CHIPKIN

Sparkplug Driver
FS-8705-114

Chipkin - Enabling Integration

salesgroup1@chipkin.com

Tel: +1 866 383 1657

Driver Version:
Document Revision: 3

TABLE OF CONTENTS

1 Sparkplug Description

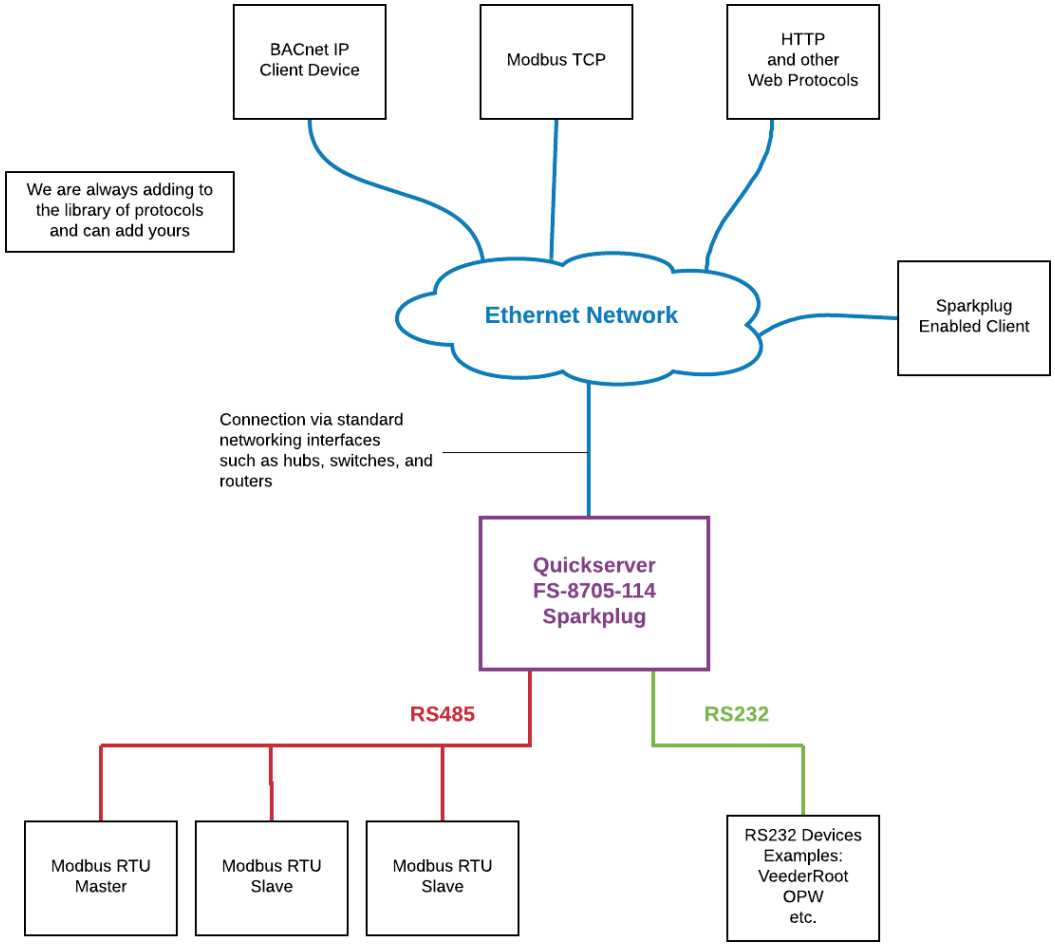
The Sparkplug Driver allows the FieldServer to publish data received from downstream devices to a Sparkplug enabled server. The Sparkplug protocol uses MQTT with a specific set of topics and payloads. Data that the FieldServer receives downstream devices can be configured as EdgeNode metrics or Device metrics. The Sparkplug Driver supports birth (NBIRTH and DBIRTH), data (NDATA and DDATA), and death (NDEATH) messages, as well as command messages (NCMD and DCMD).

The FieldServer is a Sparkplug node pushing requests to a configured URL endpoint. The FieldServer stores values to be mapped to other protocols or simply to be viewed. When configured, the FieldServer on bootup will send NBIRTH and DBIRTH data based on the configuration. As data is read and stored into the FieldServer data arrays, the driver will send NDATA and DDATA to the Sparkplug server. A keyframe of data is sent for all mapped data points based on the configured interval. If the Sparkplug server sends a command to the FieldServer, the FieldServer will store the commanded value and send it to the respective downstream device attempting to write or change the value.

The information that follows describes how to expand upon the factory defaults provided in the configuration files included with the FieldServer.

2 Connection Description

This block diagram shows data being served using other protocols like Modbus® RTU/TCP, and BACnet®. The FieldServer can use the Sparkplug Driver to publish this data to a Sparkplug Server.



3 Sparkplug Driver Configuration

To configure the Sparkplug driver, from the home page, visit the following link:
http://{IP_ADDRESS}/chipkin/ui/#/sparkplugDriver.

To configure the FieldServer, follow the instructions below to add a Connection (physical port), Nodes (Sparkplug Edge Node), and finally Tasks (Metrics for the Edge Node or DeviceId).

3.1 Create Connection

To set up the FieldServer Sparkplug Driver, first create a connection. The connection contains information about the physical port to use.

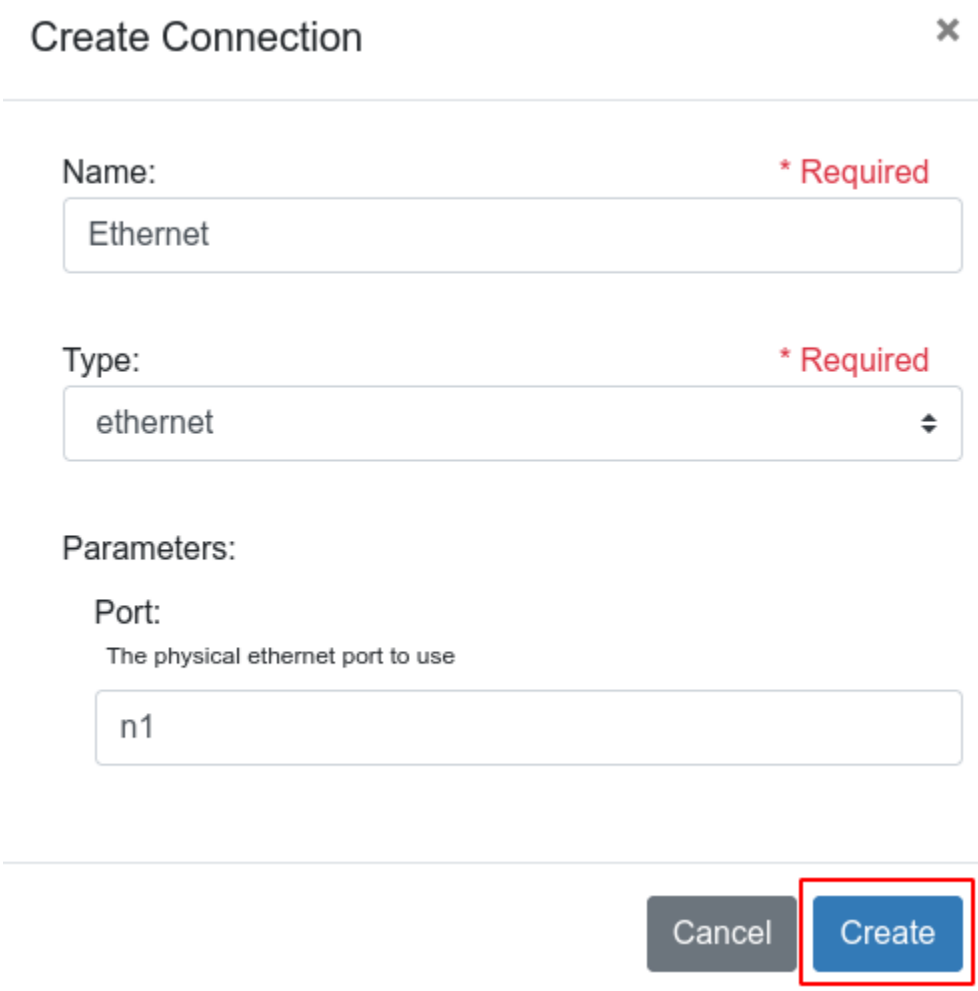
1. Click on the "Create Connection" button to open the Create Connection form.

2. The fields are as follows:

COLUMN TITLE	FUNCTION	LEGAL VALUES
Name	Name of the server, used internally as an identifier for Nodes.	Text, must be unique
Type	The type of connection this is. Currently, only ethernet is supported.	**ethernet**
Parameters: Port	The physical port on the FieldServer to use	**n1**
Port		

* Bolded values are defaults

3. Click the "Save" button to add the connection.

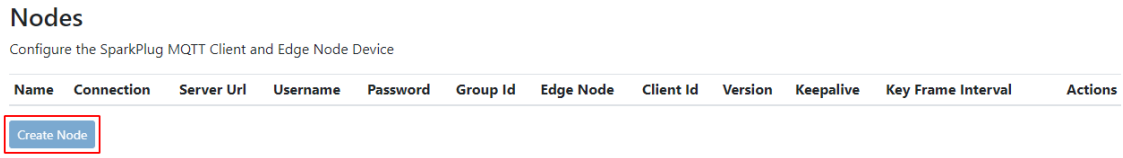


If successful, the new entry will be populated in the Connections table:

***Note*:** Only one server connection can exist. If multiple connections are created, only the first one will be used.

3.2 Create Node

Follow the instructions below to configure the SparkPlug MQTT Client and Edge Node Device.



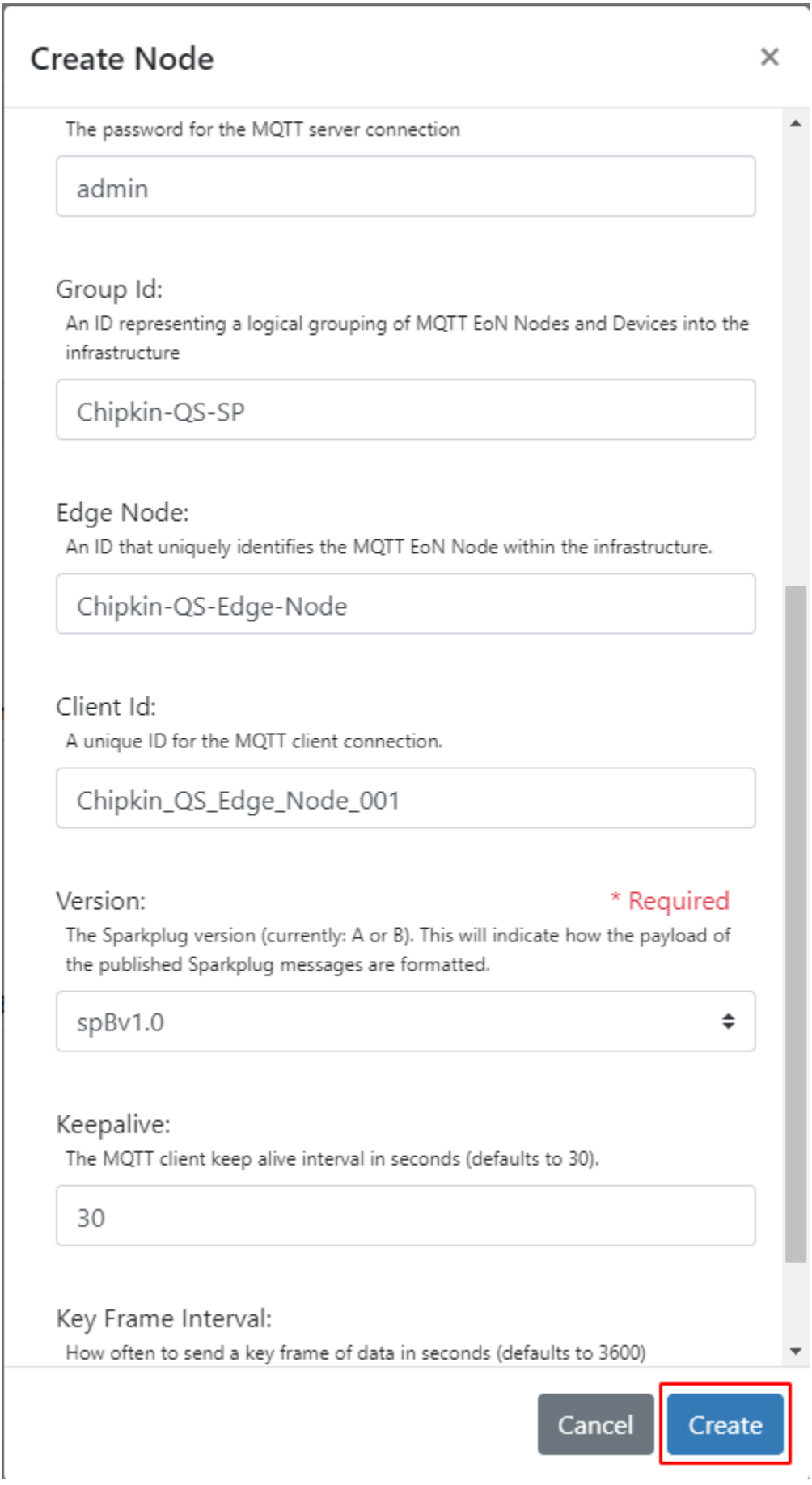
1. Click on the "Create Node" button to open the Create Node form.
2. Fill out the fields in the form. The fields are as follows:

COLUMN TITLE	FUNCTION	LEGAL VALUES
Name	A name given to this Node. A Task will reference this Node by this Name. Must be unique	Text (string)
Connection	The name of the FieldServer's physical port, linked via the Connection.	Text (Use the name of the Connection created in the previous section)
ServerUrl	The url of the MQTT server to connect to	Url string. Example: tcp://192.168.1.77:1883
Username	The username for the MQTT server connection if authorization is required	Text (string)
Password	The password for the MQTT server connection	Text (string)
GroupId	An ID representing a logical grouping of MQTT EoN Nodes and Devices into the infrastructure	Text (string) Default: **Chipkin-QS-SP**
EdgeNode	An ID that uniquely identifies the MQTT EoN Node within the infrastructure	Text (string) Default: **Chipkin-QS-Edge-Node**
ClientId	A unique ID for the MQTT client connection	Text (string) Default: **Chipkin_QS_Edge_Node_001**
Version	The Sparkplug version (currently: A or B). This will indicate how the payload of the published Sparkplug messages are formatted	spAv1.0 or **spBv1.0**
Keepalive	The MQTT client keep alive interval in seconds (defaults to 30)	0 to 3600, **30**

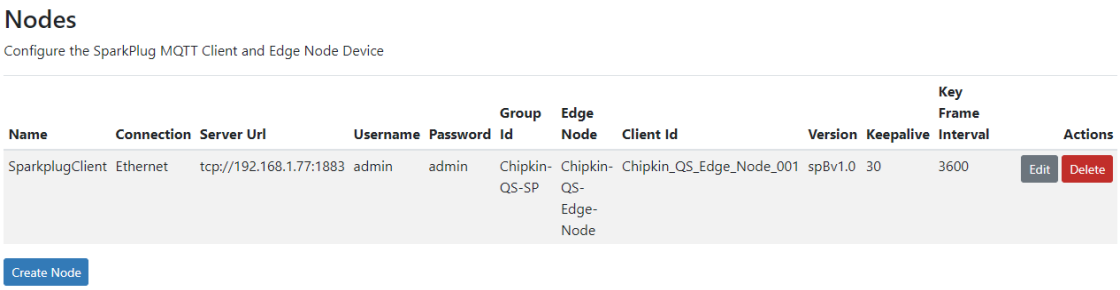
KeyFrameInterval	How often to send a key frame of data in seconds (defaults to 3600)	0 to MaxUINT32, **3600**

*Bolded values are defaults

3. Click on the “Create” button to add the node.



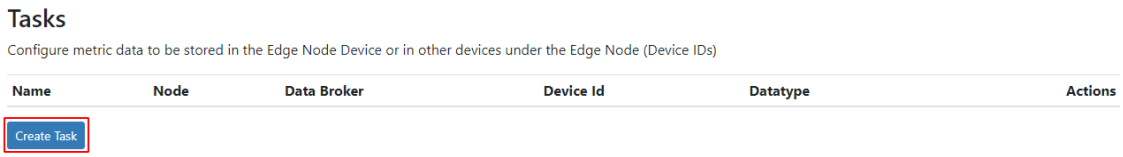
If successful, the new entry will be populated in the Nodes table:



3.3 Create Task

Create tasks in order to add metrics to either the Edge Node Device or in other devices under the Edge Node (Device IDs).

1. Click on the "Create Task" button to open the Create Task form.



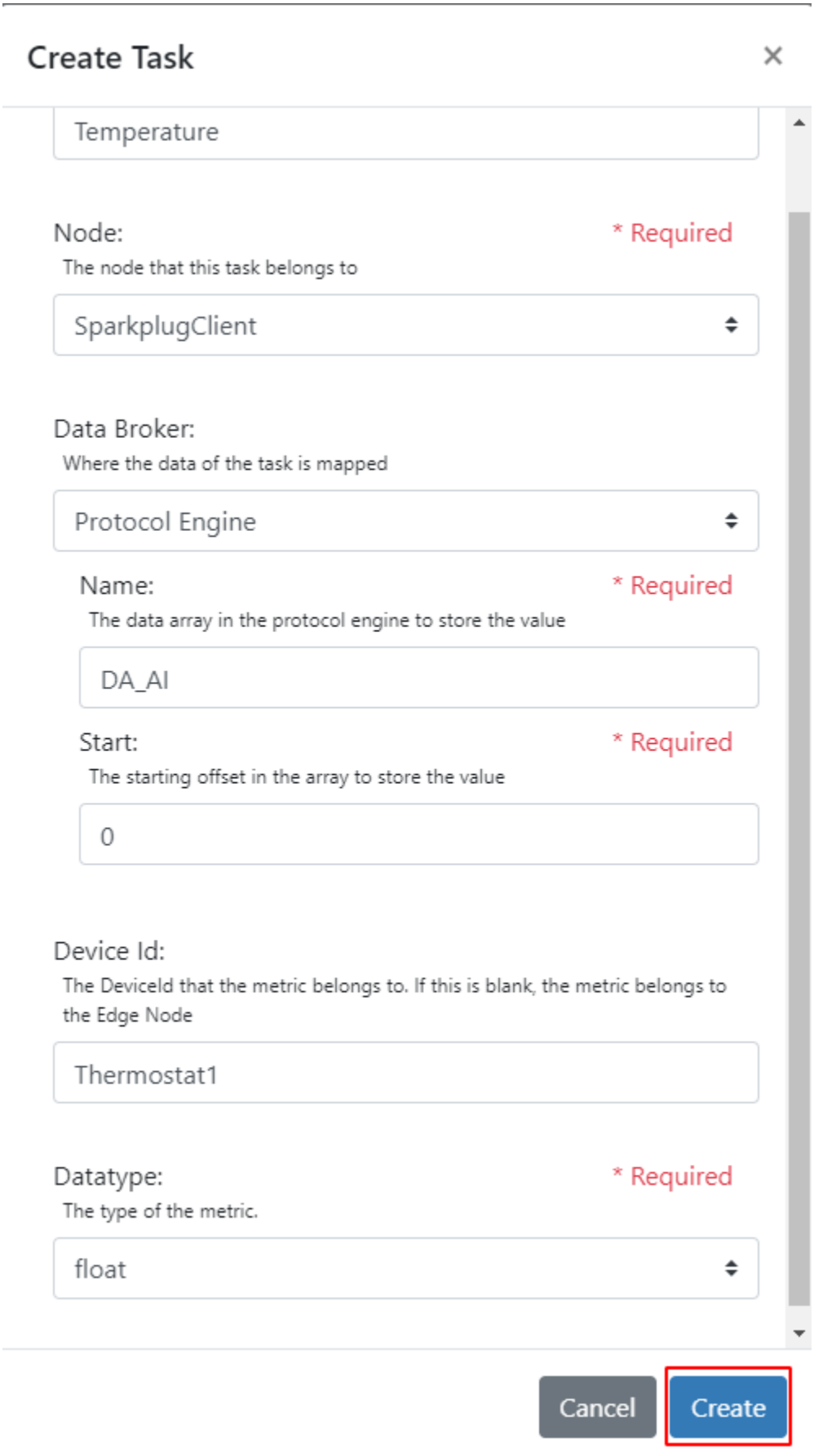
2. Fill out the fields in the form. The fields are as follows:

COLUMN TITLE	FUNCTION	LEGAL VALUES
Name	The name of the metric	Text (string)
Node	The Node that this task will reference when making requests.	Text (string)
Data Broker	The FieldServer data array map used to store data	**Protocol Engine** - The driver level data array map, configurable from the config.csv Application Engine - The in memory data object.
Data Broker PE:Name	The name of the data array to map values to.	Text (string) - `DA_AI`
Data Broker PE:Start	The offset within the data array to map values to.	Integer - `0`

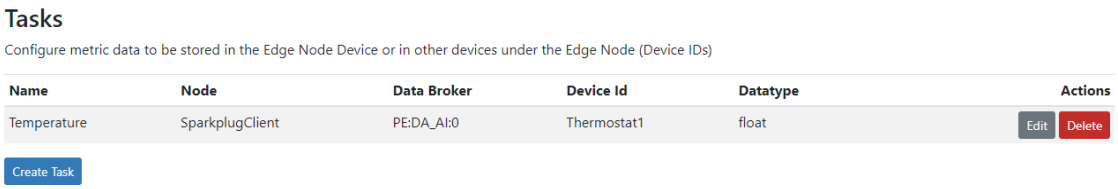
Data Broker AE:Path	The path into the AE data store where the values are mapped.	Text (string) - `example/path/to/location`
DeviceId	The DeviceId that the metric belongs to. If this is blank, the metric belongs to the Edge Node	Text (string)
Datatype	The type of the metric	int, int8, int16, int32, int64, uint8, uint16, uint32, uint64, **float**, double, boolean

*Bolded values are defaults

3. Click the “Create” button to add the node.



If successful, the new entry will be populated in the Tasks table:



3.4 Saving the Server Configuration

When the configuration is complete, click on the “Save Configuration” button to save all of the updates and changes. For the configuration to take effect, reboot the system.

3.5 Resetting the Server Configuration

To clear the configuration and start over, click the “Reset Configuration” button. Then follow the instructions in the sections above to create new connections, nodes, and tasks.

4 Sparkplug Examples

The Sparkplug Driver does the following actions:

4.1 Birth Messages

On start-up, the Sparkplug Driver will publish NBIRTH and DBIRTH Sparkplug messages based on the configuration. These messages contain all the data points that have been configured for the Edge Node as well as any devices under the Edge Node.

4.2 Data Messages

The Sparkplug Driver will send various NDATA and DDATA Sparkplug messages throughout driver operation. As data points are updated in the data arrays from the downstream devices, an NDATA or DDATA message with the updated value is published to the configured Sparkplug Server.

Also, on the configured KeyFrameInterval, the driver will generate an NDATA and DDATA message for all of the data points configured in the Edge Node and devices, similar to what is sent in the NBIRTH/DBIRTH message, just with the current values.

4.3 Death Messages

If the FieldServer ever goes offline and was able to shutdown properly, a NDEATH message will be sent to the Sparkplug Server to notify that the FieldServer Edge Node device has gone offline.

4.4 Command Messages

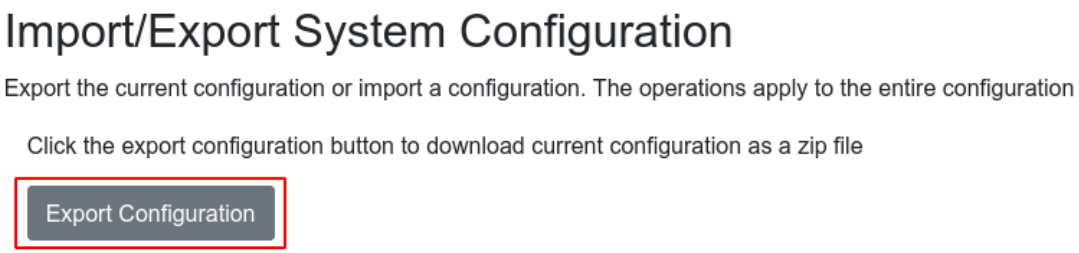
If the Sparkplug Driver receives either a Edge Node Command (NCMD) or a Device Command (DCMD), it will process the command by finding the matching data point and updating the value in its data array. This will trigger a write-through to the downstream device attempting to update the value. If successful, the updated value will be published back to the Sparkplug Server via a NDATA or DDATA message.

5 Importing and Exporting Configurations

It is possible to export the current configuration to back it up or simply to make some edits. Users can also import either the entire configuration via a zip file or a PE (Protocol Engine) configuration.

5.1 How to export the Configuration

1. Goto the system configuration page http://{IP_ADDRESS}/chipkin/ui/#/chipkinLicenseDriver
2. Click the Export Configuration button.



5.2 How to Import the Configuration

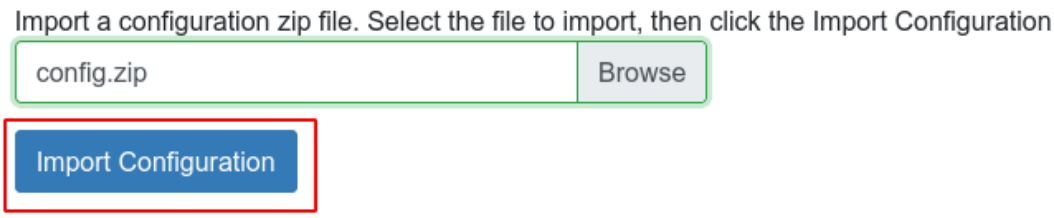
The file to import the configuration must be a zip file. The zip file should contain the following folders:

- ae - this folder contains any configuration files for the ae configuration
- documents - this folder contains any driver specific documents. For example, license product keys, etc.
- pe - this folder contains one config.csv file for the pe configuration.

To make sure the folder directory is correct, do an Export first, then extract the files, edit them, then zip them up again.

To import the configuration:

1. Goto the system configuration page http://{IP_ADDRESS}/chipkin/ui/#/chipkinConfiguration
2. Click the “Browse” button in the “Import/Export System Configuration” section and select the zip file containing the configuration to import.
3. Click the “Import Configuration” button and wait for the configuration to finish importing.
4. If successful, a success message will appear prompting a reboot of the FieldServer for the changes to take effect.



5.3 Example PE Configuration

```
Bridge
Title, System_Node_id
Example Thermostat, 389001

Data_Arrays
Data_Array_Name , Data_Format    , Data_Array_Length
DA_AI           , float          , 200
DA_AV           , float          , 200

//========================================================================================
//
//    Client Side
//

Connections
Adapter ,Protocol, Poll_Delay
N1      ,Bacnet_IP, 0.100s

Nodes
Node_name ,Node_ID ,Protocol, Adapter
vSensor01 ,1       ,Bacnet_IP, N1

Map_Descriptors
Map_Descriptor_Name , Data_Array_Name , Data_Array_Offset , Function , Node_Name ,
Object_Type , Object_Instance , Property    , Scan_Interval
Temperature    , DA_AI ,0, RDBC  , vSensor01    , AI  ,1, Present_Value , 1.0s
SetPoint       , DA_AV ,0, RDBC  , vSensor01    , AV  ,2, Present_Value , 1.0s
```

5.4 Example AE Configuration

```
{
  "ae": {
    "sparkplugDriver": {
      "connections": [
        {
          "type": "ethernet",
          "name": "Ethernet",
          "parameters": {
            "port": "n1"
          }
        }
      ],
      "nodes": [
        {
          "connection": "Ethernet",
          "version": "spBv1.0",
```

```
        "name": "IgnitionSystem",
        "serverUrl": "tcp://192.168.1.77:1883",
        "groupId": "Chipkin-QS-SP",
        "edgeNode": "Chipkin-QS-Edge-Node",
        "clientId": "Chipkin_QS_Edge_Node_001",
        "keepalive": 60,
        "keyFrameInterval": 3600,
        "username": "admin",
        "password": "admin"
      }
    ],
    "tasks": [
      {
        "node": "IgnitionSystem",
        "datatype": "float",
        "name": "Temperature",
        "deviceId": "Thermostat1",
        "dataBroker": {
          "pe": {
            "Name": "DA_AI",
            "Start": "0"
          }
        }
      },
      {
        "node": "IgnitionSystem",
        "datatype": "float",
        "name": "SetPoint",
        "deviceId": "Thermostat1",
        "dataBroker": {
          "pe": {
            "Name": "DA_AV",
            "Start": "0"
          }
        }
      }
    ]
  }
 }
}
```

6 Marketing

6.1 Case Study

n/a

6.2 Keywords

HTTP, HTTP, PE, AE, Sparkplug, MQTT

6.3 Glossary of Terms

- HTTP(s) – Hypertext Transfer Protocol (secure)
- PE – Protocol Engine
- AE – Application Engine

7 Revision History

This table summarizes the update history for this document. Please contact Chipkin for an updated version of this document if required.

DATE	RESP	DOC. REV.	COMMENT
6 Oct 2021	ACF	1	Created document
19 Oct 2021	YC	2	Updated document format
21 Jun 2024	ACF	3	Updated Example configurations