FAQ: Ignition Modules

MQTT Modules

General

- **Q:** What versions of the Cirrus Link modules are compatible with which versions of Ignition7?
  - **A:** See this page.
- **Q:** What do each of the MQTT Modules do in simple terms?
  - **A:** The MQTT modules for Ignition each have very specific functions. The very high level overview of each is listed below. For more detailed information see this page.
    - MQTT Distributor - A MQTT v3.1.1 compliant MQTT Server.
    - MQTT Engine - A module which acts as a *MQTT to Ignition Tag Bridge*. This module listens for and captures incoming MQTT messages and creates/updates Ignition tags based on those messages. In addition it listens for tag writes in Ignition and converts those to MQTT messages to send/update data and I/O on the remote MQTT devices. This module can be thought of as a tool to receive and visualize MQTT data in Ignition.
    - MQTT Transmission - An *Ignition Tag to MQTT Bridge.* This module listens for tag change events in Ignition and converts those to outgoing Sparkplug MQTT messages. In addition, it listens for incoming MQTT messages and updates tag values based on those incoming messages. This module can be thought of as a tool to MQTT enable Ignition Tag providers and push that data to an MQTT Server and MQTT Engine.

UDTs

- **Q:** What do I do if my UDT definition changes at the edge?
  - **A:** MQTT Engine will ignore any received UDT definitions that share the same name as an existing definition. In order to change or update a UDT definition within MQTT Engine the old definition must first be manually deleted.
- **Q:** If I delete and update a UDT definition in MQTT Engine will all of the existing instance's member Tags lose their custom properties and configurations (such as history)?
  - **A:** Yes, if a UDT definition is deleted and recreated, any existing UDT instances will lose any custom properties and/or configuration.

MQTT Distributor:

- Coming soon

MQTT Engine:

- **Q:** What is the Primary Host ID and why should I set it?
  - **A:** The 'Primary Host ID' is used for client state notifications to ensure MQTT clients are notified of a loss of connectivity to the primary host which is often MQTT Engine. For example, the 'Primary Host ID' must be set in order for the MQTT Distributor/Server to detect a disconnect between itself and the MQTT Engine and then publish a retained message for all connected clients stating that the MQTT Engine is offline. When MQTT Transmission sees this message stating the Engine/PrimaryHost is offline, it will then go offline itself and begin to store messages locally (assuming Store and Forward is enabled) to be sent once the Engine/PrimaryHost is back online.

MQTT Transmission:

- **Q:** What does the MQTT 'Transmission Control/Refresh' in the MQTT Transmission tag provider do?
  - **A:** This tag is used to 'refresh' the MQTT client(s) associated with MQTT Transmission. By default, any new tags added to a transmitter definition will not be published to the MQTT server. By writing to the 'Transmission Control/Refresh' tag any newly added tags will be detected and published.

- **Q:** What is the MQTT Client (Transmission) Keep Alive setting?
  - **A:** The Keep Alive setting controls how often the MQTT client (Transmission) is expected to ping the MQTT Server (Distributor) so the MQTT Server can verify the client is still connected. If the server hasn't seen a ping or control packet from the client in 1.5 x 'Keep Alive', it will forcefully close the socket connection and publish the LWT (which for a Sparkplug client is the NDEATH message) for that client denoting it as offline.

- **Q:** What is the Primary Host ID and why should I set it?
  - **A:** The 'Primary Host ID' is used for client state notifications to ensure MQTT clients are notified of a loss of connectivity to the primary host which is often MQTT Engine. For example, the 'Primary Host ID' must be set in order for the MQTT Distributor/Server to detect a disconnect between itself and the MQTT Engine and then publish a retained message for all connected clients stating that the MQTT Engine is offline. When MQTT Transmission sees this message stating the Engine/PrimaryHost is offline, it will then go offline itself and begin to store messages locally (assuming Store and Forward is enabled) to be sent once the Engine/PrimaryHost is back online.

Cloud Modules

Google Cloud Injector

- **Q:** Why am I seeing *"Connection Lost: Connection lost, attempting to reconnect"* errors coming from the **GoogleMqttIngestService** in my Ignition logs every ~15 minutes?

Example:

```
INFO   | jvm 1    | 2019/02/06 00:30:33 | E [c.c.i.g.c.GoogleMqttIngestService] [00:30:33]: Connection Lost:
Connection lost, attempting to reconnect
INFO   | jvm 1    | 2019/02/06 00:30:33 | org.eclipse.paho.client.mqttv3.MqttException: Connection lost
INFO   | jvm 1    | 2019/02/06 00:30:33 |     at org.eclipse.paho.client.mqttv3.internal.CommsReceiver.run
(CommsReceiver.java:164)
INFO   | jvm 1    | 2019/02/06 00:30:33 |     at java.lang.Thread.run(Thread.java:748)
INFO   | jvm 1    | 2019/02/06 00:30:33 | Caused by: java.io.EOFException: null
INFO   | jvm 1    | 2019/02/06 00:30:33 |     at java.io.DataInputStream.readByte(DataInputStream.java:267)
INFO   | jvm 1    | 2019/02/06 00:30:33 |     at org.eclipse.paho.client.mqttv3.internal.wire.MqttInputStream.
readMqttWireMessage(MqttInputStream.java:92)
INFO   | jvm 1    | 2019/02/06 00:30:33 |     at org.eclipse.paho.client.mqttv3.internal.CommsReceiver.run
(CommsReceiver.java:116)
INFO   | jvm 1    | 2019/02/06 00:30:33 |     ... 1 common frames omitted
```

- **A:** Google automatically closes our injector connection after 15 minutes. This is ***expected behavior***. We automatically reestablish the connection to the Google Cloud and resume pushing messages.