Sending OPC Tag Data with Transmission

Prerequisites:

- Installing the Java Runtime Environment
- Installing Ignition
- Installing the following MQTT Modules
  - MQTT Distributor
    - v2.1.X if using Ignition 7.7.x or 7.8.x
    - v3.1.0 or greater if using Ignition 7.9.X
  - MQTT Engine
    - v2.1.X if using Ignition 7.7.x or 7.8.x
    - v3.1.0 or greater if using Ignition 7.9.X
  - MQTT Transmission
    - v2.1.X if using Ignition 7.7.x or 7.8.x
    - v3.1.0 or greater if using Ignition 7.9.X
- A device that supports Modbus over TCP

Overview:

Transmission is an MQTT module for Ignition that can convert Ignition tag data and tag change events into MQTT messages to be consumed by MQTT Engine or other MQTT clients. This tutorial will show how to configure MQTT Transmission to send OPC tag data in Ignition as MQTT messages via MQTT Distributor to MQTT Engine where they will be displayed.

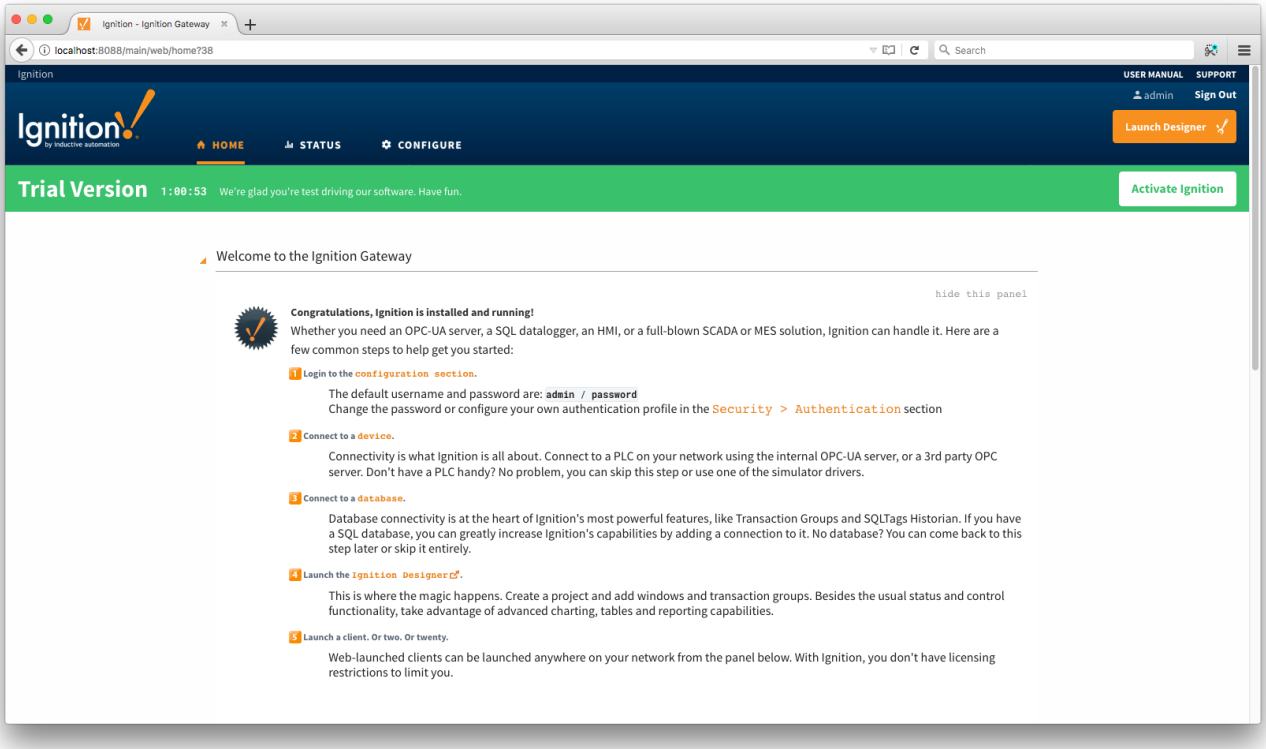
The topology of this example shows MQTT Distributor, MQTT Engine, and MQTT Transmission all running in the same Ignition instance. This is done for simplicity of the tutorial. But, this isn't required or even intended to be a real use case. In a more realistic scenario MQTT Transmission and MQTT Engine would be located on separate machines.

Variations:

This tutorial shows how to use OPC tags and MQTT Transmission to generate MQTT messages based on tag change events. However, the tag source does not have to be an OPC tag. Instead, as long as the tag structure for MQTT Transmission is followed, any Ignition tag can be used to generate MQTT messages and/or be controlled via MQTT messages.

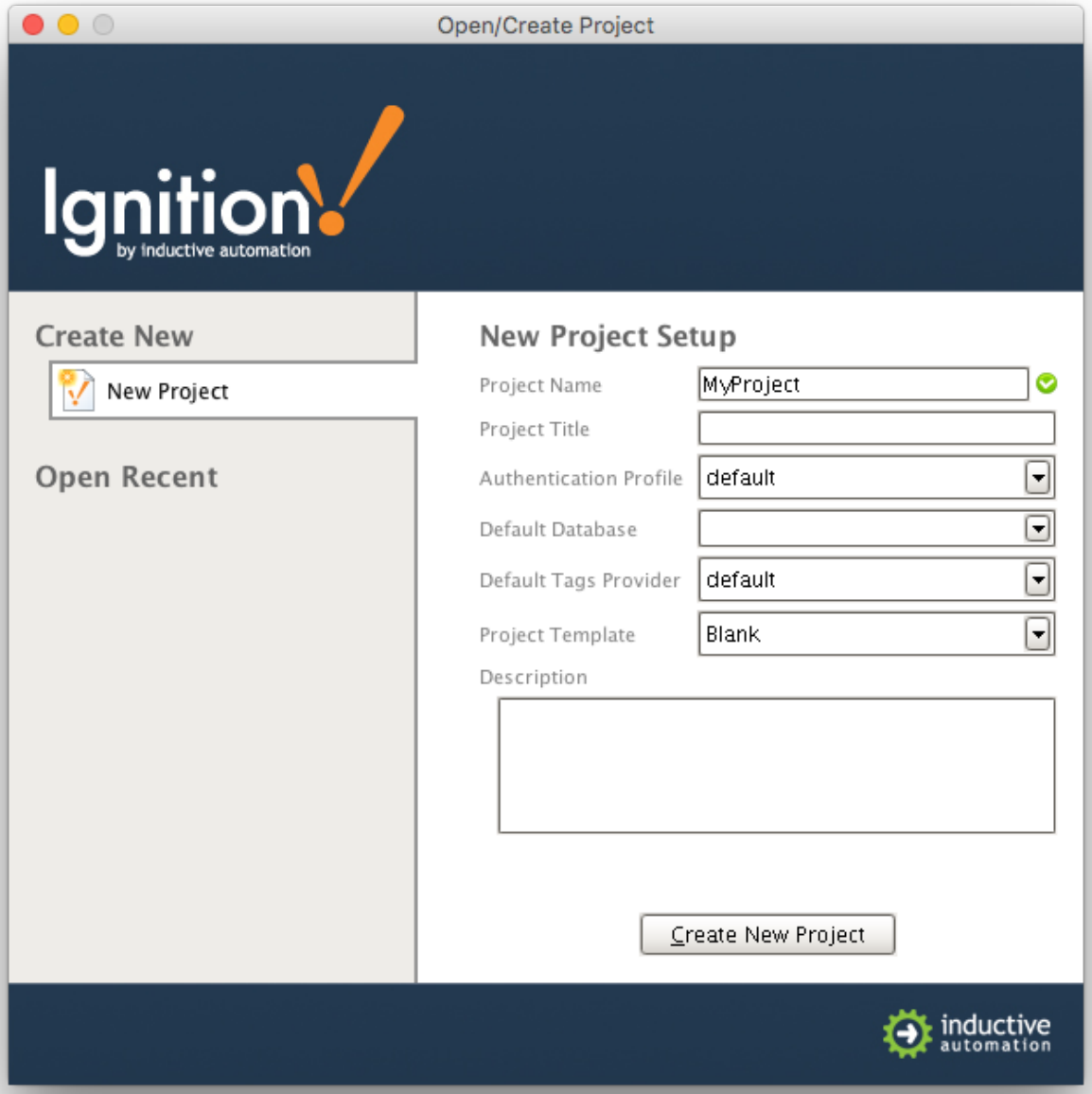
Sending OPC Tag Data with Transmission:

The first step is to configure the tag provider in Ignition in a way that MQTT Transmission understands. Start by configuring your OPC server, client, and tags. This can be done using the Inductive Automation documentation here. Once this is done, the Tag Provider needs to be set up in Ignition via the Ignition Designer. Using a Web Browser, browse to the Ignition Gateway on your Ignition Gateway. If it is running on your development machine, that is: http://localhost:8088. You should see this:

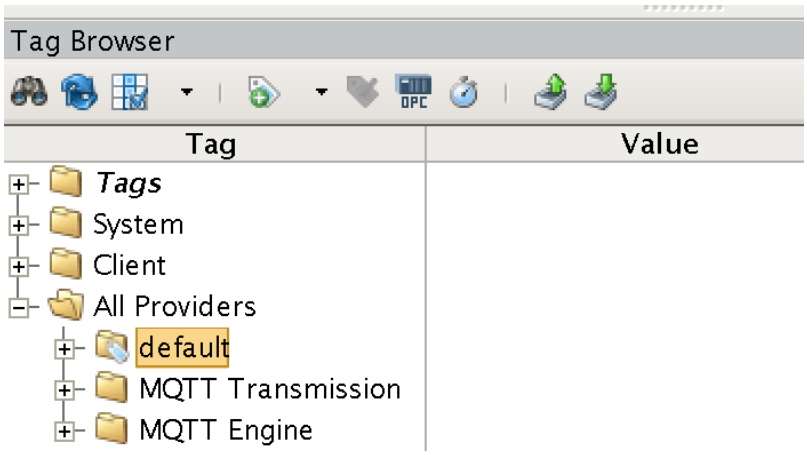


Near the upper right corner, click 'Launch Designer'. This will open the following window after downloading the JNLP file and executing it. Note the default username/password is admin/password. Type those into the appropriate fields and click 'Login'.

This will bring you to a new Window where you can select an Ignition Project or create a new one. Create a new project by giving it a name and clicking 'Create New Project'.

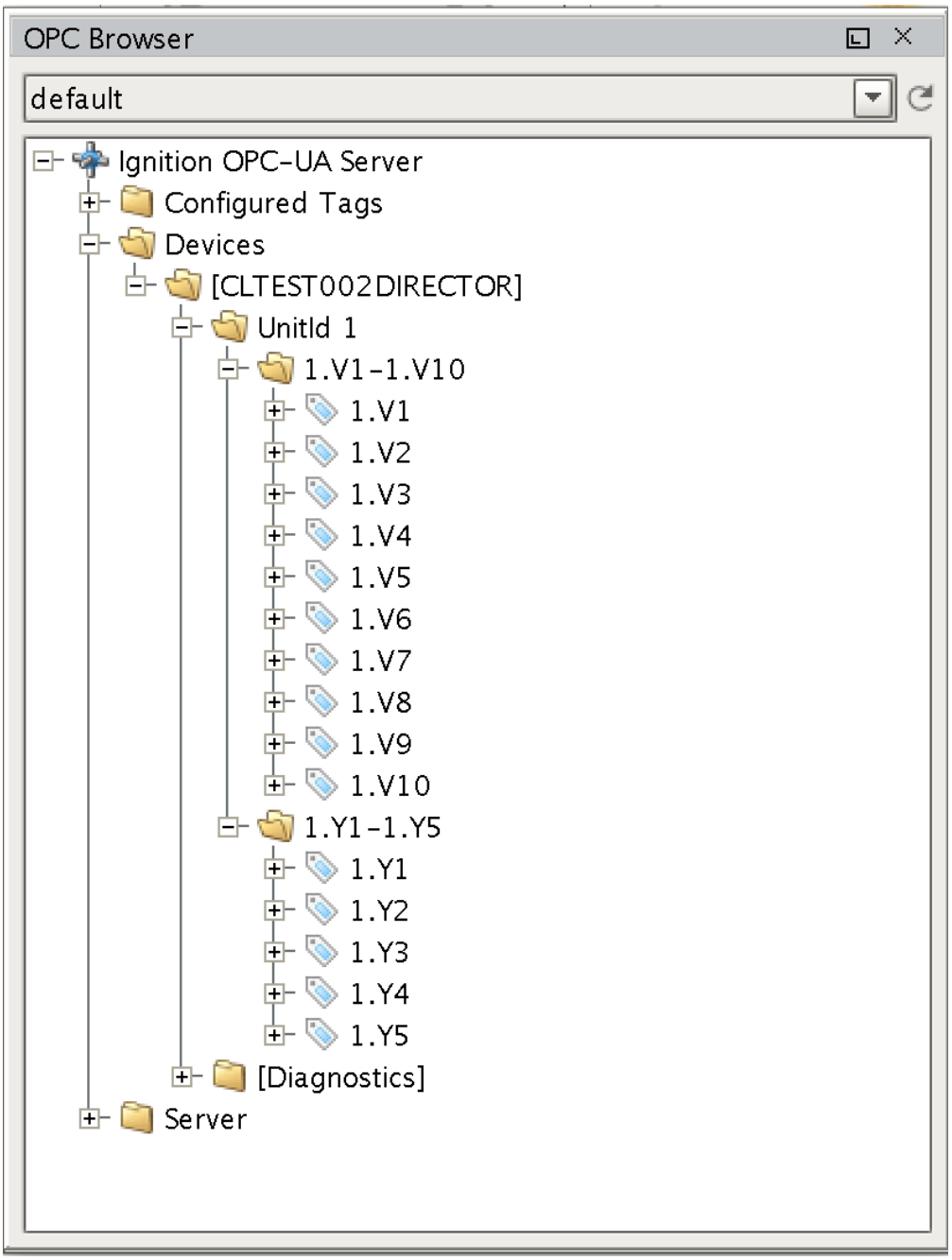


In this example we're going to use the "default" tag provider within Ignition Enterprise. Expand 'All Providers' in the Tag Browser and select 'default'. Note that if Ignition Edge is being used, the tag provider will be named 'edge'.

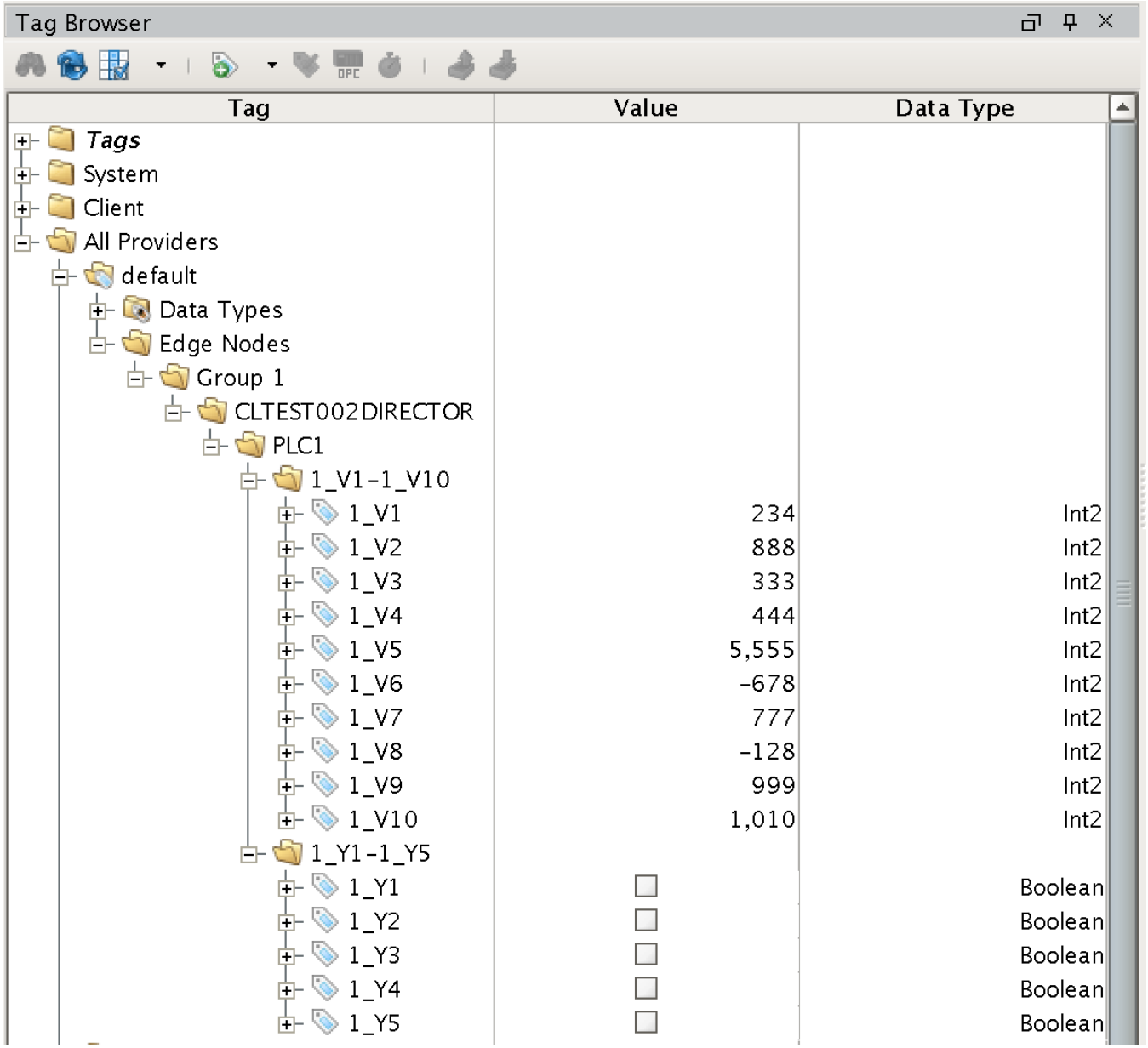


With 'default' (or 'edge') selected, click the 'OPC' icon in the Tag Browser icon list:

This will open a new window as shown below. If the OPC server and client were set up and configured properly, you should see something similar to the following:



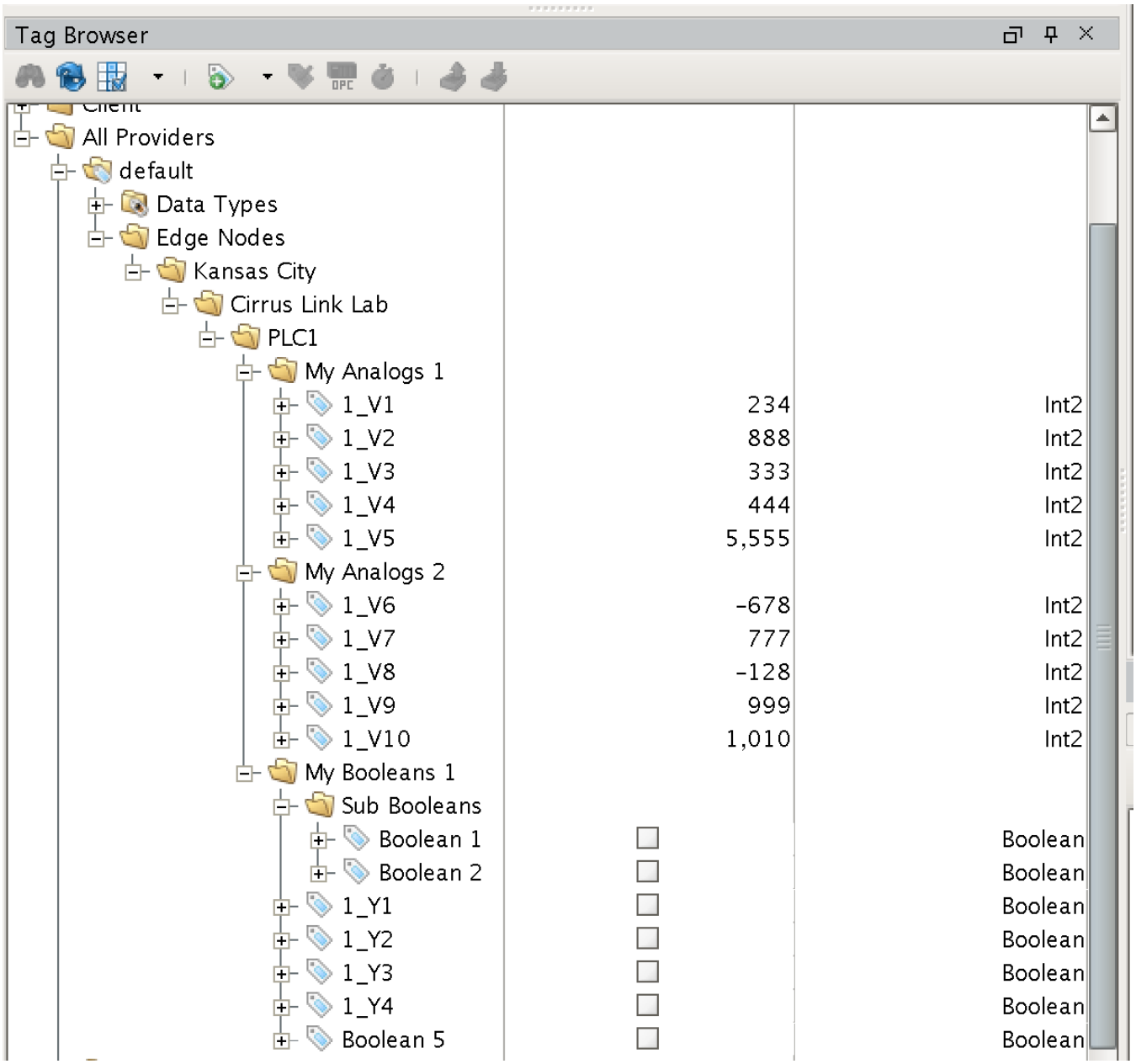
Note there is a device with an attached PLC and two sets of registers. Yours will look different based on the device you are using and how it is configured. At this point, we can select the device (CLTEST002DIRECTOR in this case) and drag it into the Tag Browser under the 'default' Tag Provider. This is shown below:



The first level of folders is very important in terms of layout and how the tags will be understood and represented by MQTT Transmission. These are the rules:

- For terminology reference, please review the Sparkplug specification if you have not done so.
- Under the Tag Provider, the first folder must be exactly 'Edge Nodes'
- Under the 'Edge Nodes' folder a 'Group ID' must be next. This can be anything you want but realize it represents a group of edge nodes. You can have as many group IDs as you want.
- Under 'Group ID' you can one or more Edge Nodes.
- Under each Edge Node, you can have one or more devices.
- Under each device, you can have any depth of folder/tag structure to represent the tags.

In our example, we have a single Edge Node of 'CLTEST002DIRECTOR', under it a single Device called PLC1, and under it, 15 registers in two different folders. We could rearrange this by renaming and moving folders and tags to a different representation as desired. The layout below is also valid as was done simply by moving/renaming the tags shown above.

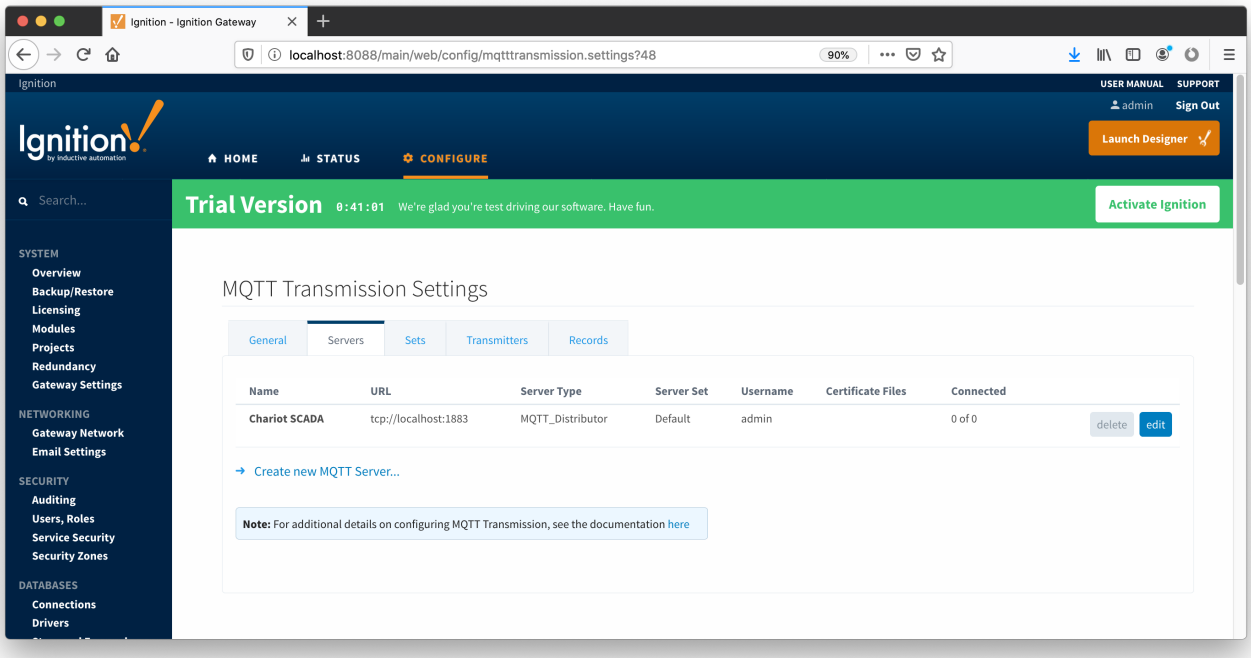


Note while the folders and tags were moved and renamed, the required basic structure stayed the same with:

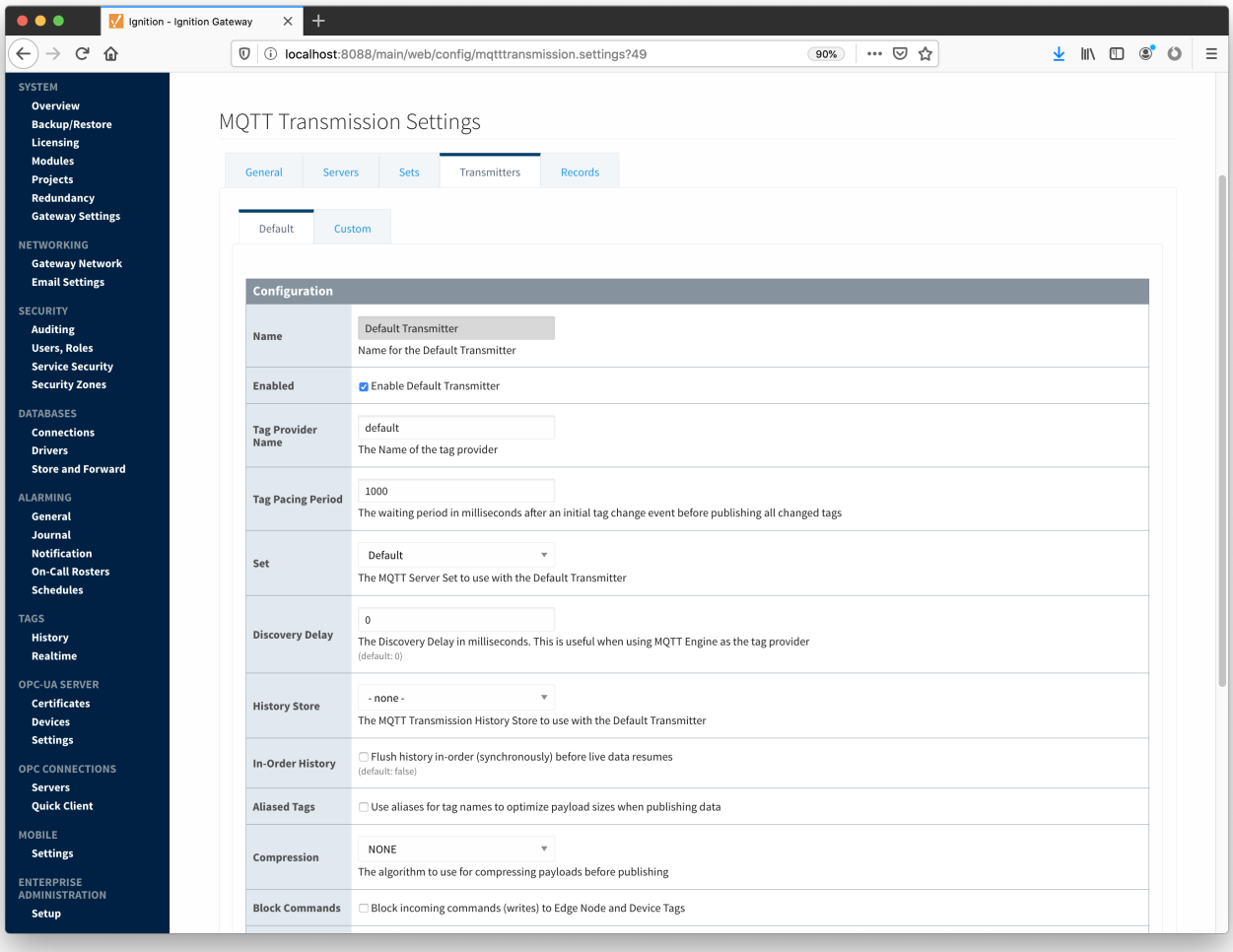
- default
  - Edge Nodes
    - Group ID (1 to n)
      - Edge Node ID (1 to n)
        - Device ID (1 to n)
          - Free form folders and tags

With our tags set up as we want them, we now must configure MQTT Transmission. Do so by browsing to the the Configure section of the Ignition Gateway web UI in the left-hand navigation pane. Find and select 'Configure MQTT Transmission Settings':

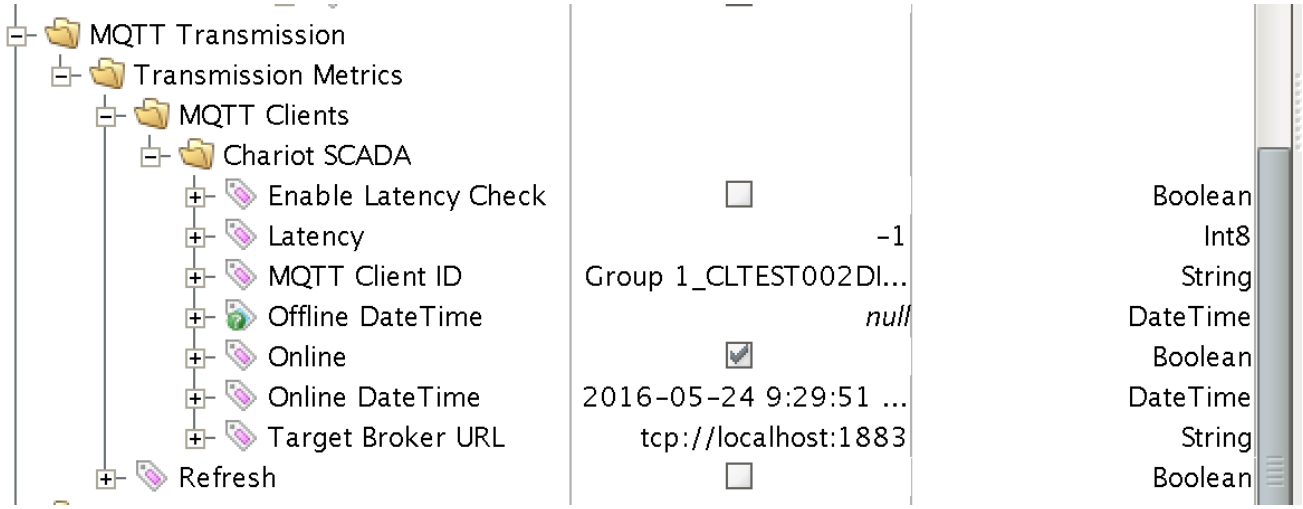
In this example we're using the default MQTT Server of MQTT Distributor:



Under the Transmitters tab, we're also using the defaults:

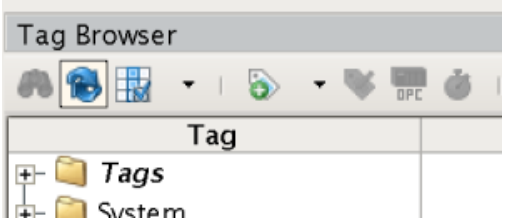


Note the Tag Provider Name is the same Tag Provider we build out our Edge Nodes from in the Tag Browser earlier. Now we can go back to Designer and force MQTT Transmission to update. MQTT Transmission does not dynamically look for changes in the tag structure and update them. If it did, you could end up with a lot of improperly structured data while the changes to the tag tree are being made. So, the update must be forced via 'Refresh'. This is a tag under the MQTT Transmission tag provider as shown below in the Tag Browser.

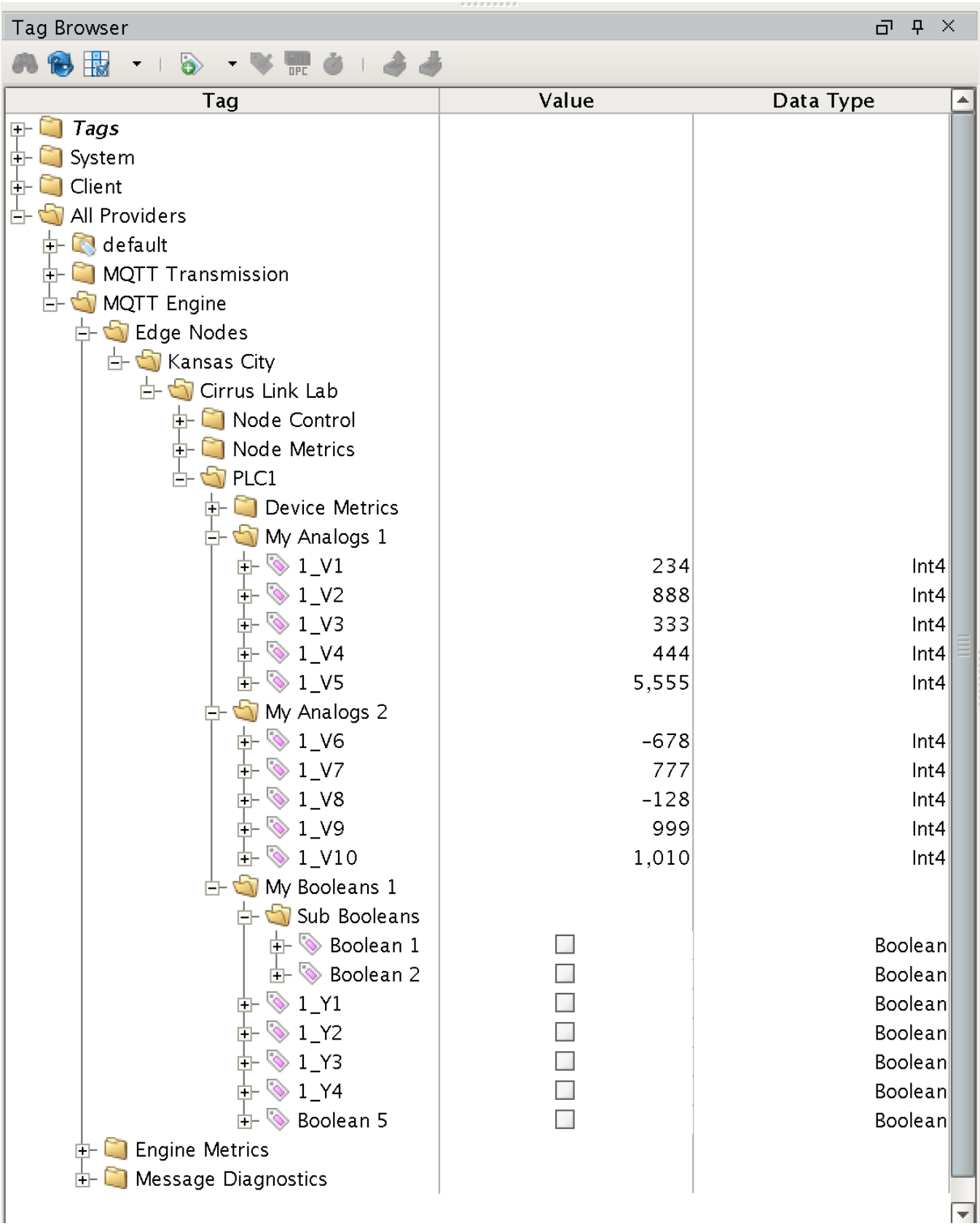


In order to refresh, Designer must be in read/write and preview mode. Do so by selecting these two buttons in the top menu of Designer:

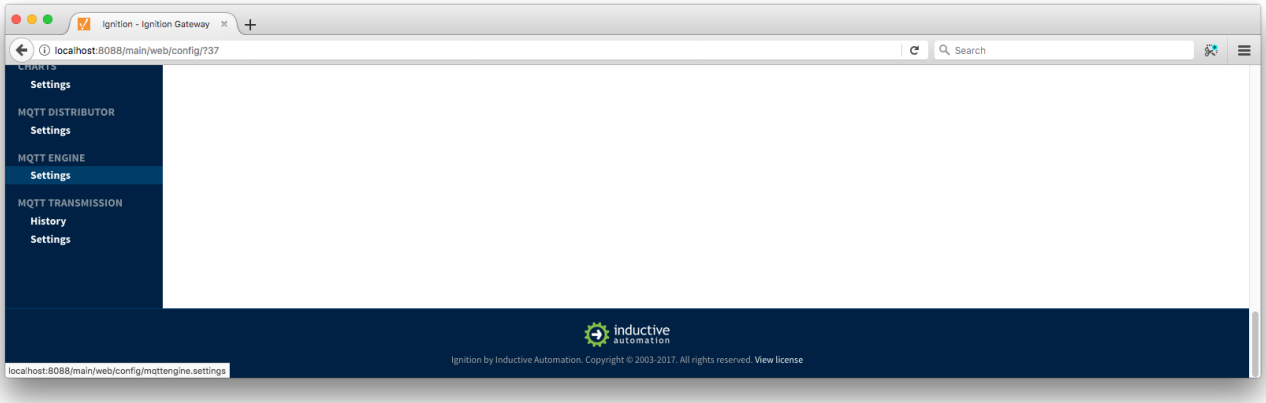
Once this is done, click 'Refresh' in the Tag Browser. This will force MQTT Transmission to read the default Tag Browser tree, find 'Edge Nodes', and begin sending MQTT messages based on tag change events. You can see this by clicking the refresh icon in the Tag Browser menu:



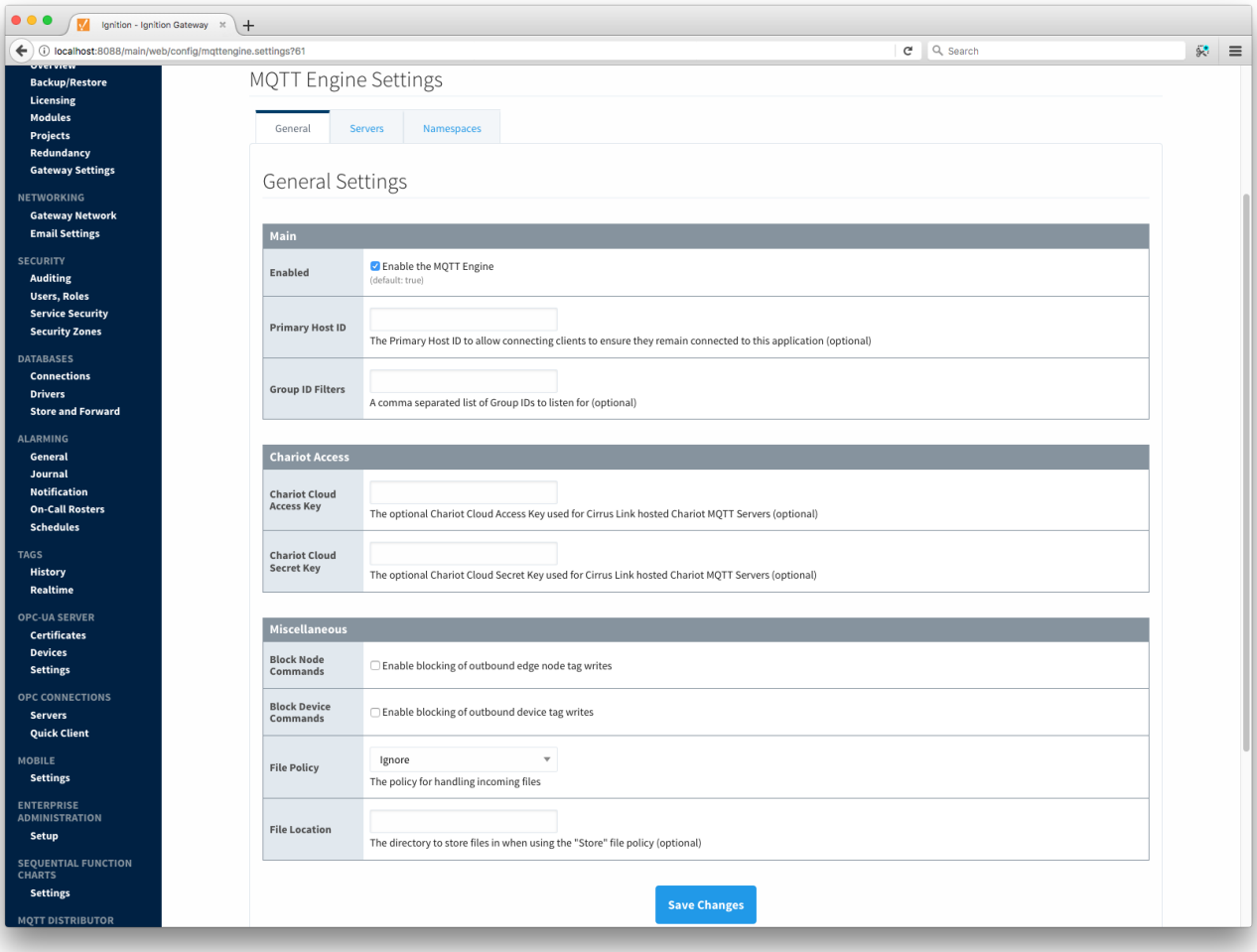
At this point, you should be able to expand the MQTT Engine tag provider and see all of the tags in MQTT Engine:



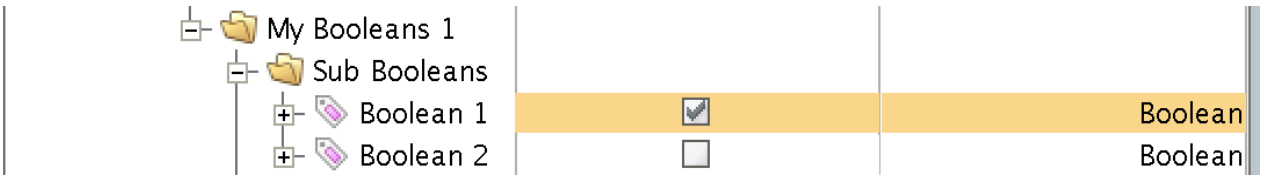
In addition to the tags being displayed in Engine, they are also writable if this enabled in MQTT Engine. By default, MQTT Engine blocks command messages from being sent to devices. To enable this feature, in the Ignition web console browse to the MQTT Engine Module Settings.



Make sure the "Block Node Commands" and "Block Device Commands" settings are disabled, as shown below.



With this enabled and Designer in read/write and preview mode, you can write to the outputs of the modbus device from the Tag Browser:



Note there is some delay in the response. This is due in part to MQTT Transmissions 'Tag Pacing Period'. This is the delay for MQTT messages to wait before being sent to allow multiple change events to buffer before putting them into a single MQTT message. This can be changed in the MQTT Transmission module configuration in the Ignition web console.

Additional Resources

- Inductive Automation's Ignition download with free trial
  - https://inductiveautomation.com/downloads/
- Azure Injector download with free trial
  - https://inductiveautomation.com/downloads/third-party-modules
- Questions about this tutorial?
  - Check out the Cirrus Link Forum: https://forum.cirrus-link.com/
  - Contact support: support@cirrus-link.com
- Sales questions
  - Email: sales@cirrus-link.com
  - Phone: +1 (844) 924-7787
- About Cirrus Link
  - https://www.cirrus-link.com/about-us/