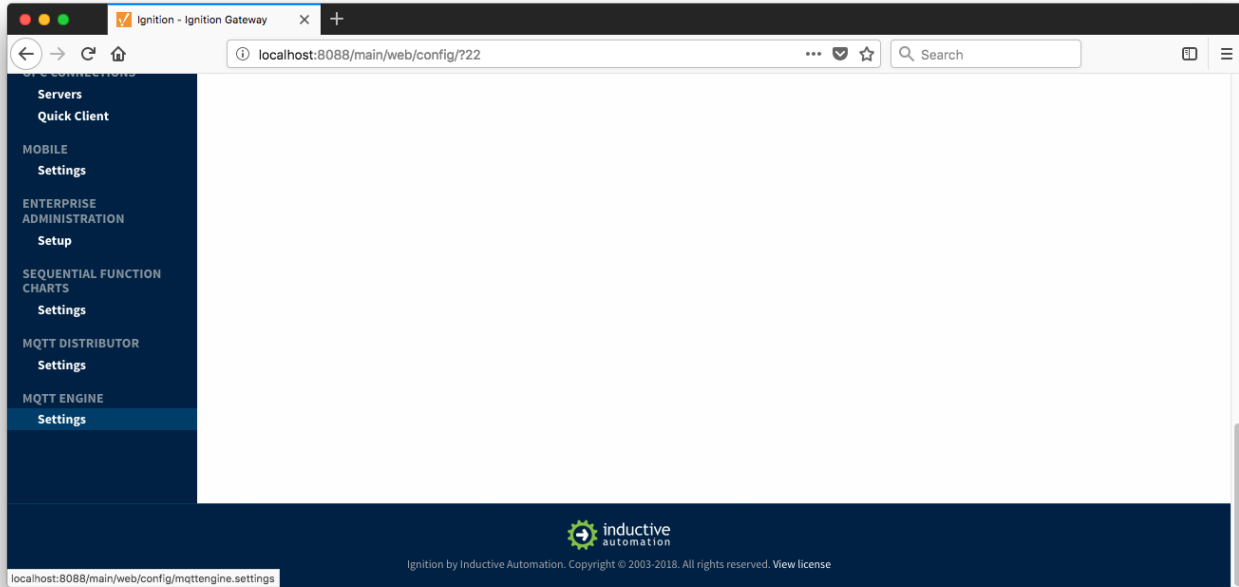


ME: Configuration

MQTT Engine provides a configuration section to the Ignition Gateway. These can be seen in the Configure section of the Ignition Gateway web UI in the left panel.



Once in the configuration section there are three tabs: Servers, Advanced, and Namespaces. Each of these tabs is described in detail in the following sections.

General

The first tab contains general settings which allows one to enable/disable the module, configure Application ID Filters, and Cirrus Link Chariot Access Settings. The Chariot access settings are used in conjunction with a Cirrus Link Chariot deployment and will be provided when purchasing Chariot Cloud services.

Main

- **Enabled**
 - Whether or not the MQTT Engine module is enabled and connecting to configured MQTT Servers.
- **Primary Host ID**
 - The primary host ID to use for 'state' checks. These checks are used to ensure that the primary backend MQTT Engine client remains connected. If the primary host ID is set, MQTT Engine will publish it's connection state on a topic that contains the Primary Host ID. In the event that MQTT Engine becomes disconnected from the server, a death certificate will be published on the same topic, setting the state to 'offline'. Any connecting client that subscribes on the state topic will then be notified of the MQTT Engine state and walk to the next server is MQTT Engine is offline.
- **Group ID Filters**
 - A comma separated list of group IDs to ignore. If no group IDs are to be ignored, this should be left empty

Chariot Access

- **Chariot Cloud Access Key**
 - If using Cirrus Link's Chariot platform, this is the provided Access Key
- **Chariot Cloud Secret Key**
 - If using Cirrus Link's Chariot platform, this is the provided Secret Key

Miscellaneous

- **Block Node Commands**
 - Whether or not to block outgoing commands from MQTT Engine to Edge Nodes. This is true by default and provides a security mechanism for preventing accidental outgoing commands from MQTT Engine.
- **Block Device Commands**
 - Whether or not to block outgoing commands from MQTT Engine to Devices. This is true by default and provides a security mechanism for preventing accidental outgoing commands from MQTT Engine.
- **File Policy**
 - The policy for handling Files contained within a Sparkplug payload.

- **Ignore:** The default policy. The file will be ignored.
 - **Store:** The file will be stored to the local file system in the directory specified by the File Location field. A Tag will be created with a String value representing the file URI.
- **File Location**
 - The directory to store the files when using the **Store** policy described above.
- **Store Historical Events**
 - Whether or not to write historical change events directly to the Historian (if history is enabled on a Tag) instead of updating the live Tag value.

The screenshot shows the Ignition Gateway web interface. The left sidebar contains a navigation menu with categories like Redundancy, Gateway Settings, NETWORKING, SECURITY, DATABASES, ALARMING, TAGS, OPC-UA SERVER, OPC CONNECTIONS, MOBILE, ENTERPRISE ADMINISTRATION, SEQUENTIAL FUNCTION CHARTS, MQTT DISTRIBUTOR, and MQTT ENGINE. The main content area is titled 'General Settings' and contains three sections: Main, Chariot Access, and Miscellaneous. The 'Main' section has fields for 'Enabled' (checked), 'Primary Host ID', and 'Group ID Filters'. The 'Chariot Access' section has fields for 'Chariot Cloud Access Key' and 'Chariot Cloud Secret Key'. The 'Miscellaneous' section has checkboxes for 'Block Node Commands', 'Block Device Commands', and 'Block Property Changes', a dropdown for 'File Policy' (set to 'Ignore'), a text field for 'File Location', and a checkbox for 'Store Historical Events' (checked). A 'Save Changes' button is at the bottom right.

Main	
Enabled	☒ Enable the MQTT Engine (default: true)
Primary Host ID	 The Primary Host ID to allow connecting clients to ensure they remain connected to this application (optional)
Group ID Filters	 A comma separated list of Group IDs to listen for (optional)

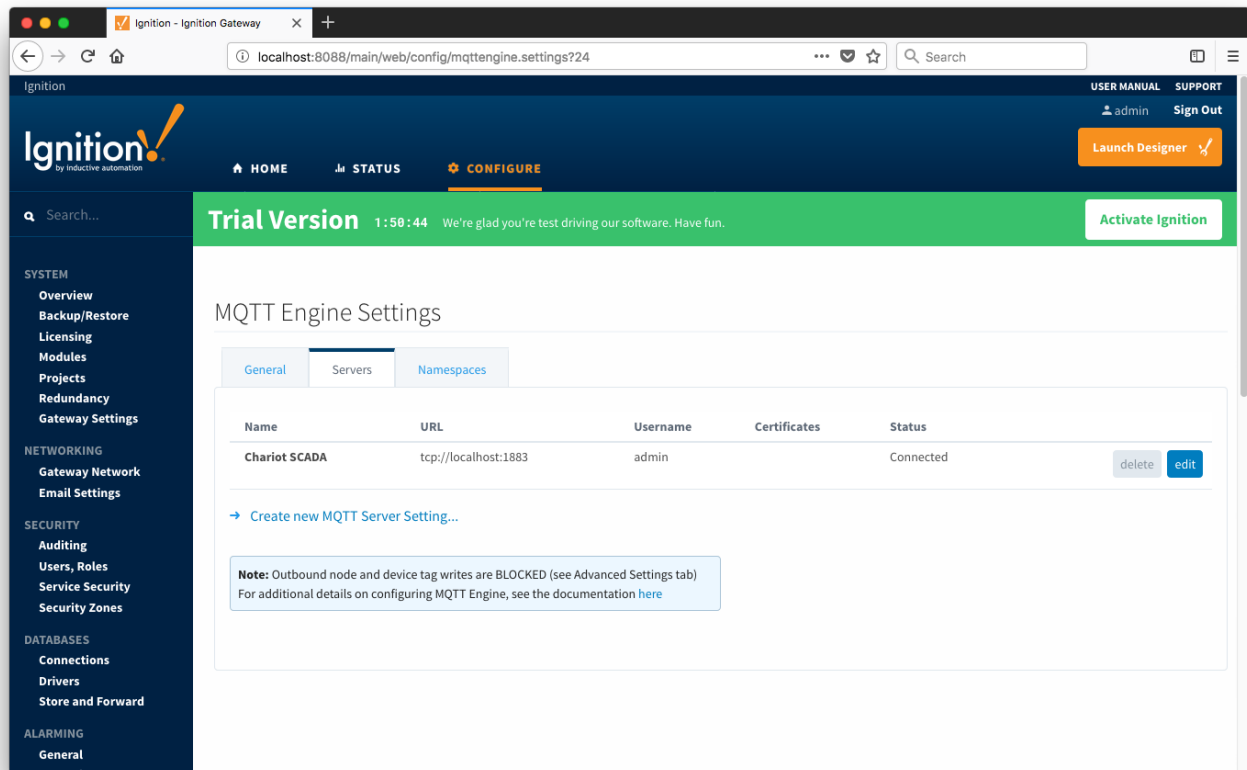
Chariot Access	
Chariot Cloud Access Key	 The optional Chariot Cloud Access Key used for Cirrus Link hosted Chariot MQTT Servers (optional)
Chariot Cloud Secret Key	 The optional Chariot Cloud Secret Key used for Cirrus Link hosted Chariot MQTT Servers (optional)

Miscellaneous	
Block Node Commands	☒ Block outbound edge node tag writes
Block Device Commands	☒ Block outbound device tag writes
Block Property Changes	☐ Block incoming Tag property changes
File Policy	Ignore The policy for handling incoming files
File Location	 The directory to store files in when using the "Store" file policy (optional)
Store Historical Events	☒ Enabled the writing of historical change events directly to the History provider instead of updating the Tag value (default: true)

[Save Changes](#)

Servers

The second tab is a list of MQTT Servers that MQTT Engine should connect to. By default, MQTT Engine is configured to connect to the local MQTT Distributor based MQTT Server. It is set up to connect to localhost, port 1883, using the default username/password pair of admin/changeme. Out of the box MQTT Engine will work with MQTT Distributor and its default configuration. The connection status of each server can be seen in the 'Status' column. Clicking on the 'Create new MQTT Server' link will bring up the following form for adding a new MQTT Server setting.



Additional or alternative MQTT Servers can be configured in MQTT Engine. Often times more than one will be configured to handle fail-over in redundant or geographically distributed systems. The configuration options for servers are listed below.

Main

- **Name**
 - This is the friendly name of the MQTT Server used to easily identify it.
- **Enabled**
 - Whether or not connections to this MQTT Server are enabled.
- **URL**
 - This is the URL of the MQTT server. Its format is as follows: [protocol]://[location]:[port]. Each of these are shown below.
 - protocol - Either tcp or ssl
 - location - The server location. e.g. localhost, [myserver.chariot.io](#), [mydomain.com](#), etc
 - port - The port the MQTT Server is listening on. Generally this is 1883 if using TCP or 8883 if using SSL
- **Server Type**
 - This is the type of MQTT Server to connect to.
 - Chariot - If connecting to a Cirrus Link Chariot on-premise or Chariot cloud based MQTT Server
 - MQTT Distributor - If connecting to a Ignition MQTT Distributor server
 - Third Party - If connecting to a third party 3.1.1 compliant MQTT Server
- **Username**
 - Optional MQTT username to use in the MQTT connect packet. This is required if the MQTT Server to connect to requires it.
- **Password**
 - Optional MQTT password to use in the MQTT connect packet. This is required if the MQTT Server to connect to requires it.

TLS Settings

- **Certificates**
 - The server certificates to use if required. These are generally only required when connecting using TLS and the MQTT server does not have a genuine certificate issued by a trusted certificate authority.
 - CA certificate must always end in 'ca.pem'
 - If using client side certificates (i.e. a public/private keypair and CA cert), make sure the following naming conventions are followed.
 - Edge/Device certificate must end in 'cert.pem'
 - Edge/Device private key must end in 'private.key'
- **Password**
 - A password associated with the certificate's private key.

Advanced Settings

- **Client ID**
 - Optional MQTT client ID to use. If specified this will be used in the MQTT Engine connect packet when connecting to the server. If left blank, a random client ID will be created of the form 'IgnitionTarget-xxxxxxx-xxxx-xxxx'.
- **Keep Alive**
 - The MQTT client keep alive time (in seconds).
- **Filtered Namespaces**
 - A comma separated list of namespaces that will be filtered/disabled for connections to this MQTT Server.

Clicking on the 'Create new MQTT Server...' link will bring up the following form to add a new Server.

The screenshot shows the 'New MQTT Server Setting' form in the Ignition Gateway web interface. The form is divided into two main sections: 'Main' and 'Advanced'.

Main Section:

- Name:** A text input field containing 'New Mqtt Server'. Below it, a description reads: 'The friendly name of this MQTT Server Setting'.
- Enabled:** A checkbox labeled 'Enable this MQTT Server Connection' which is checked.
- URL:** A text input field. Below it, a description reads: 'The MQTT Server URL. Should be of the form tcp://mydomain.com:1883 or ssl://mydomain.com:8883'.
- Username:** A text input field containing 'admin'. Below it, a description reads: 'The username for connections if required by the MQTT Server (optional)'.
- Password:** A text input field with masked characters (dots). Below it, a description reads: 'The password for connections if required by the MQTT Server (optional)'.
- Password (Verification):** A text input field for re-typing the password. Below it, a description reads: 'Re-type password for verification.'
- Certificates:** A section with a 'Browse...' button and the text 'No file selected.' Below it is a 'Files:' label and an empty file list area.

Advanced Section:

- Client ID:** A text input field. Below it, a description reads: 'The MQTT Client ID for connections to the MQTT Server. If left blank one will be auto-generated (optional)'.
- Keep Alive:** A text input field. Below it, a description reads: 'The MQTT Client keep alive time (in seconds) (default: 30)'.
- Filtered Namespaces:** A text input field. Below it, a description reads: 'Comma separated list of namespaces (e.g. B&B Wizzard, Elecsys, Xirgo) to be disabled for this MQTT Server connection (optional)'.

At the bottom right of the form is a blue button labeled 'Create New MQTT Server Setting'.

Namespaces

The third tab is used for configuring namespaces. Each namespace configuration represents a family of devices and/or data that MQTT Engine will support. A namespace defines the topics that each MQTT Engine client will subscribe on as well as indicates how the payload will be handled. There are two types of namespaces: Default and Custom.

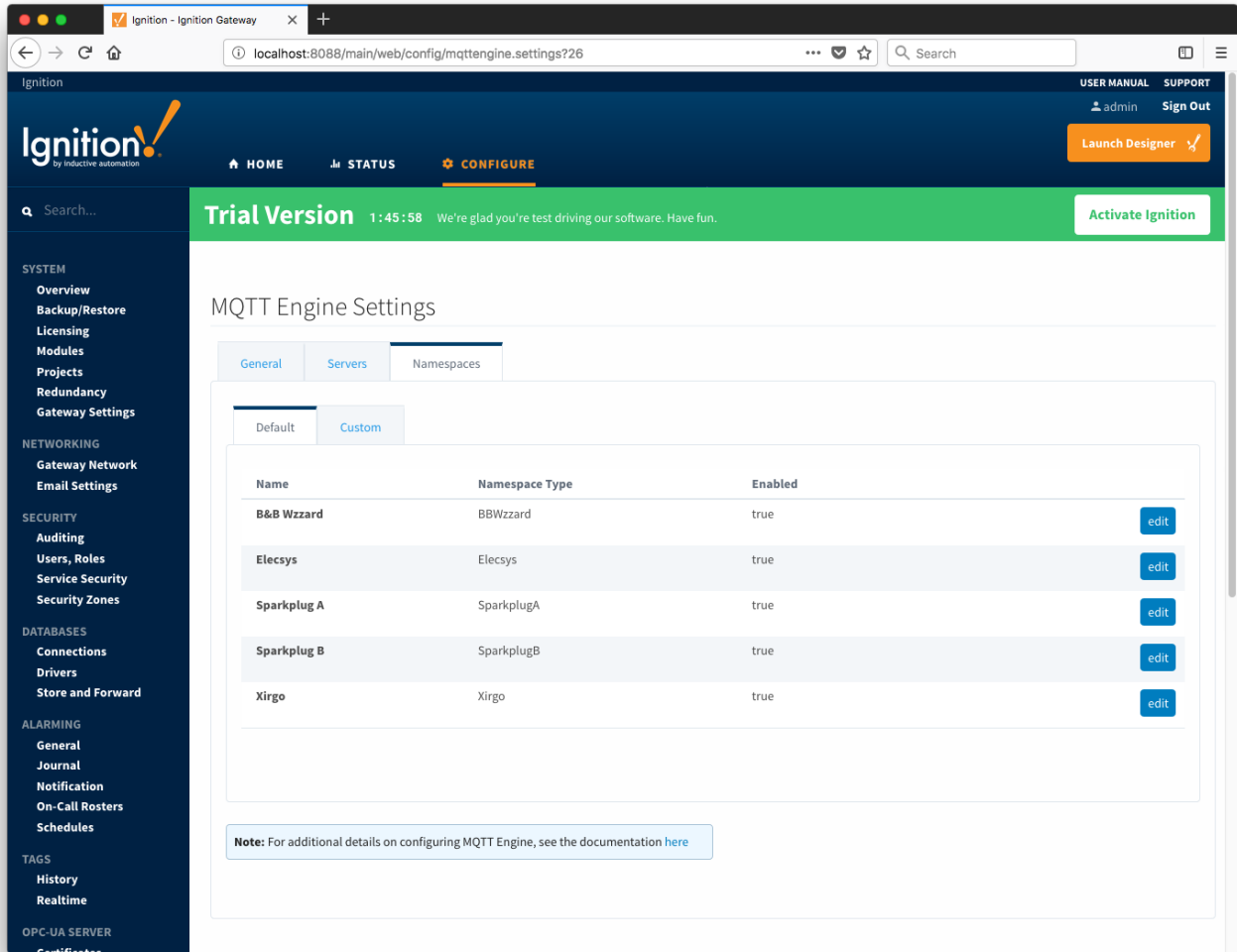
Default Namespaces

Default namespaces are provided out of the box and can simply be enabled or disabled. When MQTT Engine is first installed, all default namespaces are enabled. Each default namespace has the following properties:

- **Name**
 - A friendly name of the namespace to easily identify it.

- **Namespace Type**
 - A namespace type.
- **Enabled**
 - Whether or not the namespace is enabled.

If a namespace is enabled, MQTT Engine will subscribe to the topics necessary to provide support for devices and data associated with that namespace. If a namespace is disabled, MQTT Engine will unsubscribe from those topics and no longer support the devices and data associated with that namespace.



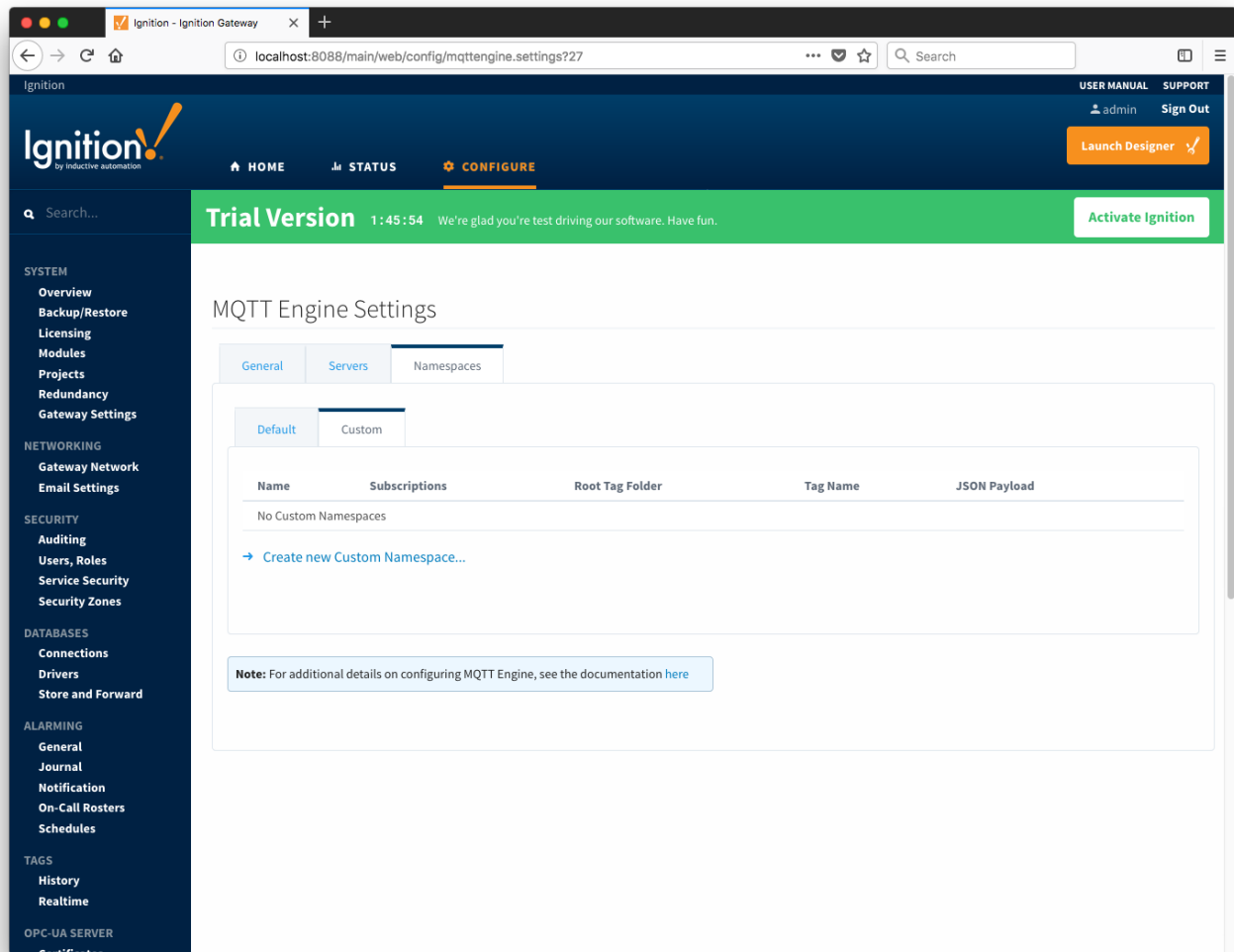
The screenshot shows the Ignition Gateway web interface. The top navigation bar includes links for HOME, STATUS, and CONFIGURE. A green banner indicates a 'Trial Version' with a timer at 1:45:58. The left sidebar contains a search bar and a list of system categories: SYSTEM, NETWORKING, SECURITY, DATABASES, ALARMING, TAGS, and OPC-UA SERVER. The main content area is titled 'MQTT Engine Settings' and has three tabs: General, Servers, and Namespaces. The 'Namespaces' tab is active, showing a table with columns for Name, Namespace Type, and Enabled. There are also 'edit' buttons for each row. A note at the bottom states: 'Note: For additional details on configuring MQTT Engine, see the documentation [here](#)'.

Name	Namespace Type	Enabled	
B&B Wizzard	BBWizzard	true	edit
Elecsys	Elecsys	true	edit
Sparkplug A	SparkplugA	true	edit
Sparkplug B	SparkplugB	true	edit
Xirgo	Xirgo	true	edit

Note: For additional details on configuring MQTT Engine, see the documentation [here](#)

Custom Namespaces

Custom namespaces are used to provide support for generic MQTT messages with string based payloads. If a custom namespace is configured MQTT Engine will convert all messages received to tags. The topic of each message will directly translate into the tag's path. The payload of the message will be that tag's value.



Each custom namespace has the following properties:

Main

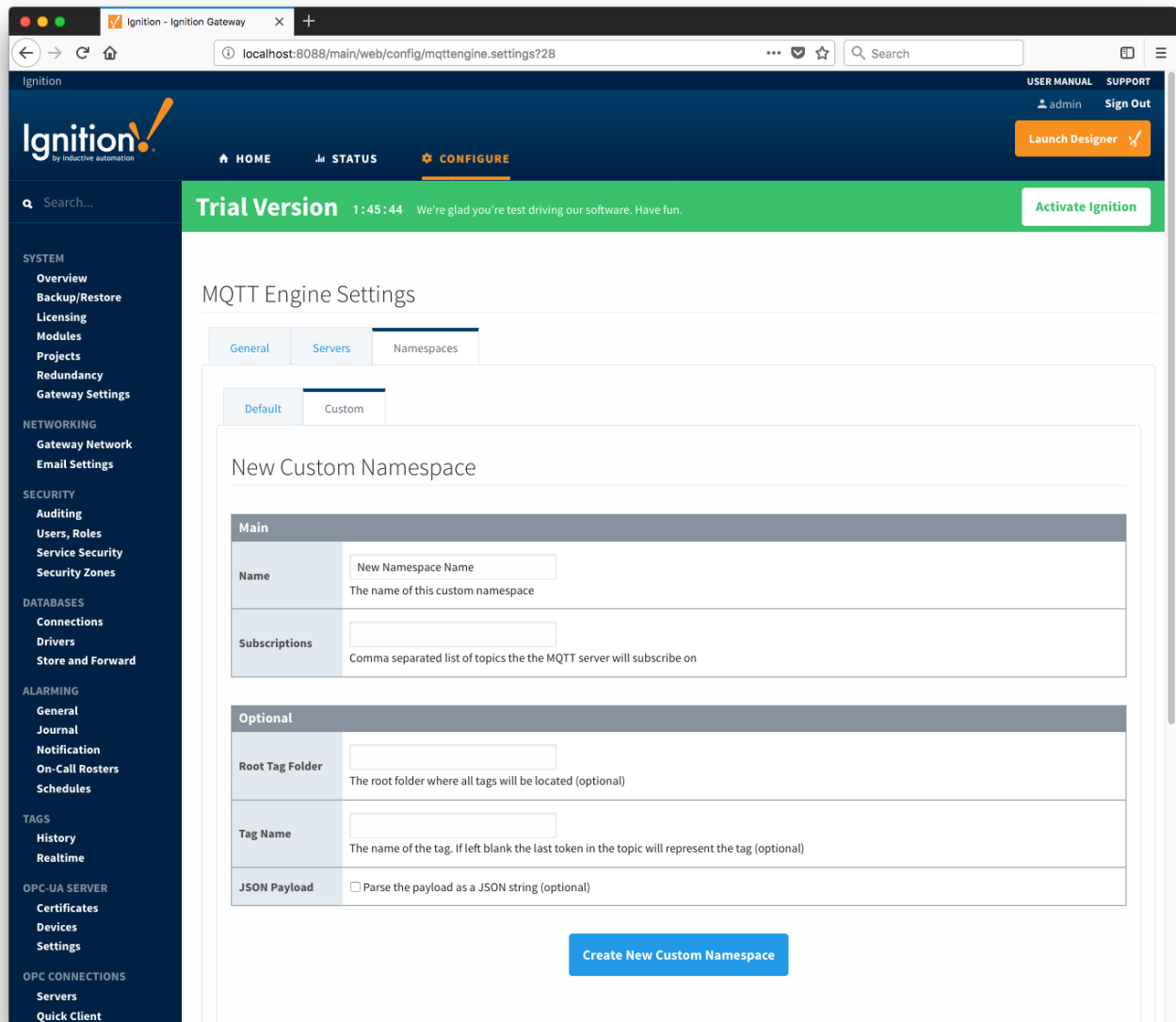
- **Name**
 - A friendly name of the namespace to easily identify it.
- **Subscriptions**
 - A comma separated list of subscriptions.

Optional

- **Root Tag Folder**
 - A name of a folder where all tags will be stored. If configured, this folder will be the base folder where all tag paths will start.
- **Tag Name**
 - A tag name to be used for all tags. If not configured, the last token in the topic will represent the tag.
- **JSON Payload**
 - A flag to indicate that the content of the string based payload is a JSON object.

Note: if a Tag Name is not specified, care must be taken so that published messages do not end up overwriting previous tags.

Clicking on the 'Create new Custom Namespace...' link will bring up the following form to add a new Custom Namespace.



Custom Namespace Example

Let say we have a publish received on the topic "test/data/point" with value "12345". If no Tag Value is configured, and a message is received, MQTT Engine will create a two folders "test" and "data" and a tag "point" with the value of "12345".

Ignition - Ignition Gateway

localhost:8088/main/web/config/mqttengine.settings?29

Search

Ignition
by inductive automation

[HOME](#)[STATUS](#)[CONFIGURE](#)

[USER MANUAL](#)[SUPPORT](#)

adminSign Out

Launch Designer

Search...

SYSTEM

Overview

Backup/Restore

Licensing

Modules

Projects

Redundancy

Gateway Settings

NETWORKING

Gateway Network

Email Settings

SECURITY

Auditing

Users, Roles

Service Security

Security Zones

DATABASES

Connections

Drivers

Store and Forward

ALARMING

General

Journal

Notification

On-Call Rosters

Schedules

TAGS

History

Trial Version 1:43:50

We're glad you're test driving our software. Have fun.

Activate Ignition

MQTT Engine Settings

GeneralServersNamespaces

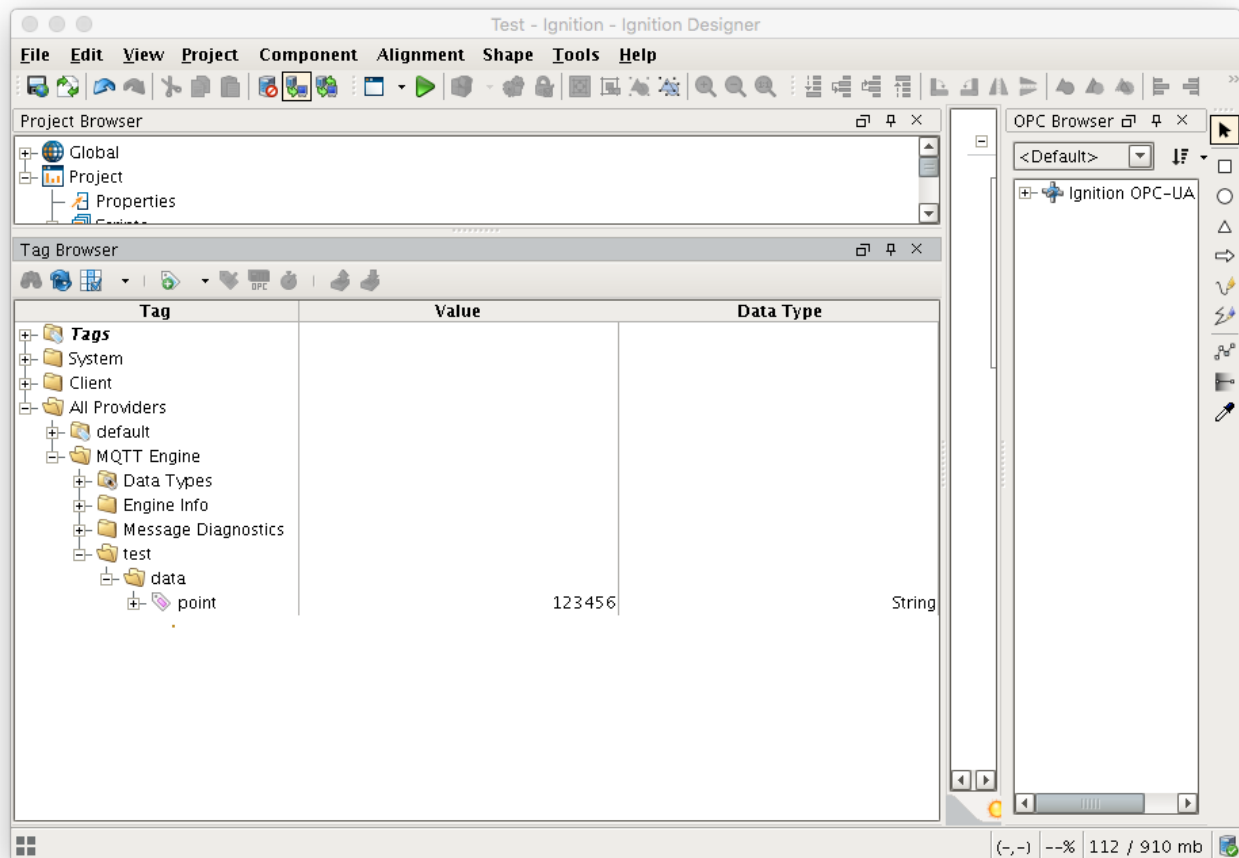
DefaultCustom

Successfully created new Custom Namespace "New Namespace Name"

Name	Subscriptions	Root Tag Folder	Tag Name	JSON Payload	
New Namespace Name	test/#			false	deleteedit

Create new Custom Namespace...

Note: For additional details on configuring MQTT Engine, see the documentation [here](#)



Alternatively if a Tag Name is configured, lets call it "payload", then MQTT Engine will convert each token in the topic to a folder and create a tag called "payload" with the value "12345"

Ignition - Ignition Gateway

localhost:8088/main/web/config/mqttengine.settings?31

Search

Ignition
by inductive automation

[HOME](#)[STATUS](#)[CONFIGURE](#)

[USER MANUAL](#)[SUPPORT](#)

adminSign Out

Launch Designer

Search...

SYSTEM

Overview

Backup/Restore

Licensing

Modules

Projects

Redundancy

Gateway Settings

NETWORKING

Gateway Network

Email Settings

SECURITY

Auditing

Users, Roles

Service Security

Security Zones

DATABASES

Connections

Drivers

Store and Forward

ALARMING

General

Journal

Notification

On-Call Rosters

Schedules

TAGS

History

Trial Version1:43:09

We're glad you're test driving our software. Have fun.

Activate Ignition

MQTT Engine Settings

GeneralServersNamespaces

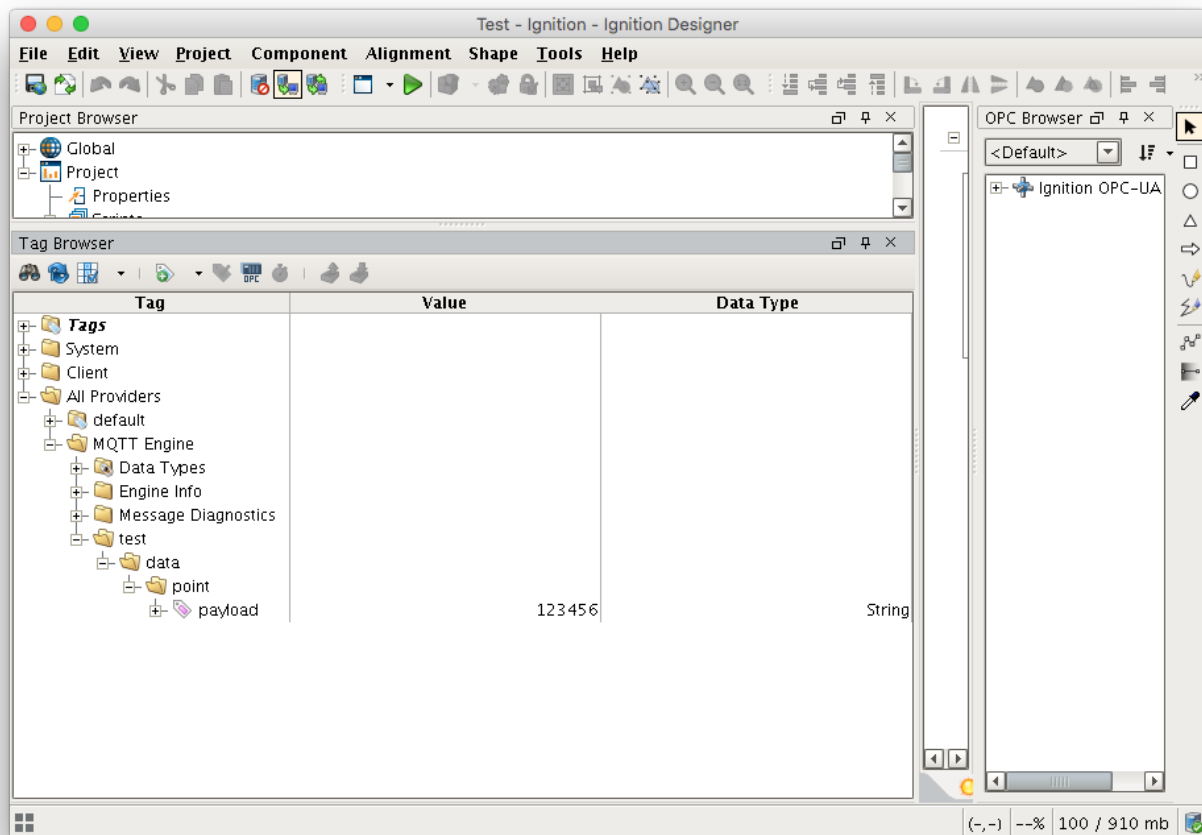
DefaultCustom

Successfully updated Custom Namespace "New Namespace Name"

Name	Subscriptions	Root Tag Folder	Tag Name	JSON Payload	
New Namespace Name	test/#		payload	false	deleteedit

Create new Custom Namespace...

Note: For additional details on configuring MQTT Engine, see the documentation [here](#)



In most cases it is useful to specify a Tag Name in order to prevent cases when a publish on a topic can overwrite a previously created tag, changing it into a folder. Consider the case where you have the following two publishes:

1st publish on topic "one/two" will create a tag named "two" in the folder "one"

2nd publish on topic "one/two/three" will create a tag named "three" in the folder "one/two"

When the 2nd publish is received it will overwrite the first tag because "two" is now a folder instead of a tag. This folder/tag name collision can be avoided by specifying the Tag Name to always use for tags.

Custom Namespace JSON Example

Let say we have a publish received on the topic "test/data/json" with value '{ "stringTag" : "12345", "folderTag" : { "intTag" : 1, "boolTag" : true } }'. MQTT Engine will create a three folders "test", "data" and "point" followed by a tag/folder structure representing the JSON value of the payload.

Ignition - Ignition Gateway

localhost:8088/main/web/config/mqttengine.settings?35

Search

USER MANUALSUPPORTadminSign OutLaunch Designer

Ignition
by inductive automation

HOMESTATUSCONFIGURE

Search...

SYSTEM

OverviewBackup/RestoreLicensingModulesProjectsRedundancyGateway Settings

NETWORKING

Gateway NetworkEmail Settings

SECURITY

AuditingUsers, RolesService SecuritySecurity Zones

DATABASES

ConnectionsDriversStore and Forward

ALARMING

GeneralJournalNotificationOn-Call RostersSchedules

TAGS

History

Trial Version 1:39:17 We're glad you're test driving our software. Have fun.

Activate Ignition

MQTT Engine Settings

GeneralServersNamespaces

DefaultCustom

Successfully updated Custom Namespace "New Namespace Name"

Name	Subscriptions	Root Tag Folder	Tag Name	JSON Payload	
New Namespace Name	test/#			true	deleteedit

Create new Custom Namespace...

Note: For additional details on configuring MQTT Engine, see the documentation [here](#)

