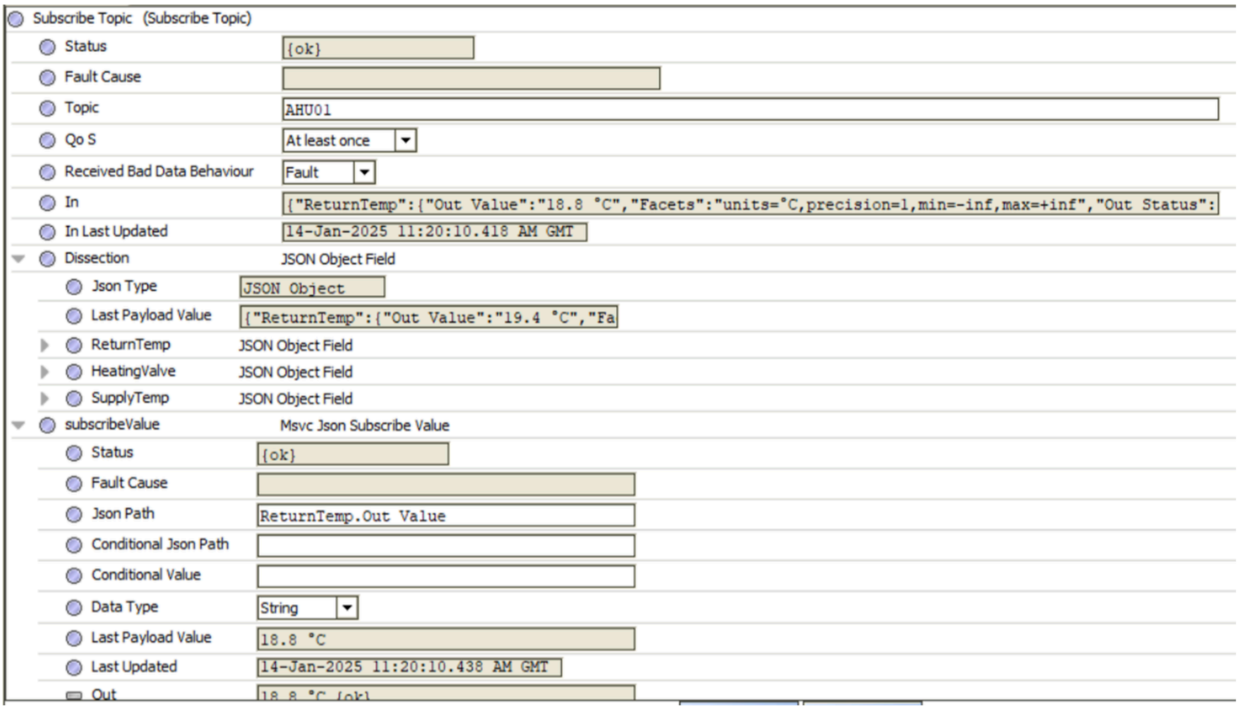
Enabled true

Mqtt Client Connected

Once connected the Status and MQTT Client > Status will come OK.

You can then configure the Service to Publish / Subscribe as required. Refer to the MQTT Service documentation for more information.

Example MQTT Service subscription with automatic JSON de-construct:



Subscribe Topic (Subscribe Topic)	
Status	{ok}
Fault Cause	
Topic	AHU01
Qo S	At least once
Received Bad Data Behaviour	Fault
In	{"ReturnTemp":{"Out Value":"18.8 °C","Facets":"units=°C,precision=1,min=-inf,max=+inf","Out Status":
In Last Updated	14-Jan-2025 11:20:10.418 AM GMT
Dissection	JSON Object Field
Json Type	JSON Object
Last Payload Value	{"ReturnTemp":{"Out Value":"19.4 °C","Fa
ReturnTemp	JSON Object Field
HeatingValve	JSON Object Field
SupplyTemp	JSON Object Field
subscribeValue	Msvc Json Subscribe Value
Status	{ok}
Fault Cause	
Json Path	ReturnTemp.Out Value
Conditional Json Path	
Conditional Value	
Data Type	String
Last Payload Value	18.8 °C
Last Updated	14-Jan-2025 11:20:10.438 AM GMT
Out	18.8 °C {ok}

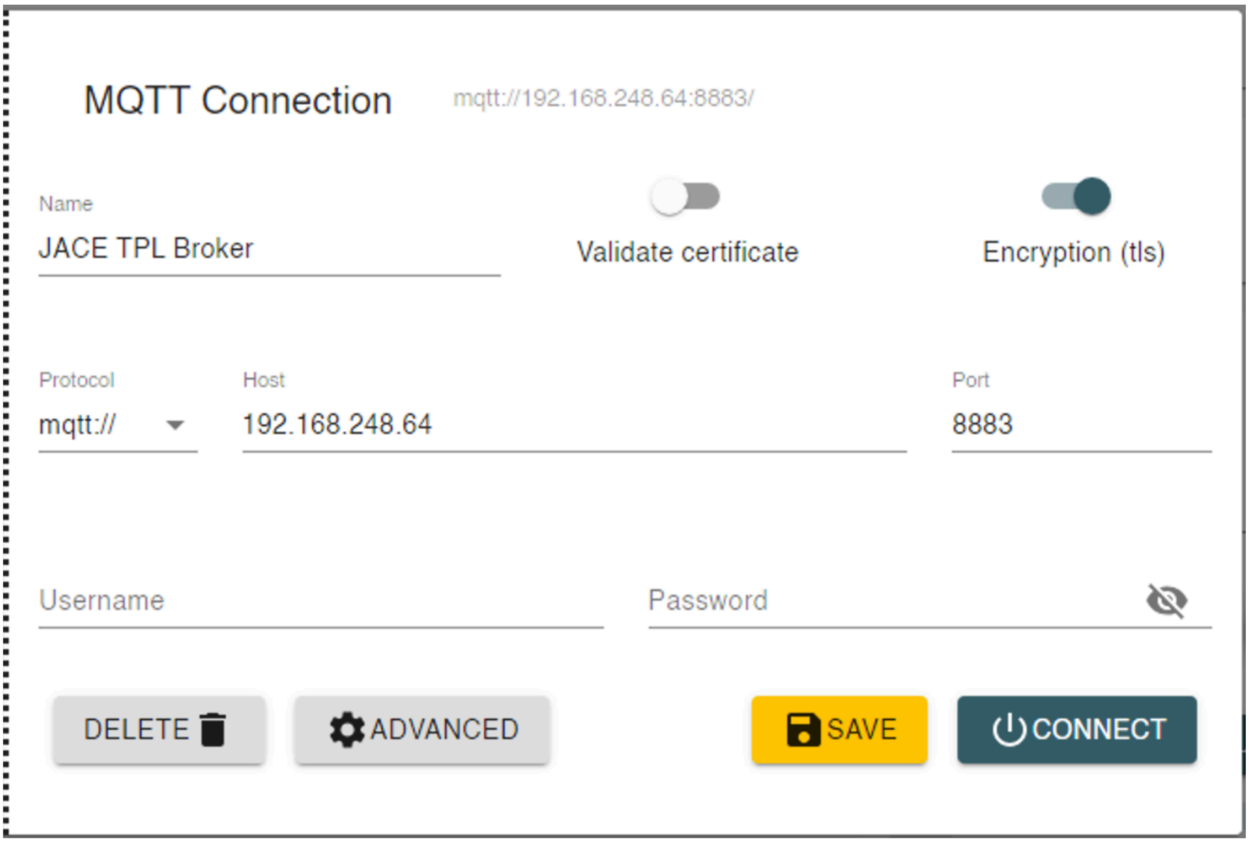
MQTT CLIENT - MQTT EXPLORER

MQTT Explorer is tool that will allow you to connect to the Broker and review the payloads that are present (retained) or arriving.

MQTT Client will require 1x free MQTT Client license at the Broker end. IF there are no free connections available either temporarily disconnect an existing MQTT Client or license an additional connection.

This is an example of the Broker connection settings

- JACE IP Address
- Broker Port 8883 (matching the configuration of the Broker Service)
- Secure TLS Connection (Validate Certificate is DISABLED)
- No Username / Password required in this example



MQTT Connection mqtt://192.168.248.64:8883/

Name: JACE TPL Broker

Validate certificate: ○ (off)

Encryption (tls): ● (on)

Protocol: mqtt://

Host: 192.168.248.64

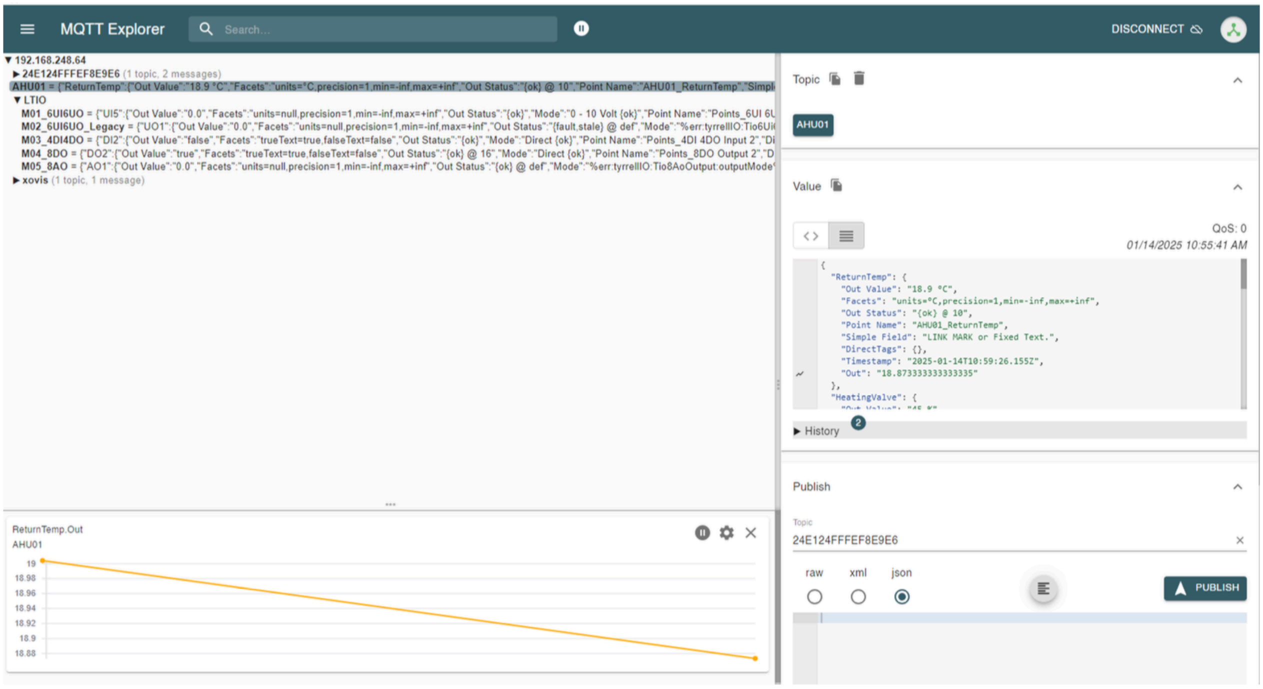
Port: 8883

Username

Password

Once configured press connect and providing all confutation is correct the MQTT Explore will connect to the Broker. This will then allow you to monitor the Topics arriving and also publish to specific Topics.

Example MQTT Broker Live Monitoring:



```
▼ 192.168.248.64
  ▶ 24E124FFFEF8E9E6 (1 topic, 2 messages)
  AHU01 = {"ReturnTemp":{"Out Value":"18.9 °C","Facets":"units=°C,precision=1,min=-inf,max=+inf","Out Status":"{ok} @ 10","Point Name":"AHU01_ReturnTemp","Simpl
  ▼ LTIO
    M01_6UI6UO = {"UI5":{"Out Value":"0.0","Facets":"units=null,precision=1,min=-inf,max=+inf","Out Status":"{ok}","Mode":"0 - 10 Volt {ok}","Point Name":"Points_6UI 6U
    M02_6UI6UO_Legacy = {"UO1":{"Out Value":"0.0","Facets":"units=null,precision=1,min=-inf,max=+inf","Out Status":"{fault,stale} @ def","Mode":"%err:tyrrellIO:Tio6Ui
    M03_4DI4DO = {"DI2":{"Out Value":"false","Facets":"trueText=true,falseText=false","Out Status":"{ok}","Mode":"Direct {ok}","Point Name":"Points_4DI 4DO Input 2","Di
    M04_8DO = {"DO2":{"Out Value":"true","Facets":"trueText=true,falseText=false","Out Status":"{ok} @ 16","Mode":"Direct {ok}","Point Name":"Points_8DO Output 2","D
    M05_8AO = {"AO1":{"Out Value":"0.0","Facets":"units=null,precision=1,min=-inf,max=+inf","Out Status":"{ok} @ def","Mode":"%err:tyrrellIO:Tio8AoOutput:outputMode
  ▶ xovis (1 topic, 1 message)
```

Topic: AHU01

QoS: 0
01/14/2025 10:55:41 AM

```
{
  "ReturnTemp": {
    "Out Value": "18.9 °C",
    "Facets": "units=°C,precision=1,min=-inf,max=+inf",
    "Out Status": "{ok} @ 10",
    "Point Name": "AHU01_ReturnTemp",
    "Simple Field": "LINK MARK or Fixed Text.",
    "DirectTags": {},
    "Timestamp": "2025-01-14T10:59:26.155Z",
    "Out": "18.873333333333335"
  },
  "HeatingValve": {
```

REVISION HISTORY

REVISION	DESCRIPTION
1.0	Draft Release For Approval
1.1	Non Secure MQTT Details Added
1.2	JACE9000 Firewall Details Added