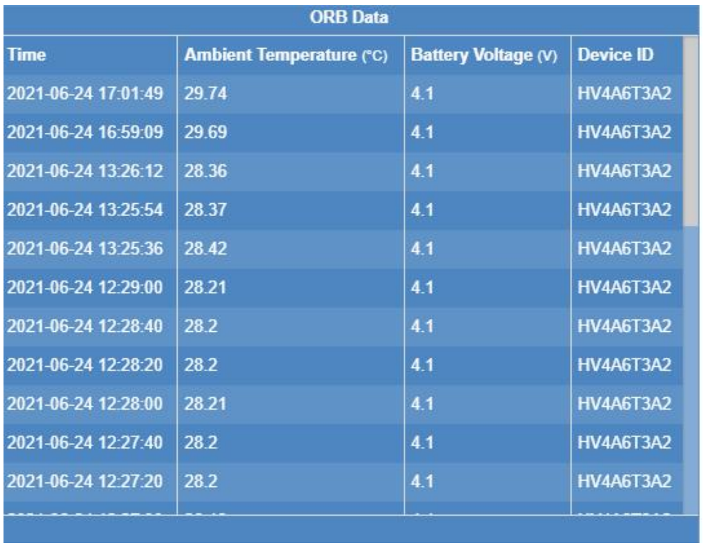
# INTEGRATING THE ORB WITH EAGLE.IO VIA MQTT

## 1. Introduction

This Application Note details how to integrate the ORB with Eagle.io using MQTT communications. MQTT stands for Message Queuing Telemetry Transport and is a lightweight, publish-subscribe network protocol that transports messages between devices.

This document will outline how to integrate an ORB-X1 with Eagle.io by publishing a JSON Time Series over MQTT. In this application note, we will assume that you have access to the scripting feature, that the ORB is subscribed to a Premium plan, and that you are familiar with the scripting environment on the Senquip Portal. For further details on scripting on the Senquip ORB, please see the scripting guide: https://docs.senquip.com/scripting_guide.



ORB Data

| Time | Ambient Temperature (°C) | Battery Voltage (V) | Device ID |
|---|---|---|---|
| 2021-06-24 17:01:49 | 29.74 | 4.1 | HV4A6T3A2 |
| 2021-06-24 16:59:09 | 29.69 | 4.1 | HV4A6T3A2 |
| 2021-06-24 13:26:12 | 28.36 | 4.1 | HV4A6T3A2 |
| 2021-06-24 13:25:54 | 28.37 | 4.1 | HV4A6T3A2 |
| 2021-06-24 13:25:36 | 28.42 | 4.1 | HV4A6T3A2 |
| 2021-06-24 12:29:00 | 28.21 | 4.1 | HV4A6T3A2 |
| 2021-06-24 12:28:40 | 28.2 | 4.1 | HV4A6T3A2 |
| 2021-06-24 12:28:20 | 28.2 | 4.1 | HV4A6T3A2 |
| 2021-06-24 12:28:00 | 28.21 | 4.1 | HV4A6T3A2 |
| 2021-06-24 12:27:40 | 28.2 | 4.1 | HV4A6T3A2 |
| 2021-06-24 12:27:20 | 28.2 | 4.1 | HV4A6T3A2 |

## 2. Requirements

The following are required for integration of a Senquip device into an existing Eagle.io workspace:

- Access to an ORB with Premium Portal access and scripting enabled.
- Access to an Eagle.io workspace

**Automation Group**
Sydney • Newcastle • Brisbane • Melbourne • Perth
**P** 1300 724 743 **E** sales@automationgroup.com.au **W** automationgroup.com.au

## 3. Eagle.io Configuration

This section outlines how to set up a Data Source in Eagle.io to receive data from an ORB.

1. Create a new Data Source in your workspace

   Data Sources automatically acquire or receive timeseries data using a variety of different transport options. Data Sources can be created inside Locations only. The type of Data Source is selected at time of creation and cannot be changed.

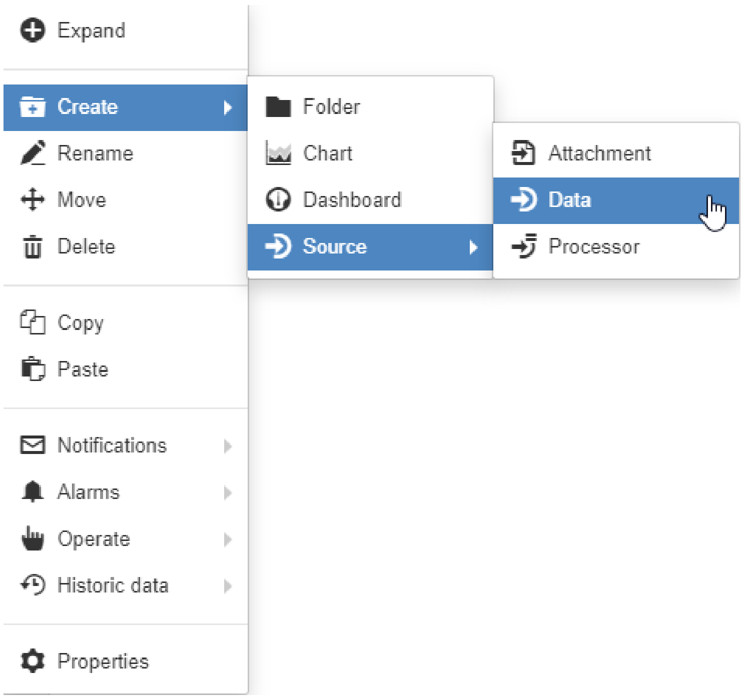
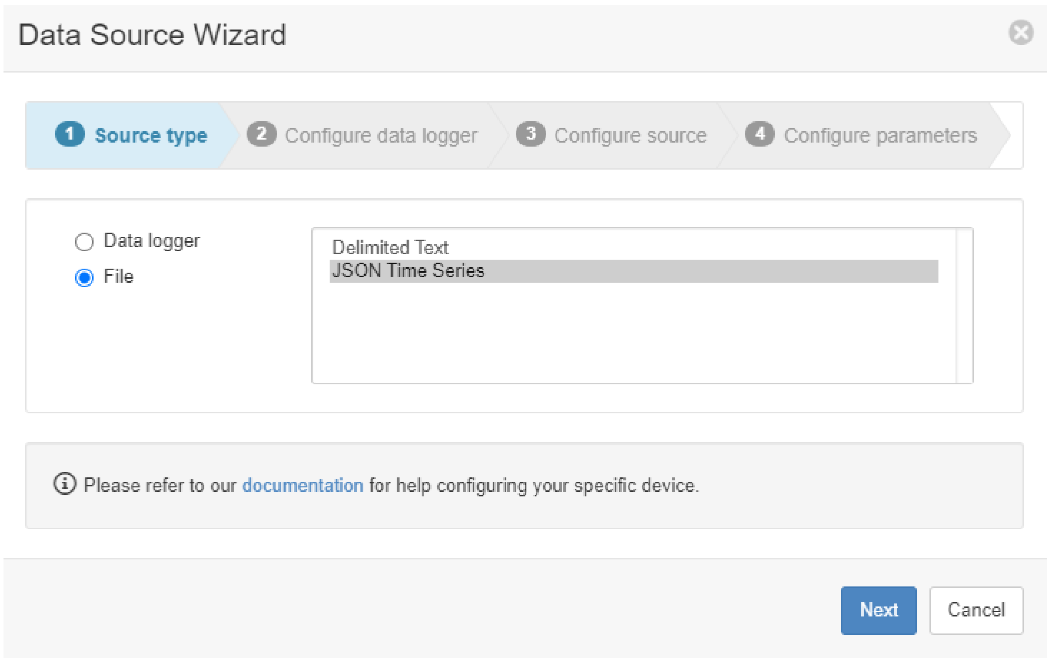


*Figure 1 - Creating a Data Source*

2. Choose File -> JSON Time Series for the Source Type.

   JSON Time Series (JTS) is a lightweight data-interchange format for time series data. It has been designed to be highly readable, parsable and extendable.

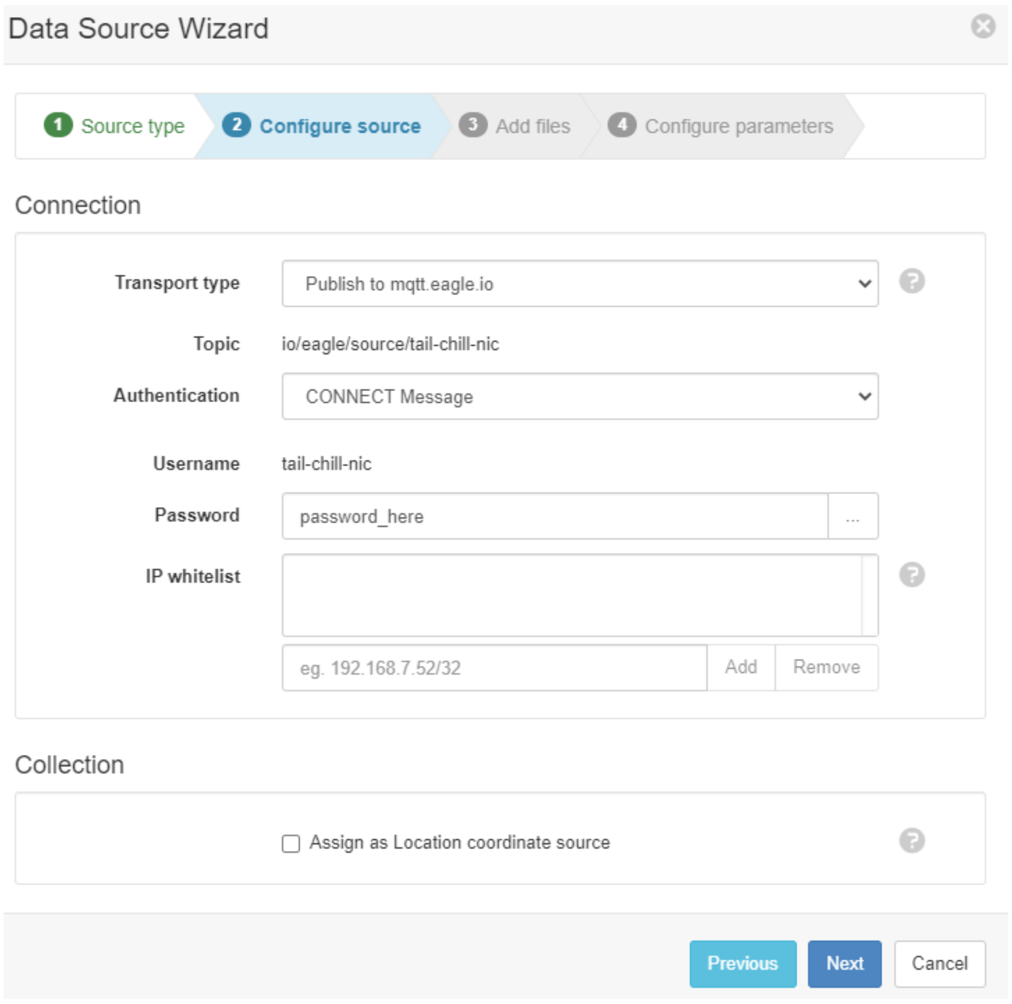


Please refer to our documentation for help configuring your specific device.

*Figure 2 - Choosing JSON as the Data Source*

3. For the source configuration, set the transport type to “publish to mqtt.eagle.io” and the authentication to “CONNECT Message”. Set a password for the ORB to authenticate with.

| Parameter | Date |
|---|---|
| Broker Address | mqtt.eagle.io |
| Broker Port | Use port 1883 for standard connection |
| Topic | Use the auto-generated topic exactly as shown - io/eagle/source/tail-chill-nic |
| MQTT Password | Optional password (leave blank for none) |
| IP Whitelist | You can optionally restrict incoming connections to this source to a list of approved IP addresses specified |



Connection

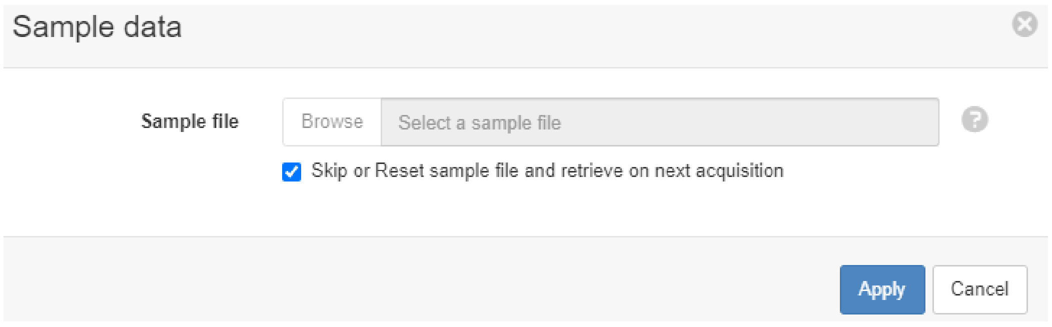
| | |
|---|---|
| Transport type | Publish to mqtt.eagle.io |
| Topic | io/eagle/source/tail-chill-nic |
| Authentication | CONNECT Message |
| Username | tail-chill-nic |
| Password | password_here |
| IP whitelist | eg. 192.168.7.52/32 |

Collection

☐ Assign as Location coordinate source

*Figure 3 - Choosing the Transport Type*

4. Choose to skip providing sample data and click apply.



Sample file: Browse | Select a sample file

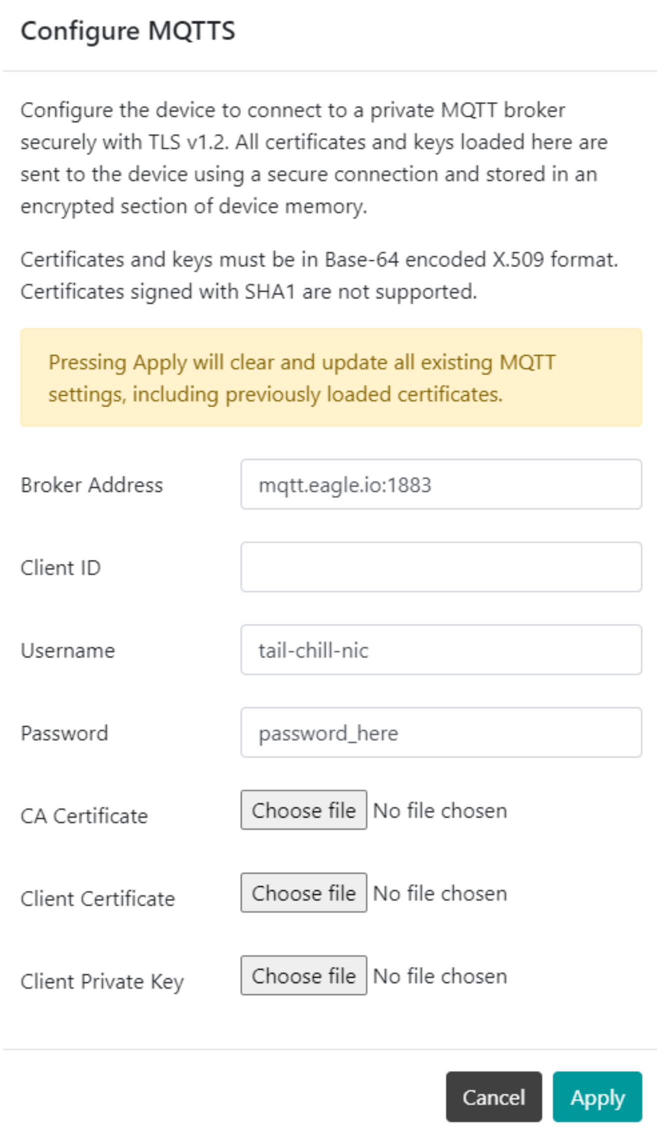
☑ Skip or Reset sample file and retrieve on next acquisition

## 4. ORB Configuration

The ORB needs to be configured to send data to Eagle.io. This section will deal with endpoint settings.

To configure an ORB to send data to Eagle.io, complete the following steps.

1. Browse to http://portal.senquip.com, log in, and select the ORB you wish to connect to Eagle.io.
2. Go to Settings > General, and confirm that your device is using firmware version *SF001-3.1.0* or newer.
3. Go to Setting > Endpoint and untick *Use Senquip Data Format*. This tells the ORB not to use the standard Senquip JSON format and that the data packet will be created from within a script.
4. In the MQTT settings, enter *mqtt.eagle.io:1883* for the broker address along with your username (tail-chill-nic ) and password previously configured when setting up the Eagle.io connection.



Configure MQTTS

Configure the device to connect to a private MQTT broker securely with TLS v1.2. All certificates and keys loaded here are sent to the device using a secure connection and stored in an encrypted section of device memory.

Certificates and keys must be in Base-64 encoded X.509 format. Certificates signed with SHA1 are not supported.

Pressing Apply will clear and update all existing MQTT settings, including previously loaded certificates.

| | |
|---|---|
| Broker Address | mqtt.eagle.io:1883 |
| Client ID | |
| Username | tail-chill-nic |
| Password | password_here |
| CA Certificate | Choose file No file chosen |
| Client Certificate | Choose file No file chosen |
| Client Private Key | Choose file No file chosen |

*Figure 4 - Configuring the ORB Endpoint*

## 5. Crafting the MQTT Payload

Eagle.io requires data sent over MQTT to be in the Eagle.io JSON Time Series format. The ORB allows the user to create custom MQTT payloads and publish them using the MQTT.pub function, we will use this feature to send data in a JSON Time Series format to Eagle.io.

It will be assumed that you are familiar with the ORB scripting functionality. For more information on scripting for Senquip devices, please see: http://docs.senquip.com/scripting_guide/. For more information on the Eagle.io JSON Time Series format please refer to https://docs.eagle.io/en/latest/reference/historic/jts.html.

An example script is provided in the appendix. This script creates a JSON Time Series with the ambient temperature, system voltage, and device ID. As a starting point it is recommended to use this script to confirm the connection to Eagle.io works as expected. All that is required is to change the MQTT publishing topic, which is the first argument of MQTT.pub, to the MQTT topic provided by Eagle.io when you configured the data source.

To send a new value to Eagle.io, you will need to add a header column for the variable and a new entry to the "f" field of the data array. To send a number set the *datatype* to VALUE, for a string set the *datatype* to *TEXT*. Note that the payload structure is very strict, a comma out of place will result in the data being ignored by Eagle.io. As such it is strongly recommended to make changes incrementally and confirm that the payload is still valid.

## 6. Conclusion

Configuring a Senquip ORB to send data to Eagle.io is simple using the ORB MQTT endpoint settings and a script.

If you have any queries about the procedures in this document or would likely to know more about Senquip devices, please contact Automation Group Support at support@automationgroup.com.au or call 1300 724 743 and select Option 1.

## Appendix 1: Source Code

```
load('senquip.js');
load('api_config.js');
load('api_endpoint.js');
load('api_timer.js');

SQ.set_data_handler(function(data) {
  let obj = JSON.parse(data);
  let now = Timer.now();
  let fsnow = Timer.fmt("%FT%T.000Z", now);  // format time is Eagle.io format
  let device_id = Cfg.get('device.id');
 MQTT.pub("io/eagle/source/tail-chill-nic", // lots of warnings due to nested quotes
    "{
    \"docType\": \"jts\",
    \"version\": \"1.0\",
    \"header\": {
        \"recordCount\": 1,
        \"columns\": {
            \"0\": {
                \"id\": \"0001\",
                \"name\": \"Ambient Temperature\",
                \"dataType\": \"NUMBER\",
                \"renderType\": \"VALUE\",
                \"format\": \"0.###\",
                \"aggregate\": \"NONE\"
            },
             \"1\": {
                \"id\": \"0002\",
                \"name\": \"Battery Voltage\",
                \"dataType\": \"NUMBER\",
                \"renderType\": \"VALUE\",
                \"format\": \"0.###\",
                \"aggregate\": \"NONE\"
            },
             \"2\": {
                \"id\": \"0003\",
                \"name\": \"Device ID\",
                \"dataType\": \"TEXT\",
                \"renderType\": \"STATE\"
            }
        }
    },
    \"data\": [
        {
            \"ts\": \"" +fsnow +"\",
            \"f\": { \"0\": {\"v\": " +JSON.stringify(obj.ambient) +"},
                     \"1\": {\"v\": " +JSON.stringify(obj.vsys) +"},
                     \"2\": {\"v\": \"" + device_id +"\"}
            }
        }
    ]
   }"
  );
}, null);
```