I have data that is toggling between stale and healthy at my subscribing MQTT Client

There are two common causes for this issue – colliding MQTT Client IDs or colliding Sparkplug Edge Node Descriptors.

Colliding MQTT Client IDs occur when there are two or more MQTT clients connecting to an MQTT broker using the same Client ID. The broker uses the Client ID to identify the client and the current state of the client and therefore this ID must be unique per client and broker.

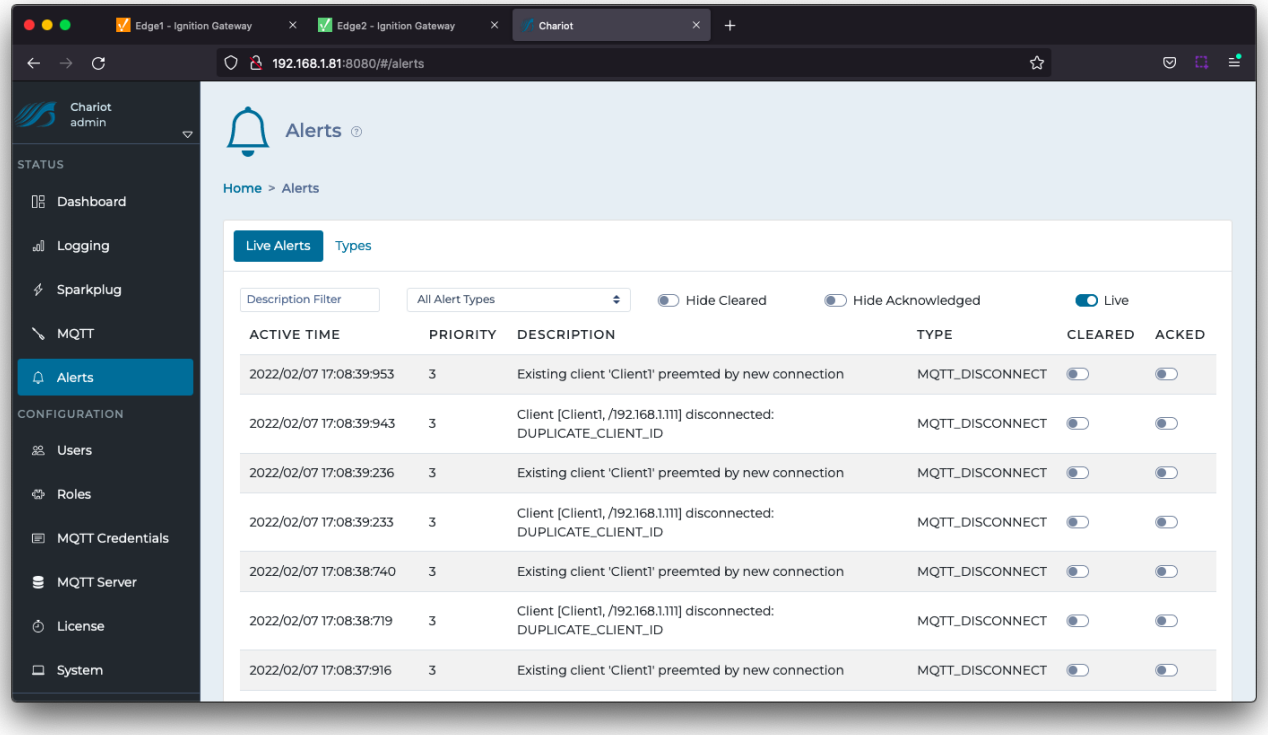
Colliding Edge Node Descriptors occur when there are two or more Edge Nodes publishing with a topic namespace that does not have a unique combination of Group ID and Edge Node ID. In a Sparkplug compliant system, it is this combination of IDs that identifies the Edge Node and is the 'Sparkplug Edge Node Descriptor'. Every Sparkplug Edge Node Descriptor within a Sparkplug environment must be unique.

Let's start by confirming the connection status of the Edge Nodes with your Chariot or MQTT Distributor server instance to identify which issue you have.

Chariot

From the Chariot UI navigate to Alerts in the left menu bar. Select Types and enable the alerts for MQTT_DISCONNECT

Under Live Alerts, if we can see in the logs that Chariot is logging the DUPLICATE_CLIENT_ID description, as shown below, you have Colliding Client IDs. If not, we have a Colliding Sparkplug Edge Node Descriptors issue.



MQTT Distributor

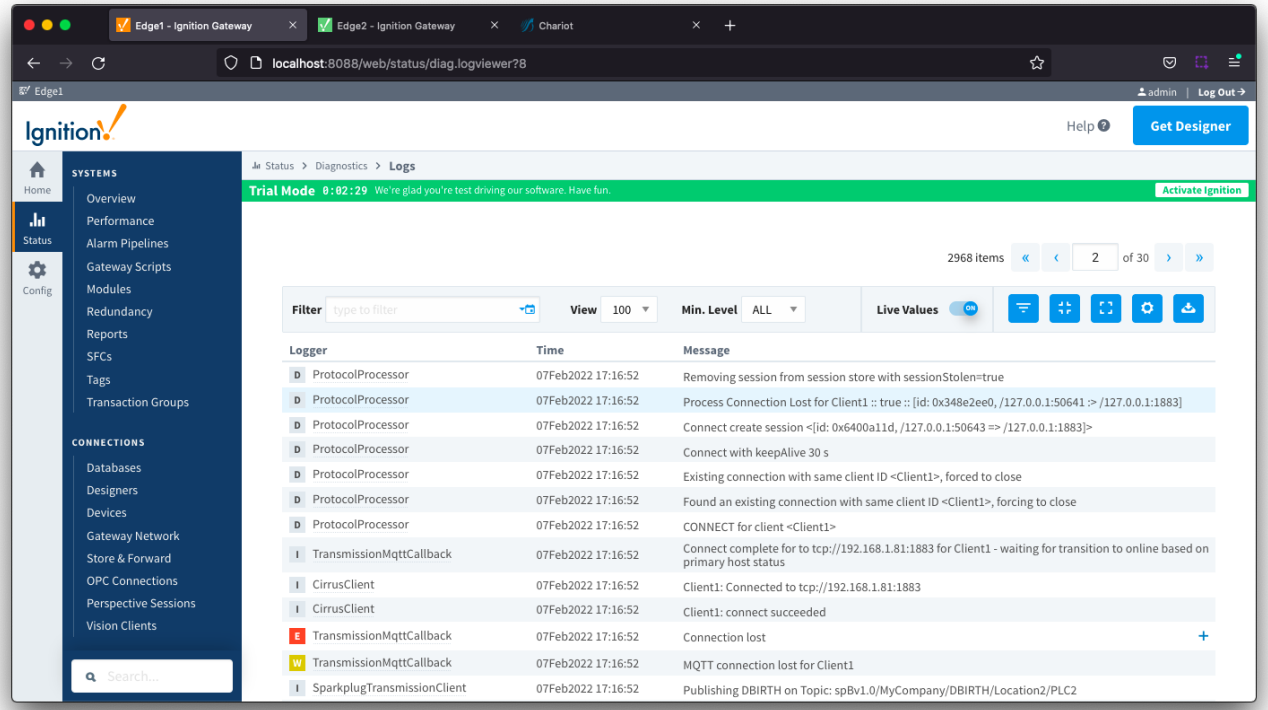
From the Ignition UI connected to your instance of MQTT Distributor, navigate to Status > Diagnostic > Logs.

> Read the user manual Diagnostics - Logs explaining how to use the Logs console in Ignition

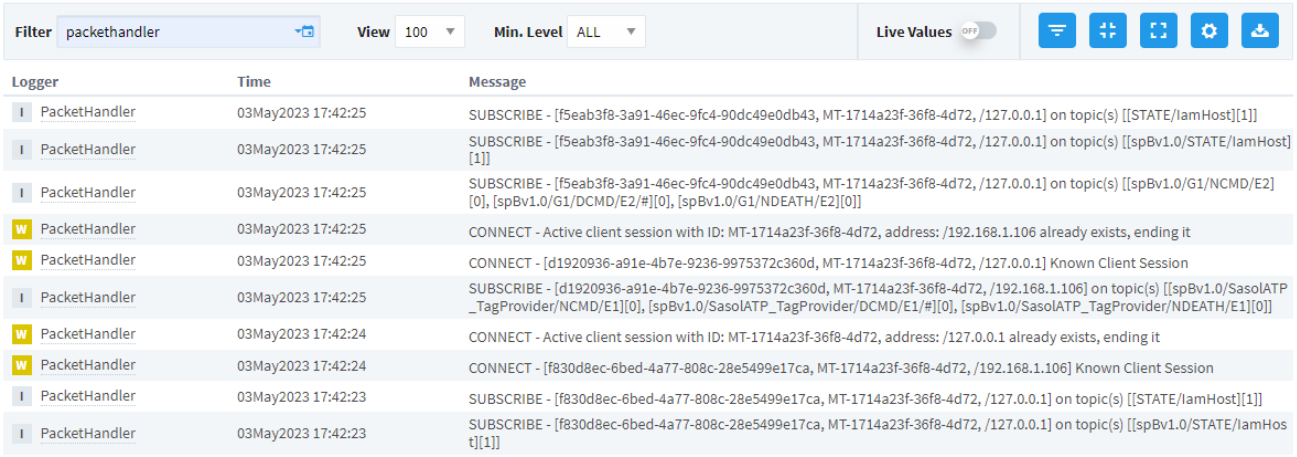
If we can see in the logs that the MQTT broker is continually forcefully disconnecting an existing connection to allow another client with the same Client ID to connect, as shown below, you have Colliding Client IDs. If not, we have a Colliding Edge Node Descriptors issue.

The logging shows both the Client Id and the associated IP addresses.

If running MQTT Distributor 4.0.13 or earlier, set the debug level for the io.moquette.spi.impl.ProtocolProcessor logger to TRACE and set the filter of the Logs view to ProtocolProcessor.



If running MQTT Distributor 4.0.14 or later, logging will come out as warnings for the com.cirruslink.chariot.server.core.PacketHandler logger.



Resolving Colliding Client ID

To resolve the colliding Client IDs you will need to review your system configurations on the physical Edge Nodes identified and remove the conflicts.

In the logs if you see different IP addresses for the Edge Nodes attempting to connect with the same Client ID, then the same MQTT Client ID has been set on different physical Edge Nodes. Review the configuration for physical Edge Nodes with these IP addresses.

If using MQTT Transmission, there are two additional scenarios to consider if the logs show the same IP address for the Edge Nodes attempting to connect with the same Client ID.

1. The MQTT Client ID is set on a single physical Edge Node device where a single Transmitter is dynamically picking up multiple virtual Edge Nodes.
2. The MQTT Client ID is set on a single physical Edge Node where multiple transmitters are configured for one or more virtual Edge Nodes.

In either of these two setups, the MQTT connection for each virtual Edge Node requires a unique Client ID. The Client ID in the the MQTT Transmission Configuration should be left blank allowing MQTT Transmission to auto-generate unique Client IDs for each Edge Node connection.

Refer to the MQTT Transmission Transmitters and Tag Trees Tutorial/HowTo for detail on how a virtual Edge Node is dynamically created.

Colliding Edge Node Descriptors

MQTT Transmission uses the Sparkplug B specification which defines the topic namespace to publish data as spBv1.0/group_ID/message_type/edge_ID/[device_ID]

In a Sparkplug compliant system, it is the combination of Group ID and Edge Node ID that identifies the Edge Node and is the 'Sparkplug Edge Node Descriptor'.

Every Sparkplug Edge Node Descriptor within a Sparkplug environment must be unique because these are used as 'addresses' in the system to identify the edge node. It is a bit like having two houses with the same postal address. It isn't possible for other MQTT clients in the system to tell where messages are coming from and when sending messages to them, they will both receive the messages.

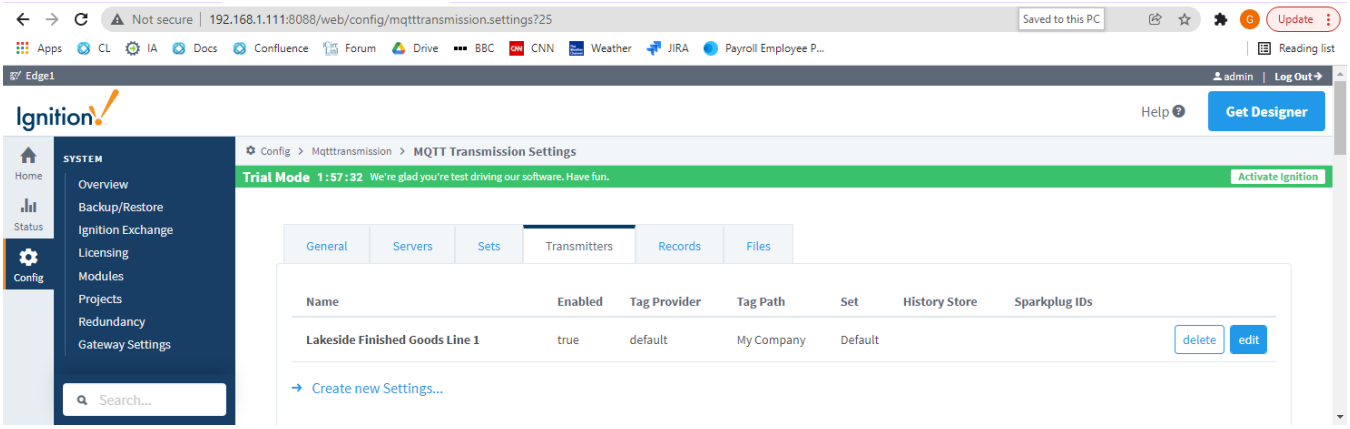
The topic used for tags published to the MQTT Server is a combination of the MQTT Transmission 'Transmitter' configuration as well as the arrangement of tags in the Ignition tag tree. In the Transmitter configuration, the Tag Path points to the folder where the tag tree will start and the next three folders will be picked up as the group_ID, edge_ID and device_ID.

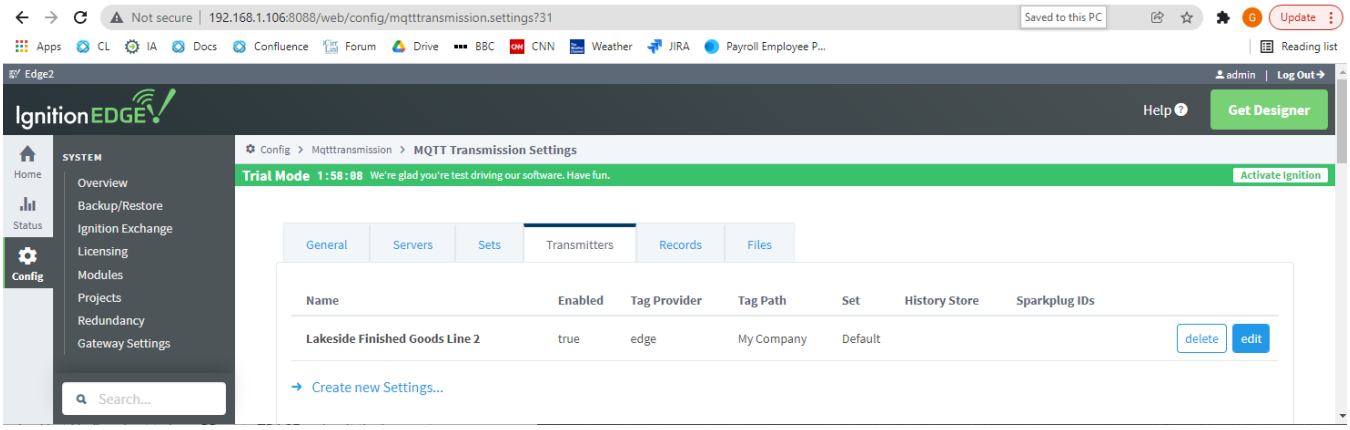
If you have not carefully managed your tag tree structure, you can create duplicate Sparkplug Edge Node Descriptors.

You can override the functionality of pulling the namespace directly from the tag path by setting the Sparkplug IDs directly for the Group ID, Edge Node ID and, optionally, Device ID. If configured, these elements will be used in the topic namespace and the payload will be the folders pointed to by the Tag Path.

As an example we have two physical Edge Nodes setup with a single Transmitter configured on each.

Whilst the worked example below uses two physical Edge Nodes setup with a single Transmitter configured on each, this can also occur on a single physical Edge Node and the same trouble shooting steps will apply.





The Ignition tag tree structure is *Company/Location/Process Area/Line/PLC* where Line is the physical Edge Node device connected to PLCs. With the tag path for both transmitters set to My Company, we can see that both Edge Nodes will publish on the following namespaces:

Transmitter on Edge 1 named Lakeside Finished Goods Line 1 spV1.0/Lakeside/message_type/Finished Goods/Line1

Transmitter on Edge 2 named Lakeside Finished Goods Line 2 spV1.0/Lakeside/message_type/Finished Goods/Line2

Since the Sparkplug Edge Node Descriptor (group_id = Lakeside and edge_id = Finished Goods) does not uniquely define the edge node, data from these two transmitters will sent with the same topic resulting in the next message sequence number expected by the MQTT client being incorrect.

As a result, the MQTT Client will mark the data as stale and request a rebirth from the transmitter. Depending on the frequency of the published data this manifests as the data from different edge nodes toggling between stale and healthy. If you have multiple MQTT Clients subscribing to the namespace, this will also likely create a firestorm of rebirth requests across the system.

Your browser does not support the HTML5 video element

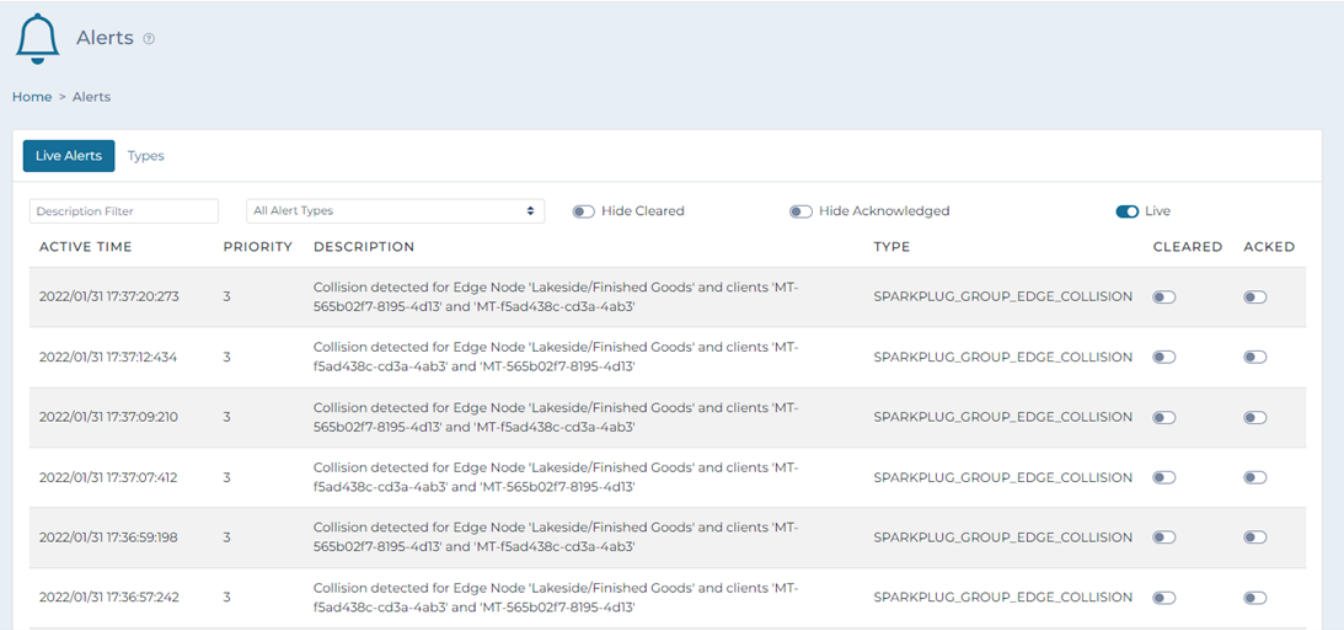
Now we can use the logging associated with your Chariot or MQTT Engine/Distributor instance to determine the physical or virtual Edge Nodes with duplicate Sparkplug Edge Node Descriptors.

Chariot

From the Chariot UI navigate to Alerts in the left menu bar. Select Types and enable the alerts for SPARKPLUG_GROUP_EDGE_COLLISION

Under Live Alerts, we can see in the logs the Edge Node ID along with the Client IDs causing the collisions.

The next step is to review and update as necessary the configuration for each of the listed Client IDs. If using MQTT Transmission, see here for how to identify the Client IDs from Ignition Designer.



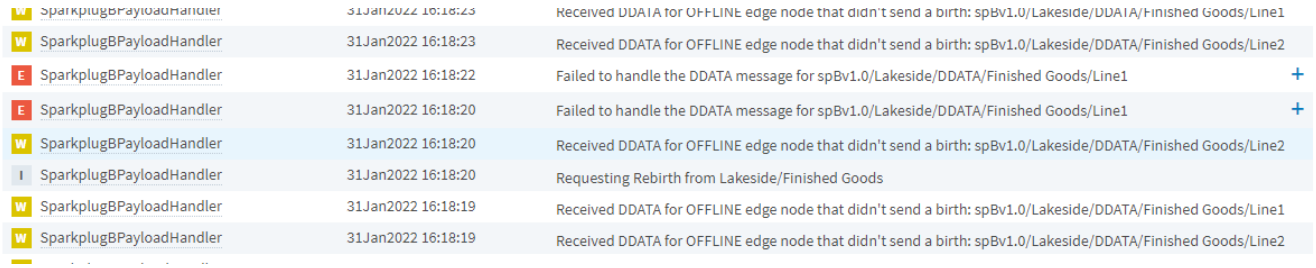
MQTT Engine

From the Ignition UI connected to your instance of MQTT Engine, navigate to Status > Diagnostic > Logs.

Read the user manual Diagnostics - Logs explaining how to use the Logs console in Ignition

Set the debug level for the com.cirruslink.mqtt.engine.gateway.sparkplug.SparkplugBPayloadHandler logger to TRACE and set the filter of the Logs view to SparkplugBPayloadHandler.

You will see errors logged indicating that data messages (type of DDATA) are not being handled correctly along with rebirth requests to the duplicate Sparkplug Edge Node Descriptors.

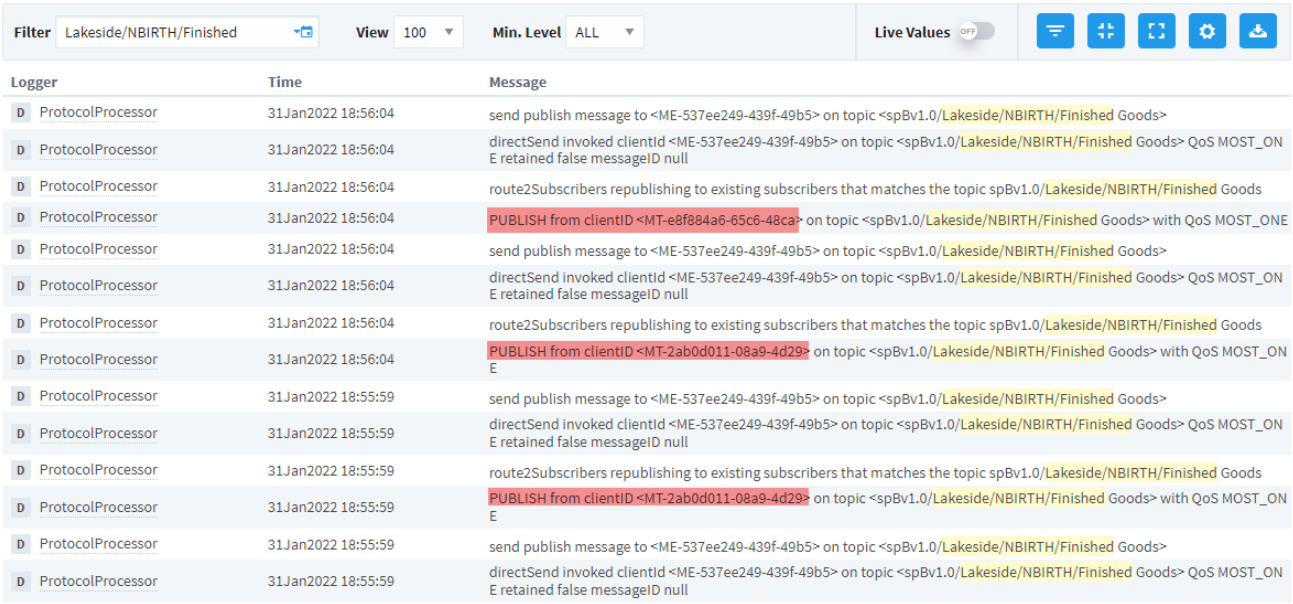


Expanding the *Failed to handle DDATA message* exposes the sequence number error we would expect in this scenario.

From the Ignition UI connected to your instance of MQTT Distributor, navigate to Status > Diagnostic > Logs.

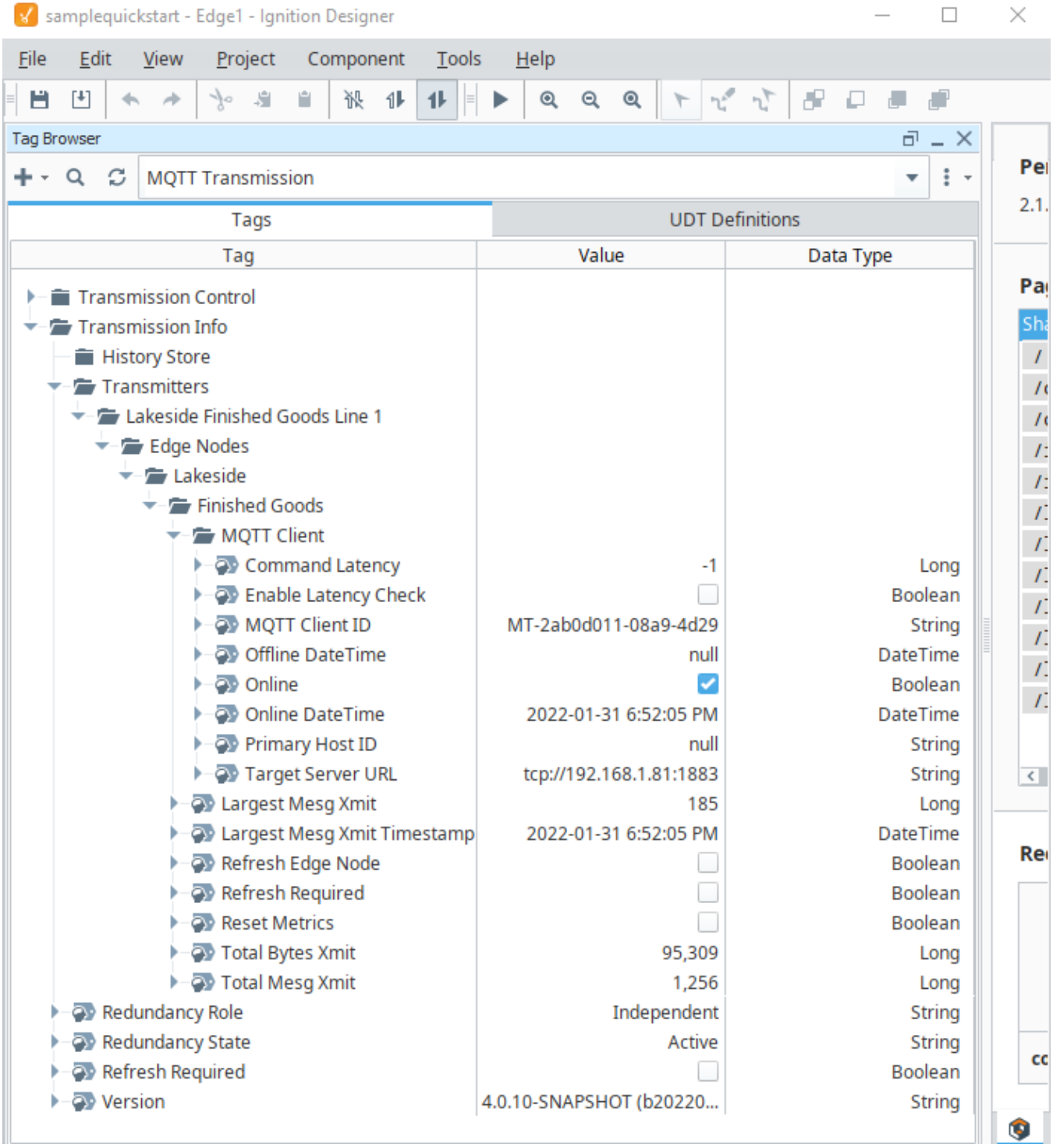
Set the debug level for the io.moquette.spi.impl.ProtocolProcessor logger to TRACE and in the filter search for the NBIRTH messages for the duplicate Edge Node ID. In this example the filter will be for Lakeside/NBIRTH/Finished Goods. Now we can see which Client IDs are responding to the rebirth requests from MQTT Engine.

The next step is to review and update as necessary the configuration for each of the listed Client IDs. If using MQTT Transmission, see here for how to identify the Client IDs from Ignition Designer.



Finding Client IDs using Ignition Designer

To identify the MQTT Client ID for each Edge Node in Designer, select the Tag Browser *MQTT Transmission* and expand the *Transmission Info* folder tree for each of your transmitters to expose the MQTT Client. ID.



Resolving Colliding Edge Nodes Descriptors

To resolve the colliding Edge Node Descriptors you will need to review your system configurations which generated each of the conflicting Edge Nodes Descriptors and remove the conflicts.

Refer to the MQTT Configuration guide and the MQTT Transmission Transmitters and Tag Trees Tutorial/HowTo for configuration help.

When troubleshooting remember that an update to any MQTT Transmission configuration parameter or a forced refresh of a transmitter, will mean that any dynamic Client IDs will be recreated. New logs will have reviewed after the update or refresh in order to correctly identify the problematic edge Nodes.

Unable to Resolve?

If the troubleshooting tips did not help you resolve your issues, please open a ticket with Support making sure to include the MQTT Engine or MQTT Distributor logs as appropriate.

From the Ignition Logs view, select the Download icon to download a copy of the system-name.idb file to your local file system. You will need to compress (zip, 7z or rar) this file before sending to Support.

Additional Resources

- Inductive Automation's Ignition download with free trial
  - Current Ignition Release
- Cirrus Link Solutions Modules for Ignition
  - Ignition Strategic Partner Modules
- Support questions
  - 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/