SMC Cloud RESTful API

Start-up Guide

APPLICABILITY & EFFECTIVITY

Explains SMC Cloud RESTful API.

The instructions are effective for the above as of January 2020.

Document Revision: 1.G
T18663

Technical Support

Please call us for any technical support needs related to the FieldServer product.

MSA Safety
1991 Tarob Court
Milpitas, CA 95035

Website: www.sierramonitor.com

U.S. Support Information:

+1 408 964-4443
+1 800 727-4377

Email: smc-support@msasafety.com

EMEA Support Information:

+31 33 808 0590

Email: smc-support.emea@msasafety.com

Assumptions

The following items should be complete before moving forward with SMC Cloud RESTful API setup.

- ✓ Install preferred RESTful client program.
- ✓ Setup a SMC Cloud account.
- ✓ Login to a computer with access to the internet and web browser.

Quick Start Guide

1. Open the preferred RESTful client.
2. Authenticate with www.FieldPoP.io/rest/login.
3. Test the API.
    a. Set up a sample
    b. Run the sample
4. Run GET device data log commands.

TABLE OF CONTENTS

1 Overview 6

2 SMC Cloud Restful API Supported Functions 7

2.1 HTTP Usage 7
2.2 Authentication 7
2.2.1 Login 7
2.3 Data Read/Write 8
2.3.1 Data Path Parameter 8
2.3.2 All Field Device Types 9
2.3.2.1 DeviceDataLog 9
2.3.3 Device Type: System View 10
2.3.3.1 GetDeviceData (System View v.2, v.3) 10
2.3.3.2 GetDeviceData (System View v.4) 13
2.3.3.3 SetDeviceData (System View) 18
2.3.4 Device Type: BACnet Explorer NG 19
2.3.4.1 GetDeviceData (BACnet Explorer NG) 19
2.3.4.2 SetDeviceData (BACnet Explorer NG) 20
2.3.5 Device Type: BACnet IoT Gateway 21
2.3.5.1 GetDeviceData (v.1, v.2) 21
2.3.5.2 CallDeviceMethod – GetBACnetDeviceInfo (v.1) 28
2.3.5.3 CallDeviceMethod – GetBACnetDeviceInfo (v.2) 29
2.3.5.4 CallDeviceMethod – GetBACnetPropertyValue (v.1, v.2) 30
2.3.5.5 CallDeviceMethod – SetBACnetPropertyValue (v.1, v.2) 31
2.4 User Methods 32
2.4.1 GetUserInfo 32
2.4.2 GetUserDevices 32
2.4.3 ListDeviceUsers 35
2.5 ProtoCast Methods 36
2.5.1 CreateOEMUser 36
2.5.2 GetFieldCastOEMList 36
2.5.3 GetDeviceData 37
2.5.4 GetDeviceDataWithTime 38
2.6 Firmware Methods 39
2.6.1 GetDeviceFirmwareVersions 39
2.6.2 UpgradeFirmware 40

Appendix A. Additional Information 41

Appendix A.1. General Notes 41
Appendix A.1.1. How to Obtain an IIoT Device ID (DeviceID Parameter Value) 41
Appendix A.1.2. Device API Version 41
Appendix A.1.3. Data Retention 41
Appendix A.1.4. Data Structure 41
Appendix A.2. Using CURL to Generate SMC Cloud RESTful API Requests 42
Appendix A.3. How to Access the RESTful API Through Fiddler 43
Appendix A.4. Generic Information for Access to SMC Cloud RESTful API 46
Appendix A.4.1. Authenticate via GET Command 46
Appendix A.4.2. Authenticate via POST Command 46
Appendix A.4.3. Run GET Command to Deliver Device Data Logs 46

Appendix B. Troubleshooting 47

Appendix B.1. Lost SMC Cloud Login Password 47

Appendix C. Useful Features 48

Appendix C.1. Security 48
Appendix C.1.1. PC to SMC Cloud 48
Appendix C.1.2. FieldServer to SMC Cloud 48
Appendix C.1.3. Viewing the Certificate 48

Appendix D. Warranty 50

1 OVERVIEW

Setting up the SMC Cloud as a RESTful API allows SMC Cloud users to automate the sending of recorded device data (gathered from the SMC Cloud) to the cloud.

NOTE: For general SMC Cloud registration and use instructions, see the SMC Cloud Start-up Guide.

The SMC Cloud RESTful API acts as a RESTful server using HTTPS. The API will support authentication, read operations, and write operations, including:

- Read (HTTP GET)
  - data logs recorded on the SMC Cloud
  - event logs recorded on the SMC Cloud
  - current data values exposed to the SMC Cloud by devices
- Write (HTTP PUT)
  - writable values (e.g. setpoints) exposed to the SMC Cloud by devices

In a Javascript environment, a Happner client would be the suggested access method.

NOTE: All URLs are given relative to https://www.fieldpop.io/.

2 SMC CLOUD RESTFUL API SUPPORTED FUNCTIONS

2.1 HTTP Usage

Most RESTful requests can be made using either POST or GET requests. POST requests must specify parameters in a JSON payload and GET requests must specify parameters as query parameters in the URL. Parameters that are too complex for the query parameter syntax will necessitate a JSON POST request.

All POST requests containing a JSON payload must set the following HTTP header:

Content-Type: application/json

2.2 Authentication

A RESTful client acquires an API token using the Authenticate RESTful call. This token must accompany every RESTful request as a query parameter in order to authenticate the request. Authentication tokens inherit the permissions of the user who created them.

Important Note:

All GET and POST requests other than Authentication requests must provide the authentication token as a query parameter, happn_token:
/rest/method/fieldpop-api/deviceDataLog?happn_token=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.ImlkfGNiMmQ0NDg5LTAyNTQtNGJlYy1hZWNiLThkZDM3NDY0NjA3Mnx0eXBlfHRpbWVzdGFtcHxpc0VuY3J5cHRlZHxvcmlnaW58ZmllbGRwb3AtbWFpbnxwb2xpY3l8MHx0dGx8aW5hY3Rpdml0eV90aHJlc2hvbGR8MXxwZXJtaXNzaW9uU2V0S2V5fC9PRU0rQWRtaW4vL19NRVNIX0dTVC98aXNUb2tlbnx1c2VybmFtZXxlZ2dlcnNqZW5zQGdtYWlsLmNvbV4wfEozWDRKU0pLfDB8MF5eJDB8MXwyfEh8M3xJfDR8LTJ8NXw2fDd8JDh8JDl8SnxBfC0zXXxCfCQ5fEt8QXwtM11dfEN8RHxFfC0xfR10i.H7BAXttwX1aug4HXDnJwBh8m4njAi2ggQAMCNBGWRE

This query parameter will be shortened in the examples below for brevity.

2.2.1 Login

NOTE: Check if the SMC Cloud password uses "#". If so, the password must be changed to remove it before moving forward. See the SMC Cloud Start-up Guide for instructions. The # symbol is a reserved character in URL web services.

Description: Use user credentials to obtain a new API token.

HTTP Verb: GET or POST

URL: /rest/login

JSON Parameters: username (email address), password

Example GET URL: rest/login?username=admin@company.com&password=Ip6ISBKiOy5U

Example POST URL: rest/login

Example POST Payload:

```
{
  "username": "admin@company.com",
  "password": "Ip6ISBKiOy5U"
}
```

Example JSON Response:


```
{
  "message": "Logged in ok",
  "data": {
    "token":
"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.ImlkfDI1MGYxYWIwLWQzMjktNGQyNS1hMTE5LThlOTJhZDc0MzcyZXx0eXBlfH
RpbWVzdGFtcHxpc0VuY3J5cHRlZHxvcmlnaW58MTAtMC0yOC0xODFfNTcwMDB8cG9saWN5fDB8dHRsfGluYWN0aXZpdHlf
dGhyZXNob2xkfDF8cGVybWlzc2lvblNldEtleXwvT0VNK0FkbWluLy9fTUVTSF9HU1QvfGlzVG9rZW58dXNlcm5hbWV8ZWdnZ
XJzamVuc0BnbWFpbC5jb21eMHxKNjZCOFBGR3wwfDBeXiQwfDF8MnxIfDN8SXw0fC0yfDV8Nnw3fCQ4fCQ5fEp8QXwtM11
8QnwkOXxLfEF8LTNdXXxDfER8RXwtMXxGfEddIg.ihjURRHILcXp-iTl4FvjjAma1sI1_CDINGyQuSNacDM"
  },
  "error": null
}
```


2.3 Data Read/Write

2.3.1 Data Path Parameter

Whenever a parameter is specified as a path, this acts as a wildcard and includes all items that appear below the specified path.

Examples:

- /device1/plc1/point3 returns the JSON value of point3 (e.g. 1.234)
- /device1/plc1 returns a JSON value with all points belonging to plc1:

```
{
  "point1": true,
  "point2": false,
  "point3": 1.234
}
```

2.3.2 All Field Device Types

2.3.2.1 DeviceDataLog

Description: Get all data logs available to user, narrowed down by optional parameters.

HTTP Verb: GET or POST

URL: /rest/method/fieldpop-api/deviceDataLog

Parameters:

- **happn_token** - Token obtained from the login request.
- **deviceID** – the IIoT gateway ID (see **Appendix A.1.1**).
- **startUTCsec** (optional) - Start time in seconds since start of UTC epoch.
- **endUTCsec** (optional) - End time in seconds since start of UTC epoch.

JSON Response: Object containing time series data.

Example GET URL:
/rest/method/fieldpop-api/deviceDataLog?happn_token=eyJ0eXAiOiJKV1QiLCJhbGci&deviceID=lavenderquester_4k9q5oU-Q&startUTCsec=1502375626

Example POST URL: /rest/method/fieldpop-api/deviceDataLog?happn_token=eyJ0eXAiOiJKV1QiLCJhbGci

Example POST Payload:

```
{
  "parameters": {
    "deviceID": "lavenderquester_4k9q5oU-Q",
    "path": "lavenderquester_4k9q5oU-Q/BAC_IP_Steam_1/Blower",
    "startUTCsec": 1499332783,
    "endUTCsec": 1499334577
  }
}
```

Example JSON Response:

```
{
  "message": "Call successful",
  "data": {
    "Steam_2": {
      "Main_Valve": [
        {
          "time": 1502375626,
          "value": 0
        },
        {
          "time": 1502376528,
          "value": 0
        },
        {
          "time": 1502377421,
          "value": 0
        }
      ],
      "FIRING_RATE": [
        {
          "time": 1502375626,
          "value": 0
        },
        {
          "time": 1502376528,
          "value": 0
        },
        {
          "time": 1502377421,
          "value": 0
        }
      ]
    }
  },
  "error": null
}
```

2.3.3 Device Type: System View

2.3.3.1 GetDeviceData (System View v.2, v.3)

Description: Retrieve the data available under the specified device and data path.

HTTP Verb: GET or POST

URL: /rest/method/fieldpop-api/getDeviceData

Parameters:

- **happn_token** - token obtained from the login request.
- **IIoT device selector** (optional) – one of two ways of specifying the target IIoT device(s):
  - **deviceID** - the IIoT gateway ID (see **Appendix A.1.1**)
  - **deviceFilter** – MongoDB-style query object (see https://www.npmjs.com/package/sift); this option can only be specified by posting parameters in a JSON payload
- **path** (optional) - the path of the values to read from the specified gateway devices. This can be done at multiple levels:
  - no path - the response will contain all devices with all their properties
  - device - the response will contain the properties of the specified device
  - device/property - the response will just contain the property value

JSON Response: data object - includes all data matching specified path parameter:

- **no path** - all available data for all devices.
- **device** - all available properties for specified device.
- **device/property** - value of specified property on specified device.

Example GET URL: (no path specified)
/rest/method/fieldpop-api/getDeviceData?happn_token=eyJ0eXAiOiJKV1QiLC&deviceID=blazescowl_VygDWeqmz

Example JSON Response: (no path specified)


```
{
 "message": "Call successful",
 "data": {
  "Device_1_Profile_1": {
   "parameters": {
    "node_id": "1",
    "parameter1": "1",
    "parameter2": "1"
   },
   "type": "Device1",
   "mapping": "Device1",
   "name": "Device_1_Profile_1",
   "path": "/profileView/devices/Device_1_Profile_1",
   "node": "Dev_1_1",
   "events": {
    "alarm": false,
    "trouble": false,
    "warning": false
   },
   "activeEvents": {},
   "status_class": "status_normal",
   "status": "Normal",
   "oldStatus": "Normal",
   "property0": 1,
   "property1": 1,
   "alarm_property": 0,
   "trouble_property": 0
  },
  "Device_1_Profile_2": {
   "parameters": {
    "node_id": "2",
    "parameter1": "1",
    "parameter2": "2"
   },
   "type": "Device1",
   "mapping": "Device1",
   "name": "Device_1_Profile_2",
   "path": "/profileView/devices/Device_1_Profile_2",
   "node": "Dev_1_2",
   "events": {
    "alarm": false,
    "trouble": false,
    "warning": false
   },
   "activeEvents": {},
   "status_class": "status_normal",
   "status": "Normal",
   "oldStatus": "Normal",
   "property0": 1,
   "property1": 2,
   "alarm_property": 0,
   "trouble_property": 0
  },
  "Device_2_Profile_1": {
   "parameters": {
    "node_id": "3",
```

```
      "parameter1": "2",
      "parameter2": "1"
    },
    "type": "Device2",
    "mapping": "Device2",
    "name": "Device_2_Profile_1",
    "path": "/profileView/devices/Device_2_Profile_1",
    "events": {
      "alarm": false,
      "trouble": false,
      "warning": false
    },
    "activeEvents": {},
    "status_class": "status_normal",
    "status": "Normal",
    "oldStatus": "Normal",
    "property0": 1
  },
  "Device_2_Profile_2": {
    "parameters": {
      "node_id": "4",
      "parameter1": "2",
      "parameter2": "2"
    },
    "type": "Device2",
    "mapping": "Device2",
    "name": "Device_2_Profile_2",
    "path": "/profileView/devices/Device_2_Profile_2",
    "events": {
      "alarm": false,
      "trouble": false,
      "warning": false
    },
    "activeEvents": {},
    "status_class": "status_normal",
    "status": "Normal",
    "oldStatus": "Normal",
    "property0": 2
  }
},
"error": null
}
```


Example JSON Response: (path = Device_1_Profile_1)


```
{
 "message": "Call successful",
 "data": {
   "parameters": {
     "node_id": "1",
     "parameter1": "1",
     "parameter2": "1"
   },
   "type": "Device1",
   "mapping": "Device1",
   "name": "Device_1_Profile_1",
   "path": "/profileView/devices/Device_1_Profile_1",
   "node": "Dev_1_1",
   "events": {
     "alarm": false,
     "trouble": false,
     "warning": false
   },
   "activeEvents": {},
   "status_class": "status_normal",
   "status": "Normal",
   "oldStatus": "Normal",
   "property0": 1,
   "property1": 1,
   "alarm_property": 0,
   "trouble_property": 0
 },
 "error": null
}
```


Example JSON Response: (path = Device_1_Profile_1/property0)


```
{
 "message": "Call successful",
 "data": 1,
 "error": null
}
```


Example POST URL: /rest/method/fieldpop-api/getDeviceData?happn_token=eyJ0eXAiOiJKV1QiLCJhbGci

Example POST Payload: (array of paths)


```
{
  "parameters": {
    "deviceFilter": {
      "$ne": "Normal"
    },
    "path": [
      "BAC_IP_Falcon_Hydronic_1_/type",
      "BAC_IP_Falcon_Hydronic_1_/BURN_CTL_STATUS",
      "BAC_IP_Falcon_Hydronic_1_/INLETWATERTEMPDEGF",
      "BAC_IP_Falcon_Steam_2_/type",
      "BAC_IP_Falcon_Steam_2_/BURN_CTL_STATUS",
      "BAC_IP_Falcon_Steam_2_/INLETWATERTEMPDEGF"
    ]
  }
}
```

Example JSON Response: (array of paths)


```
{
  "message": "Call successful",
  "data": {
    "lightninghide_EJ2PQsKJb": {
      "data": [
        {
          "error": {
            "message": "Unknown field device:
BAC_IP_Falcon_Hydronic_1_"
          }
        },
        {
          "error": {
            "message": "Unknown field device:
BAC_IP_Falcon_Hydronic_1_"
          }
        },
        {
          "error": {
            "message": "Unknown field device:
BAC_IP_Falcon_Hydronic_1_"
          }
        },
        {
          "data": "BAC_IP_Falcon_Steam"
        },
        {
          "data": 1
        },
        {
          "error": {
            "message": "Unknown property for device
BAC_IP_Falcon_Steam_2_: INLETWATERTEMPDEGF"
          }
        }
      ]
    },
    "shagcrest_Nk9hhqHFz": {
      "data": [
        {
          "data": "BAC_IP_Falcon_Hydronic"
        },
        {
          "data": 48
        },
        {
          "data": 78.8
        },
        {
          "error": {
            "message": "Unknown field device:
BAC_IP_Falcon_Steam_2_"
          }
        },
        {
          "error": {
            "message": "Unknown field device:
BAC_IP_Falcon_Steam_2_"
          }
        },
        {
          "error": {
            "message": "Unknown field device:
BAC_IP_Falcon_Steam_2_"
          }
        }
      ]
    },
    "cybergecko_VJU": {
      "data": [
        {
          "data": "BAC_IP_Falcon_Hydronic"
        },
        {
          "data": 48
        },
        {
          "data": 82.94
        },
        {
          "error": {
            "message": "Unknown field device:
BAC_IP_Falcon_Steam_2_"
          }
        },
        {
          "error": {
            "message": "Unknown field device:
BAC_IP_Falcon_Steam_2_"
          }
        },
        {
          "error": {
            "message": "Unknown field device:
BAC_IP_Falcon_Steam_2_"
          }
        }
      ]
    },
    "lavenderquester_4k9q5oU-Q": {
      "data": [
        {
          "error": {
            "message": "Unknown field device:
BAC_IP_Falcon_Hydronic_1_"
          }
        },
        {
          "error": {
            "message": "Unknown field device:
BAC_IP_Falcon_Hydronic_1_"
          }
        },
        {
          "error": {
            "message": "Unknown field device:
BAC_IP_Falcon_Hydronic_1_"
          }
        },
        {
          "data": "BAC_IP_Falcon_Steam"
        },
        {
          "data": 1
        },
        {
          "error": {
            "message": "Unknown property for device
BAC_IP_Falcon_Steam_2_: INLETWATERTEMPDEGF"
          }
        }
      ]
    },
    "crackfisher_N1-": {
      "error": {
        "message": "timeout"
      }
    },
    "hurricanestone_NkvQCitk-": {
      "error": {
        "message": "timeout"
      }
    },
```

```
    "tulipgorilla_EyGpqsFkb": {
      "error": {
        "message": "timeout"
      }
    }
  },
  "error": null
}
```


2.3.3.2 GetDeviceData (System View v.4)

Description: Retrieve the data available under the specified device(s) and data path(s).

HTTP Verb: GET or POST

URL: /rest/method/fieldpop-api/getDeviceData

Data Model: The underlying data is structured as an object keyed by IIoT gateway identifier (deviceID) values at the top level. Each gateway value is an array of downstream field device objects. Here is a schematic example of the full available data for a given user.


```
{
  "gateway_1": {
    "data": [
      {
        "parameters": {
          "node_id": "1"
        },
        "type": "BAC_IP_Falcon_Hydronic",
        "mapping": "BAC_IP_Falcon_Hydronic",
        "name": "BAC_IP_Falcon_Hydronic_1_",
        "path":
"/profileView/devices/BAC_IP_Falcon_Hydronic_1_",
        "node": "Dev_1",
        "events": {
          "alarm": false,
          "trouble": false,
          "warning": false
        },
        "activeEvents": {},
        "address": "1",
        "location": "R&amp;D Lab Unit 1",
        "status_class": "status_normal",
        "status": "Normal",
        "oldStatus": "Normal",
        "BOILERLEADLAGSETPOINTDEGC": 58.9
      },
      {
        "parameters": {
          "node_id": "2"
        },
        "type": "BAC_IP_Falcon_Steam",
        "mapping": "BAC_IP_Falcon_Steam",
        "name": "BAC_IP_Falcon_Steam_2_",
        "path":
"/profileView/devices/BAC_IP_Falcon_Steam_2_",
        "node": "Dev_2",
        "events": {
          "alarm": false,
          "trouble": false,
          "warning": false
        },
        "activeEvents": {},
        "address": "2",
        "status_class": "status_normal",
        "status": "Normal",
        "oldStatus": "Normal",
        "BURNER_ENABLE": 1
      }
    ]
  },
  "gateway_2": {
    "data": [
      {
        "parameters": {
          "node_id": "1"
        },
        "type": "BAC_IP_Falcon_Hydronic",
        "mapping": "BAC_IP_Falcon_Hydronic",
        "name": "BAC_IP_Falcon_Hydronic_1_",
        "path":
"/profileView/devices/BAC_IP_Falcon_Hydronic_1_",
        "node": "Dev_1",
        "events": {
          "alarm": false,
          "trouble": false,
          "warning": false
        },
        "activeEvents": {},
        "address": "1",
        "location": "R&amp;D Lab Unit 1",
        "status_class": "status_normal",
        "status": "Normal",
        "oldStatus": "Normal",
        "BOILERLEADLAGSETPOINTDEGC": 58.9
      },
      {
        "parameters": {
          "node_id": "2"
        },
        "type": "BAC_IP_Falcon_Steam",
        "mapping": "BAC_IP_Falcon_Steam",
        "name": "BAC_IP_Falcon_Steam_2_",
        "path":
"/profileView/devices/BAC_IP_Falcon_Steam_2_",
        "node": "Dev_2",
        "events": {
          "alarm": true,
          "trouble": false,
          "warning": false
        },
        "activeEvents": {},
        "address": "2",
        "status_class": "status_normal",
        "status": "Alarm",
        "oldStatus": "Normal",
        "BURNER_ENABLE": 1
      }
    ]
  }
}
```

A query can narrow the entire data set down by:

- selecting only certain gateway devices (e.g. gateways reporting an Alarm state), using the **deviceFilter** parameter.
- selecting only certain downstream devices within the selected gateway devices (e.g. those reporting an Alarm status), using the **dataFilter** parameter.
- selecting only certain data paths to be read from the selected downstream devices, using the path parameter.

Parameters:

- **happn_token** - token obtained from the login request.
- **IIoT device selector** (optional) – one of two ways of specifying the target IIoT device(s):
  - **deviceID** – the IIoT gateway ID (see **Appendix A.1.1**)
  - **deviceFilter** – MongoDB-style query object (see https://www.npmjs.com/package/sift); this option can only be specified by posting parameters in a JSON payload

  If no gateway device selection criteria are given, all gateways are queried. **Use with care! Not specifying IIoT device(s) will cause requests to be sent to all IIoT devices!**
- **path** (optional):
  - either a path string, pointing to a value within each downstream field device (boiler) object, e.g. "propertyName/subPropertyName"
  - or an array of such path strings, e.g. ["propertyName_1/subPropertyName_1", "propertyName_2/subPropertyName_2"]
  - note that any unmatched path value is simply ignored and left out of the result
  - if no path is specified, all available values are returned
- **options** (optional) – an optional object containing:
  - dataFilter – a MongoDB-style sift query object (see https://www.npmjs.com/package/sift); result objects relating to downstream field devices are only included in the response if they match this filter

JSON Response: data object containing an **Array** of filtered query response objects.

Example 1 – POST URL: (no JSON parameters specified)
/rest/method/fieldpop-api/getDeviceData?happn_token=eyJ0eXAiOiJKV1QiLC

Example 1 – JSON Response:


```
{
  "message": "Call successful",
  "data": {
    "gateway_1": {
      "data": [
        {
          "parameters": {
            "node_id": "1"
          },
          "type": "BAC_IP_Falcon_Hydronic",
          "mapping": "BAC_IP_Falcon_Hydronic",
          "name": "BAC_IP_Falcon_Hydronic_1_",
          "path":
"/profileView/devices/BAC_IP_Falcon_Hydronic_1_",
          "node": "Dev_1",
          "events": {
            "alarm": false,
            "trouble": false,
            "warning": false
          },
          "activeEvents": {},
          "address": "1",
          "location": "R&amp;D Lab Unit 1",
          "status_class": "status_normal",
          "status": "Normal",
          "oldStatus": "Normal",
          "BOILERLEADLAGSETPOINTDEGC": 58.9
        },
        {
          "parameters": {
            "node_id": "2"
          },
          "type": "BAC_IP_Falcon_Steam",
          "mapping": "BAC_IP_Falcon_Steam",
          "name": "BAC_IP_Falcon_Steam_2_",
          "path":
"/profileView/devices/BAC_IP_Falcon_Steam_2_",
          "node": "Dev_2",
          "events": {
            "alarm": false,
            "trouble": false,
            "warning": false
          },
          "activeEvents": {},
```

```
          "address": "2",
          "status_class": "status_normal",
          "status": "Normal",
          "oldStatus": "Normal",
          "BURNER_ENABLE": 1
        }
      ]
    },
    "gateway_2": {
      "data": [
        {
          "parameters": {
            "node_id": "1"
          },
          "type": "BAC_IP_Falcon_Hydronic",
          "mapping": "BAC_IP_Falcon_Hydronic",
          "name": "BAC_IP_Falcon_Hydronic_1_",
          "path":
"/profileView/devices/BAC_IP_Falcon_Hydronic_1_",
          "node": "Dev_1",
          "events": {
            "alarm": false,
            "trouble": false,
            "warning": false
          },
          "activeEvents": {},
          "address": "1",
          "location": "R&amp;D Lab Unit 1",
          "status_class": "status_normal",
          "status": "Normal",
          "oldStatus": "Normal",
          "BOILERLEADLAGSETPOINTDEGC": 58.9
        },
        {
          "parameters": {
            "node_id": "2"
          },
          "type": "BAC_IP_Falcon_Steam",
          "mapping": "BAC_IP_Falcon_Steam",
          "name": "BAC_IP_Falcon_Steam_2_",
          "path":
"/profileView/devices/BAC_IP_Falcon_Steam_2_",
          "node": "Dev_2",
          "events": {
            "alarm": true,
            "trouble": false,
            "warning": false
          },
          "activeEvents": {},
          "address": "2",
          "status_class": "status_normal",
          "status": "Alarm",
          "oldStatus": "Normal",
          "BURNER_ENABLE": 1
        }
      ]
    }
  },
  "error": null
}
```


Example 2 – POST URL: /rest/method/fieldpop-api/getDeviceData?happn_token=eyJ0eXAiOiJKV1QiLC

Example 2 – JSON POST: (deviceID specified)


```
{
  "parameters": {
    "deviceID": "gateway_1"
  }
}
```


Example 2 – JSON Response:


```
{
  "message": "Call successful",
  "data": [
    {
      "parameters": {
        "node_id": "1"
      },
      "type": "BAC_IP_Falcon_Hydronic",
      "mapping": "BAC_IP_Falcon_Hydronic",
      "name": "BAC_IP_Falcon_Hydronic_1_",
      "path":
"/profileView/devices/BAC_IP_Falcon_Hydronic_1_",
      "node": "Dev_1",
      "events": {
        "alarm": false,
        "trouble": false,
        "warning": false
      },
      "activeEvents": {},
      "address": "1",
      "location": "R&amp;D Lab Unit 1",
      "status_class": "status_normal",
      "status": "Normal",
      "oldStatus": "Normal",
      "BOILERLEADLAGSETPOINTDEGC": 58.9
    },
    {
      "parameters": {
        "node_id": "2"
      },
      "type": "BAC_IP_Falcon_Steam",
      "mapping": "BAC_IP_Falcon_Steam",
      "name": "BAC_IP_Falcon_Steam_2_",
      "path":
"/profileView/devices/BAC_IP_Falcon_Steam_2_",
      "node": "Dev_2",
      "events": {
        "alarm": false,
        "trouble": false,
        "warning": false
      },
      "activeEvents": {},
      "address": "2",
      "status_class": "status_normal",
      "status": "Normal",
      "oldStatus": "Normal",
      "BURNER_ENABLE": 1
    }
  ],
  "error": null
}
```

Example 3 – POST URL:
/rest/method/fieldpop-api/getDeviceData?happn_token=eyJ0eXAiOiJKV1QiLC

Example 3 – JSON POST: (path specified, but no IIoT device selector or dataFilter)

```
{
  "parameters": {
    "path": [
      "name",
      "type",
      "status",
      "BOILERLEADLAGSETPOINTDEGC",
      "BURNER_ENABLE"
    ]
  }
}
```

Example 3 – JSON Response:

```
{
  "message": "Call successful",
  "data": {
    "gateway_1": {
      "data": [
        {
          "type": "BAC_IP_Falcon_Hydronic",
          "name": "BAC_IP_Falcon_Hydronic_1_",
          "status": "Normal",
          "BOILERLEADLAGSETPOINTDEGC": 58.9
        },
        {
          "type": "BAC_IP_Falcon_Steam",
          "name": "BAC_IP_Falcon_Steam_2_",
          "status": "Normal",
          "BURNER_ENABLE": 1
        }
      ]
    },
```

```
    "gateway_2": {
      "data": [
        {
          "type": "BAC_IP_Falcon_Hydronic",
          "name": "BAC_IP_Falcon_Hydronic_1_",
          "status": "Normal",
          "BOILERLEADLAGSETPOINTDEGC": 58.9
        },
        {
          "type": "BAC_IP_Falcon_Steam",
          "name": "BAC_IP_Falcon_Steam_2_",
          "status": "Alarm",
          "BURNER_ENABLE": 1
        }
      ]
    }
  },
  "error": null
}
```

Example 4 – POST URL:
/rest/method/fieldpop-api/getDeviceData?happn_token=eyJ0eXAiOiJKV1QiLC

Example 4 – JSON POST: (path and dataFilter specified, but no IIoT device selector)

```
{
  "parameters": {
    "path": [
      "property0",
      "property1",
      "events/alarm"
    ],
    "options": {
      "dataFilter": {
        "name": {
          "$eq": "Device_1_Profile_1"
        }
      }
    }
  }
}
```

Example 4 – JSON Response:


```
{
  "message": "Call successful",
  "data": {
    "gateway_1": {
      "data": [
        {
          "events": {
            "alarm": false
          },
          "property0": 1,
          "property1": 1
        }
      ]
    },
    "gateway_2": {
      "data": [ ]
    }
  },
  "error": null
}
```


Example 5 – JSON POST: (deviceFilter, path and dataFilter specified)


```
{
  "parameters": {
    "deviceFilter": {
      "state": {
        "$eq": "Alarm"
      }
    },
    "path": [
      "name",
      "type",
      "status",
      "BOILERLEADLAGSETPOINTDEGC",
      "BURNER_ENABLE"
    ],
    "options": {
      "dataFilter": {
        "status": {
          "$eq": "Alarm"
        }
      }
    }
  }
}
```


Example 5 – JSON Response:


```
{
  "message": "Call successful",
  "data": {
    "gateway_2": {
      "data": [
        {
          "type": "BAC_IP_Falcon_Steam",
          "name": "BAC_IP_Falcon_Steam_2_",
          "status": "Alarm",
          "BURNER_ENABLE": 1
        }
      ]
    }
  },
  "error": null
}
```

2.3.3.3 SetDeviceData (System View)

Description: Set the data available under the specified device and data path.

HTTP Verb: POST

URL: /rest/method/fieldpop-api/setDeviceData

Query String Parameters:

- **happn_token** - token obtained from the login request.

HTTP Headers:

- **Content-Type** - application/json

Body Parameters (JSON):

- **parameters** - object containing the following fields:
  - **deviceID** - the IIoT gateway ID (see **Appendix A.1.1**)
  - **path** - the path of the value to set; this must be done at the leaf level "deviceName/propertyName"
  - **value** - the value to set

Example Body Parameters:

```
{
  "parameters": {
    "deviceID": "blazescowl_VygDWeqmz",
    "path": "BAC_IP_EVG_1_/BoilerModel",
    "value": 50
  }
}
```

Example JSON Response:

```
{
  "message": "Call successful",
  "data": null,
  "error": null
}
```

2.3.4 Device Type: BACnet Explorer NG

Please note that the 'BACnet Explorer NG' product is obsolete. Contact support to upgrade your product to a 'BACnet IoT Gateway' – see BACnet IoT Gateway.

2.3.4.1 GetDeviceData (BACnet Explorer NG)

Description: Retrieve the data available under the specified device and data path.

HTTP Verb: POST

URL: /rest/method/fieldpop-api/getDeviceData

Query String Parameters:

- **happn_token** - token obtained from the login request.

HTTP Headers:

- **Content-Type** - application/json

Body Parameters (JSON):

- **parameters** - object containing the following fields:
  - **deviceID** - the IIoT gateway ID (see **Appendix A.1.1**)
  - **path** - the path of the value to get (must be 3 levels deep: "deviceInstance/objectType:objectInstance/propertyIdentifier")
  - **options** - object containing the following fields:
    - **asn1** - must be set to **true** since default representation might change in the future

Example Body Parameters:

```
{
  "parameters": {
    "deviceID": "bigelf_rkr",
    "path": "200/analog-value:0/present-value",
    "options": {
      "asn1": true
    }
  }
}
```

Example JSON Response:

```
{
  "message": "Call successful",
  "data": {
    "value": 28,
    "type": "REAL"
  },
  "error": null
}
```

2.3.4.2 SetDeviceData (BACnet Explorer NG)

Description: Retrieve the data available under the specified device and data path.

HTTP Verb: POST

URL: /rest/method/fieldpop-api/setDeviceData

Query String Parameters:

- **happn_token** - token obtained from the login request.

HTTP Headers:

- **Content-Type** - application/json

Body Parameters (JSON):

- **parameters** - object containing the following fields:
  - **deviceID** - this can be obtained from the local SMC Cloud GUI on the device (also visible on the SMC Cloud website by clicking on the device pin on the map)
  - **path** - the path of the value to get (must be 3 levels deep: "deviceInstance/objectType:objectInstance/propertyIdentifier")
  - **value** - ASN.1 representation of the property value to set
  - **options** - object containing the following fields:
    - **asn1** - must be set to **true** since default representation might change in the future
    - **priority** (optional) - priority to set from **1** to **16**

Example Body Parameters:

```
{
  "parameters": {
    "deviceID": "bigelf_rkr",
    "path": "200/analog-value:0/present-value",
    "value": {
      "value": 24,
      "type": "REAL"
    },
    "options": {
      "asn1": true,
      "priority": 16
    }
  }
}
```

Example JSON Response:

```
{
  "message": "Call successful",
  "data": null,
  "error": null
}
```

2.3.5 Device Type: BACnet IoT Gateway

Please note that this product makes use of a device API version - see **Appendix A.1.2**.

- Product versions below "4.0.0" do not have a device API version and can be assumed to support "deviceAPIVersion" 1. Version 1 requests may specify this parameter.
- Product version "4.0.0" supports only "deviceAPIVersion" 2. Version 2 requests must specify this parameter.

2.3.5.1 GetDeviceData (v.1, v.2)

Description: Retrieve the data available under the specified device(s) and data path(s).

HTTP Verb: GET or POST

URL: /rest/method/fieldpop-api/getDeviceData

Data Model: The underlying data is structured as an object keyed by IIoT gateway identifier (deviceID) values at the top level. Each gateway value is an array of downstream field device objects.

A query can narrow the entire data set down by:

- selecting only certain gateway devices (e.g. gateways reporting an Alarm state), using the **deviceFilter** parameter.
- selecting only certain downstream devices within the selected gateway devices (e.g. those reporting an Alarm status), using the **dataFilter** parameter.
- selecting only certain data paths to be read from the selected downstream devices, using the **path** parameter.

Parameters:

- **happn_token** – token obtained from the login request.
- **deviceAPIVersion** – see **Section 2.3.5**.
- **IIoT device selector** (optional) – one of two ways of specifying the target IIoT device(s):
  - **deviceID** – the IIoT gateway ID (see **Appendix A.1.1**)
  - **deviceFilter** – MongoDB-style query object (see https://www.npmjs.com/package/sift); this option can only be specified by posting parameters in a JSON payload

If no gateway device selection criteria are given, all gateways are queried. **Use with care! Not specifying IIoT device(s) will cause requests to be sent to all IIoT devices!**

- **path** (optional)
  - either a path string, pointing to a value within each downstream field device (boiler) object, e.g. "propertyName/subPropertyName"
  - or an array of such path strings, e.g. ["propertyName_1/subPropertyName_1", "propertyName_2/subPropertyName_2"]
  - note that any unmatched path value is simply ignored and left out of the result
  - if no path is specified, all available values are returned
- **options** (optional) – an optional object containing:
  - **dataFilter** – a MongoDB-style sift query object (see https://www.npmjs.com/package/sift); result objects relating to downstream field devices are only included in the response if they match this filter

JSON Response: data object containing an **Array** of filtered query response objects.

Example 1 – GET URL: (HTTP GET with "deviceID" parameter not specified - will send to all gateways you have access to)
/rest/method/fieldpop-api/getDeviceData?happn_token=eyJ0eXAiOiJKV1QiLC&deviceAPIVersion=2

Example 1 – JSON Response:


```
{
  "message": "Call successful",
  "error": null,
  "data": {
    "gateway_1": {
      "data": [
        {
          "deviceInstance": 1,
          "deviceName": "WeatherLink_1",
          "pollInterval": 5,
          "offline": false,
          "objects": {
            "analog-input:1": {
              "objectType": "analog-input",
              "objectInstance": 1,
              "objectName": "INSIDE_TEMPERATURE",
              "properties": {
                "present-value": {
                  "propertyIdentifier": "present-value",
                  "value": {
                    "value": 42.70000076293945,
                    "type": "REAL"
                  },
                  "lastRead": 1513687777060,
                  "log": {
                    "enabled": false,
                    "keys": []
                  }
                }
              }
            },
            "analog-input:2": {
              "objectType": "analog-input",
              "objectInstance": 2,
              "objectName": "OUTSIDE_TEMPERATURE",
              "properties": {
                "present-value": {
                  "propertyIdentifier": "present-value",
                  "value": {
                    "value": 39.79999923706055,
                    "type": "REAL"
                  },
                  "lastRead": 1513687777060,
                  "log": {
                    "enabled": true,
                    "keys": [
                      "1/analog-input:2/present-value"
                    ]
                  }
                }
              }
            }
          }
        },
        {
          "deviceInstance": 2,
          "deviceName": "WeatherLink_2",
          "pollInterval": 5,
          "offline": false,
          "objects": {
            "analog-input:1": {
              "objectType": "analog-input",
              "objectInstance": 1,
              "objectName": "INSIDE_TEMPERATURE",
              "properties": {
                "present-value": {
                  "propertyIdentifier": "present-value",
                  "value": {
                    "value": 87.69999694824219,
                    "type": "REAL"
                  },
                  "lastRead": 1513687777497,
                  "log": {
                    "enabled": false,
                    "keys": []
                  }
                }
              }
            },
            "analog-input:2": {
              "objectType": "analog-input",
              "objectInstance": 2,
              "objectName": "OUTSIDE_TEMPERATURE",
              "properties": {
                "present-value": {
                  "propertyIdentifier": "present-value",
                  "value": {
                    "value": 77.5999984741211,
                    "type": "REAL"
                  },
                  "lastRead": 1513687777497,
                  "log": {
                    "enabled": true,
                    "keys": [
                      "2/analog-input:2/present-value"
                    ]
                  }
                }
              }
            }
          }
        }
      ]
    },
    "gateway_2": {
      "data": [
        {
          "deviceInstance": 1,
          "deviceName": "WeatherLink_1",
          "pollInterval": 5,
          "offline": false,
          "objects": {
            "analog-input:1": {
              "objectType": "analog-input",
              "objectInstance": 1,
              "objectName": "INSIDE_TEMPERATURE",
              "properties": {
                "present-value": {
                  "propertyIdentifier": "present-value",
                  "value": {
                    "value": 42.70000076293945,
                    "type": "REAL"
                  },
                  "lastRead": 1513687777060,
                  "log": {
                    "enabled": false,
                    "keys": []
                  }
                }
              }
            },
            "analog-input:2": {
```

```
            "objectType": "analog-input",
            "objectInstance": 2,
            "objectName": "OUTSIDE_TEMPERATURE",
            "properties": {
              "present-value": {
                "propertyIdentifier": "present-value",
                "value": {
                  "value": 39.79999923706055,
                  "type": "REAL"
                },
                "lastRead": 1513687777060,
                "log": {
                  "enabled": true,
                  "keys": [
                    "1/analog-input:2/present-value"
                  ]
                }
              }
            }
          }
        }
      },
      {
        "deviceInstance": 2,
        "deviceName": "WeatherLink_2",
        "pollInterval": 5,
        "offline": false,
        "objects": {
          "analog-input:1": {
            "objectType": "analog-input",
            "objectInstance": 1,
            "objectName": "INSIDE_TEMPERATURE",
            "properties": {
              "present-value": {
                "propertyIdentifier": "present-value",
                "value": {
                  "value": 87.69999694824219,
                  "type": "REAL"
                },
                "lastRead": 1513687777497,
                "log": {
                  "enabled": false,
                  "keys": []
                }
              }
            }
          },
          "analog-input:2": {
            "objectType": "analog-input",
            "objectInstance": 2,
            "objectName": "OUTSIDE_TEMPERATURE",
            "properties": {
              "present-value": {
                "propertyIdentifier": "present-value",
                "value": {
                  "value": 77.5999984741211,
                  "type": "REAL"
                },
                "lastRead": 1513687777497,
                "log": {
                  "enabled": true,
                  "keys": [
                    "2/analog-input:2/present-value"
                  ]
                }
              }
            }
          }
        }
      }
    ]
  }
}
```


Example 2 – URL: (HTTP GET with "deviceID" parameter specified in URL)
/rest/method/fieldpop-api/getDeviceData?happn_token=eyJ0eXAiOiJKV1QiLC&deviceID=gateway_1&deviceAPIVersion=2

Example 2 – JSON Response:


```
{
  "message": "Call successful",
  "data": [
    {
      "deviceInstance": 1,
      "deviceName": "WeatherLink_1",
      "pollInterval": 5,
      "offline": false,
      "objects": {
        "analog-input:1": {
          "objectType": "analog-input",
          "objectInstance": 1,
          "objectName": "INSIDE_TEMPERATURE",
          "properties": {
            "present-value": {
              "propertyIdentifier": "present-value",
              "value": {
                "value": 42.39999771118164,
                "type": "REAL"
              },
              "lastRead": 1513689232174,
              "log": {
                "enabled": false,
                "keys": []
```



```
            }
          }
        }
      },
      "analog-input:2": {
        "objectType": "analog-input",
        "objectInstance": 2,
        "objectName": "OUTSIDE_TEMPERATURE",
        "properties": {
          "present-value": {
            "propertyIdentifier": "present-value",
            "value": {
              "value": 39.5,
              "type": "REAL"
            },
            "lastRead": 1513689232174,
            "log": {
              "enabled": true,
              "keys": [
                "1/analog-input:2/present-value"
              ]
            }
          }
        }
```

```
        }
      }
    },
    {
      "deviceInstance": 2,
      "deviceName": "WeatherLink_2",
      "pollInterval": 5,
      "offline": false,
      "objects": {
        "analog-input:1": {
          "objectType": "analog-input",
          "objectInstance": 1,
          "objectName": "INSIDE_TEMPERATURE",
          "properties": {
            "present-value": {
              "propertyIdentifier": "present-value",
              "value": {
                "value": 87.9000015258789,
                "type": "REAL"
              },
              "lastRead": 1513689232285,
              "log": {
                "enabled": false,
                "keys": []
              }
            }
          }
        },
        "analog-input:2": {
          "objectType": "analog-input",
          "objectInstance": 2,
          "objectName": "OUTSIDE_TEMPERATURE",
          "properties": {
            "present-value": {
              "propertyIdentifier": "present-value",
              "value": {
                "value": 78.19999694824219,
                "type": "REAL"
              },
              "lastRead": 1513689232285,
              "log": {
                "enabled": true,
                "keys": [
                  "2/analog-input:2/present-value"
                ]
              }
            }
          }
        }
      }
    }
  ],
  "error": null
}
```


Example 3 – URL: (HTTP POST with "deviceID" parameter specified in body)
/rest/method/fieldpop-api/getDeviceData?happn_token=eyJ0eXAiOiJKV1QiLC

Example 3 – JSON body:


```
{
  "parameters": {
    "deviceID": "gateway_1",
    "deviceAPIVersion": "2"
  }
}
```


Example 3 – JSON Response:


```
{
  "message": "Call successful",
  "data": [
    {
      "deviceInstance": 1,
      "deviceName": "WeatherLink_1",
      "pollInterval": 5,
      "offline": false,
      "objects": {
        "analog-input:1": {
          "objectType": "analog-input",
          "objectInstance": 1,
          "objectName": "INSIDE_TEMPERATURE",
          "properties": {
            "present-value": {
              "propertyIdentifier": "present-value",
              "value": {
                "value": 42.39999771118164,
                "type": "REAL"
              },
              "lastRead": 1513689232174,
              "log": {
                "enabled": false,
                "keys": []
              }
            }
          }
        },
        "analog-input:2": {
          "objectType": "analog-input",
          "objectInstance": 2,
          "objectName": "OUTSIDE_TEMPERATURE",
          "properties": {
            "present-value": {
              "propertyIdentifier": "present-value",
              "value": {
                "value": 39.5,
                "type": "REAL"
              },
              "lastRead": 1513689232174,
              "log": {
                "enabled": true,
                "keys": [
                  "1/analog-input:2/present-value"
                ]
              }
            }
          }
        }
      }
    }
```

```
    },
    {
      "deviceInstance": 2,
      "deviceName": "WeatherLink_2",
      "pollInterval": 5,
      "offline": false,
      "objects": {
        "analog-input:1": {
          "objectType": "analog-input",
          "objectInstance": 1,
          "objectName": "INSIDE_TEMPERATURE",
          "properties": {
            "present-value": {
              "propertyIdentifier": "present-value",
              "value": {
                "value": 87.9000015258789,
                "type": "REAL"
              },
              "lastRead": 1513689232285,
              "log": {
                "enabled": false,
                "keys": []
              }
            }
          }
        },
        "analog-input:2": {
          "objectType": "analog-input",
          "objectInstance": 2,
          "objectName": "OUTSIDE_TEMPERATURE",
          "properties": {
            "present-value": {
              "propertyIdentifier": "present-value",
              "value": {
                "value": 78.19999694824219,
                "type": "REAL"
              },
              "lastRead": 1513689232285,
              "log": {
                "enabled": true,
                "keys": [
                  "2/analog-input:2/present-value"
                ]
              }
            }
          }
        }
      }
    }
  ],
  "error": null
}
```


Example 4 – URL: (HTTP POST with "path" parameter specified)
/rest/method/fieldpop-api/getDeviceData?happn_token=eyJ0eXAiOiJKV1QiLC

Example 4 – JSON body:


```
{
  "parameters": {
    "deviceID": "gateway_1",
    "deviceAPIVersion": "2",
    "path": [
      "deviceName",
      "offline",
      "objects/analog-input:1/objectName",
      "objects/analog-input:1/properties/present-value/value",
      "objects/analog-input:1/properties/present-value/lastRead"
    ]
  }
}
```


Example 4 – JSON Response:


```
{
  "message": "Call successful",
  "data": [
    {
      "deviceName": "WeatherLink_1",
      "offline": false,
      "objects/analog-input:1/objectName":
"INSIDE_TEMPERATURE",
      "objects/analog-input:1/properties/present-value/value": {
        "value": 41.70000076293945,
        "type": "REAL"
      },
      "objects/analog-input:1/properties/present-
value/lastRead": 1513692188395
    },
    {
      "deviceName": "WeatherLink_2",
      "offline": false,
      "objects/analog-input:1/objectName":
"INSIDE_TEMPERATURE",
      "objects/analog-input:1/properties/present-value/value": {
        "value": 88.5,
        "type": "REAL"
      },
      "objects/analog-input:1/properties/present-
value/lastRead": 1513692188494
    }
  ],
  "error": null
}
```


Example 5 – URL: (HTTP POST with "dataFilter" parameter specified)
/rest/method/fieldpop-api/getDeviceData?happn_token=eyJ0eXAiOiJKV1QiLC

Example 5 – JSON body:


```
{
  "parameters": {
    "deviceID": "gateway_1",
    "deviceAPIVersion": "2",
    "path": [
      "deviceName",
      "offline",
      "objects/analog-input:1/objectName",
      "objects/analog-input:1/properties/present-value/value",
      "objects/analog-input:1/properties/present-value/lastRead"
    ],
    "options": {
      "dataFilter": {
        "deviceInstance": {
          "$eq": 1
        }
      }
    }
  }
}
```


Example 5 – JSON Response:


```
{
  "message": "Call successful",
  "data": [
    {
      "deviceName": "WeatherLink_1",
      "offline": false,
      "objects/analog-input:1/objectName": "INSIDE_TEMPERATURE",
      "objects/analog-input:1/properties/present-value/value": {
        "value": 41.599998474121094,
        "type": "REAL"
      },
      "objects/analog-input:1/properties/present-value/lastRead": 1513692583828
    }
  ],
  "error": null
}
```


Example 6 – URL: (HTTP POST with "deviceFilter" parameter specified instead of "deviceID")
/rest/method/fieldpop-api/getDeviceData?happn_token=eyJ0eXAiOiJKV1QiLC

Example 6 – JSON body:


```
{
  "parameters": {
    "deviceFilter": {
      "state": {
        "$eq": "Alarm"
      }
    },
    "deviceAPIVersion": "2"
  }
}
```


Example 6 – JSON Response:


```
{
  "message": "Call successful",
  "data": {
    "gateway_1": {
      "data": [
        {
          "deviceInstance": 1,
          "deviceName": "WeatherLink_1",
          "pollInterval": 5,
          "offline": false,
          "objects": {
            "analog-input:1": {
              "objectType": "analog-input",
              "objectInstance": 1,
              "objectName": "INSIDE_TEMPERATURE",
              "properties": {
                "present-value": {
                  "propertyIdentifier": "present-value",
                  "value": {
                    "value": 41.5,
```

```
                "type": "REAL"
              },
              "lastRead": 1513693293092,
              "log": {
                "enabled": false,
                "keys": []
              }
            }
          }
        },
        "analog-input:2": {
          "objectType": "analog-input",
          "objectInstance": 2,
          "objectName": "OUTSIDE_TEMPERATURE",
          "properties": {
            "present-value": {
              "propertyIdentifier": "present-value",
              "value": {
                "value": 38.89999771118164,
                "type": "REAL"
              },
              "lastRead": 1513693293092,
              "log": {
                "enabled": true,
                "keys": [
                  "1/analog-input:2/present-value"
                ]
              }
            }
          }
        }
      }
    },
    {
      "deviceInstance": 2,
      "deviceName": "WeatherLink_2",
      "pollInterval": 5,
      "offline": false,
      "objects": {
        "analog-input:1": {
          "objectType": "analog-input",
          "objectInstance": 1,
          "objectName": "INSIDE_TEMPERATURE",
          "properties": {
            "present-value": {
              "propertyIdentifier": "present-value",
              "value": {
                "value": 88.5999984741211,
                "type": "REAL"
              },
              "lastRead": 1513693293192,
              "log": {
                "enabled": false,
                "keys": []
              }
            }
          }
        },
        "analog-input:2": {
          "objectType": "analog-input",
          "objectInstance": 2,
          "objectName": "OUTSIDE_TEMPERATURE",
          "properties": {
            "present-value": {
              "propertyIdentifier": "present-value",
              "value": {
                "value": 79.9000015258789,
                "type": "REAL"
              },
              "lastRead": 1513693293192,
              "log": {
                "enabled": true,
                "keys": [
                  "2/analog-input:2/present-value"
                ]
              }
            }
          }
        }
      }
    }
  ]
}
},
"error": null
}
```

2.3.5.2 CallDeviceMethod – GetBACnetDeviceInfo (v.1)

Description: Get BACnet device info by sending a 'who-is' request out on BACnet.

HTTP Verb: GET or POST

URL: /rest/method/fieldpop-api/callDeviceMethod

Parameters:

- **happn_token** – token obtained from the login request.
- **deviceID** – the IIoT gateway ID (see **Appendix A.1.1**).
- **methodName** – "getBACnetDeviceInfo"
- **deviceInstance** – the BACnet device instance.

Example – URL: (HTTP POST)
/rest/method/fieldpop-api/callDeviceMethod?happn_token=eyJ0eXAiOiJKV1QiLC

Example – JSON body:

```
{
  "parameters": {
    "deviceID": "gateway_1",
    "methodName": "getBACnetDeviceInfo",
    "deviceInstance": 200
  }
}
```

Example – JSON Response:

```
{
  "message": "Call successful",
  "data": {
    "DNET": 59915,
    "MAC": "c0a86409bac0"
  },
  "error": null
}
```

2.3.5.3 CallDeviceMethod – GetBACnetDeviceInfo (v.2)

Description: Get BACnet device info by sending a 'who-is' request out on BACnet.

Device API Changes: In v2 response uses DADR terminology instead of MAC in v1.

HTTP Verb: GET or POST

URL: /rest/method/fieldpop-api/callDeviceMethod

Parameters:

- **happn_token** – token obtained from the login request.
- **deviceAPIVersion** – see **Section 2.3.5**.
- **deviceID** – the IIoT gateway ID (see **Appendix A.1.1**).
- **methodName** – "getBACnetDeviceInfo"
- **deviceInstance** – the BACnet device instance.

Example – URL: (HTTP POST)
/rest/method/fieldpop-api/callDeviceMethod?happn_token=eyJ0eXAiOiJKV1QiLC

Example – JSON body:

```
{
  "parameters": {
    "deviceID": "gateway_1",
    "deviceAPIVersion": "2",
    "methodName": "getBACnetDeviceInfo",
    "deviceInstance": 200
  }
}
```

Example – JSON Response:

```
{
  "message": "Call successful",
  "data": {
    "DNET": 59915,
    "DADR": "c0a86409bac0"
  },
  "error": null
}
```

2.3.5.4 CallDeviceMethod – GetBACnetPropertyValue (v.1, v.2)

Description: Get BACnet property value by sending a 'readProperty' request out on BACnet.

Device API Changes: In v2 responses are formatted according to ANSI/ASHRAE Standard 135-2016 (minor changes including property name changes and case changes) instead of ANSI/ASHRAE Standard 135-2012 in v1.

HTTP Verb: GET or POST

URL: /rest/method/fieldpop-api/callDeviceMethod

Parameters:

- **happn_token** – token obtained from the login request.
- **deviceAPIVersion** – see **Section 2.3.5**.
- **deviceID** – the IIoT gateway ID (see **Appendix A.1.1**).
- **methodName** – "getBACnetPropertyValue"
- **deviceInstance** – the BACnet device instance.
- **objectType** – the BACnet object type.
- **objectInstance** – the BACnet object instance.
- **propertyIdentifier** – the BACnet property identifier.

Example – URL: (HTTP POST)
/rest/method/fieldpop-api/callDeviceMethod?happn_token=eyJ0eXAiOiJKV1QiLC

Example – JSON body:

```
{
  "parameters": {
    "deviceID": "gateway_1",
    "deviceAPIVersion": "2",
    "methodName": "getBACnetPropertyValue",
    "deviceInstance": 200,
    "objectType": "analog-value",
    "objectInstance": 0,
    "propertyIdentifier": "present-value"
  }
}
```

Example – JSON Response:

```
{
  "message": "Call successful",
  "data": {
    "value": 27,
    "type": "REAL"
  },
  "error": null
}
```

2.3.5.5 CallDeviceMethod – SetBACnetPropertyValue (v.1, v.2)

Description: Set BACnet property value by sending a 'writeProperty' request out on BACnet.

HTTP Verb: POST

URL: /rest/method/fieldpop-api/callDeviceMethod

Parameters:

- **happn_token** – token obtained from the login request.
- **deviceID** – the IIoT gateway ID (see **Appendix A.1.1**).
- **deviceAPIVersion** – see **Section 2.3.5**.
- **methodName** – "setBACnetPropertyValue"
- **deviceInstance** – the BACnet device instance.
- **objectType** – the BACnet object type.
- **objectInstance** – the BACnet object instance.
- **propertyIdentifier** – the BACnet property identifier.
- **propertyValue** – the value to write.
- **priority** (optional) – priority to set from 1 to 16.

Example – URL: (HTTP POST)
/rest/method/fieldpop-api/callDeviceMethod?happn_token=eyJ0eXAiOiJKV1QiLC

Example – JSON body:

```
{
  "parameters": {
    "deviceID": "gateway_1",
    "deviceAPIVersion": "2",
    "methodName": "getBACnetPropertyValue",
    "deviceInstance": 200,
    "objectType": "analog-value",
    "objectInstance": 0,
    "propertyIdentifier": "present-value",
    "propertyValue": {
      "type": "REAL",
      "value": 444
    },
    "priority": 16
  }
}
```

Example – JSON Response:

```
{
  "message": "Call successful",
  "data": null,
  "error": null
}
```

2.4 User Methods

2.4.1 GetUserInfo

Description: Retrieve info about currently authenticated user, based on token

HTTP Verb: GET

URL: /rest/method/fieldpop-api/getUserInfo

Query String Parameters:

- **happn_token** - Token obtained from login request.

Example GET URL: /rest/method/fieldpop-api/getUserInfo?happn_token=eyJ0eXAiOiJKV1QiLC

Example JSON Response:


```
{
  "message": "Call successful",
  "data": {
    "username": "admin@gmail.com",
    "firstName": "Joe",
    "lastName": "Bloggs",
    "email": "admin@gmail.com",
    "phoneNumber": "123456789",
    "oem": "test oem",
    "company": "test company",
    "role": "OEM Admin"
  },
  "error": null
}
```


2.4.2 GetUserDevices

Description: Retrieve list of devices visible to current user

HTTP Verb: GET or POST

URL: /rest/method/fieldpop-api/getUserDevices

Query String Parameters:

- **happn_token** - Token obtained from login request.
- **List** (optional) - set to true to get array of deviceID strings only.

Body Parameters (JSON):

- **List** (optional) - set to true to get array of deviceID strings only.
- **Filter** (optional) - MongoDB-style query object to filter results see https://www.npmjs.com/package/sift.

Example 1 – GET URL: (no list parameter set)
/rest/method/fieldpop-api/getUserDevices?happn_token=eyJ0eXAiOiJKV1QiLC

Example 1 – JSON Response:


```
{
  "message": "Call successful",
  "data": [
    {
      "company": "SMC",
      "description": "ProtoAir monitoring US and SA weather",
      "deviceId": "flickerpalm_N1ytNF-y7",
      "location": "-26.13226, 27.90566000000001",
      "macAddress": "00:50:4E:60:00:16",
      "name": "WeatherLink SA",
      "oem": "SMC",
      "state": "Normal",
      "packageInfo": {
        "productName": "System View Template",
        "customerName": "SMC",
        "productVersion": "0.4.1",
        "deviceAPIVersion": [
          "1.2.3",
          "2.3.4"
        ],
        "moduleVersions": {
          "ae": {
            "name": "@smc/field-pop",
            "version": "3.2.0"
          },
          "pe": {
            "Build_Revision": "unknown",
            "Driver_Configuration": "unknown",
            "FieldServer_Model": "unknown",
            "Carrier_Type": "unknown"
          },
          "fsweb": {
            "name": "unknown",
            "version": "unknown"
          },
          "os": "unknown"
        }
```

```
    }
  },
  {
    "company": "SMC",
    "description": "---",
    "deviceId": "cybersparrow_NkWMIrKbyQ",
    "location": "Access Not Permitted",
    "macAddress": "00:50:56:C0:00:01",
    "name": "---",
    "oem": "unknown",
    "state": "Alarm",
    "packageInfo": {
      "productName": "unknown",
      "customerName": "unknown",
      "productVersion": "unknown",
      "moduleVersions": {
        "ae": {
          "name": "@smc/field-pop",
          "version": "3.2.0"
        },
        "pe": {
          "Build_Revision": "unknown",
          "Driver_Configuration": "unknown",
          "FieldServer_Model": "unknown",
          "Carrier_Type": "unknown"
        },
        "fsweb": {
          "name": "unknown",
          "version": "unknown"
        },
        "os": "unknown"
      }
    }
  }
  ],
  "error": null
}
```


Example 2 – GET URL: (list parameter set)
/rest/method/fieldpop-api/getUserDevices?happn_token=eyJ0eXAiOiJKV1QiLC&list=true

Example 2 – JSON Response:


```
{
  "message": "Call successful",
  "data": [
    "flickerpalm_N1ytNF-y7",
    "cybersparrow_NkWMIrKbyQ"
  ],
  "error": null
}
```


Example 3 – POST URL: (list and filter specified)
/rest/method/fieldpop-api/getUserDevices?happn_token=eyJ0eXAiOiJKV1QiLC

Example 3 – JSON Payload:

```
{
  "parameters": {
    "list": true,
    "filter": {
      "state": {
        "$ne": "Normal"
      }
    }
  }
}
```

Example 3 – JSON Response:


```
{
  "message": "Call successful",
  "data": [
    "cybersparrow_NkWMIrKbyQ"
  ],
  "error": null
}
```


Example 4 (sift filter for single deviceID)
/rest/method/fieldpop-api/getUserDevices?happn_token=eyJ0eXAiOiJKV1QiLC

Example 4 – JSON Payload: (deviceId specified – note the capitalization of Id)

```
{
  "parameters": {
    "list": false,
    "filter": {
      "deviceId": {
        "$eq": "thundercentaur_MEc"
      }
    }
```

```
  }
}
```


Example 4 – JSON Response:


```
{
  "message": "Call successful",
  "data": [
    {
      "company": "SMC",
      "description": "http://192.168.100.180",
      "deviceId": "thundercentaur_MEc",
      "location": "-18.766947, 46.869106999999985",
      "macAddress": "00:50:4E:60:05:4E",
      "name": "HS-LTT-180",
      "oem": "SMC",
      "state": "Normal",
      "packageInfo": {
        "productName": "BACnet IoT Gateway",
        "customerName": "Sierra Monitor Corporation",
        "productVersion": "5.0.0",
        "deviceAPIVersion": [
          "2.0.0"
        ],
        "moduleVersions": {
          "ae": {
            "name": "@smc/bacnet-explorer",
            "version": "6.0.0"
          },
          "pe": {
            "Build_Revision": "4.30.2",
            "Driver_Configuration": "DCC000",
            "FieldServer_Model": "ProtoAir-WLAN485",
            "Carrier_Type": "-"
          },
          "fsweb": {
            "name": "web_default",
            "version": "3.2.1"
          },
          "os": "2.3.0"
        },
        "cpu": "armv7",
        "platform": "ProtoAir_ARMv7",
        "platform-revision": 1
      },
      "deviceOnline": true
    }
  ],
  "error": null
}
```

2.4.3 ListDeviceUsers

Description: Retrieve a list of users who can see the specified device

HTTP Verb: GET

URL: /rest/method/fieldpop-api/listDevicesUsers

Query String Parameters:

- **happn_token** - Token obtained from login request.
- **deviceID** - the IIoT gateway ID (see **Appendix A.1.1**)

Example GET URL:
/rest/method/fieldpop-api/getUserInfo?happn_token=eyJ0eXAiOiJKV1QiLC&deviceID=cybersparrow_NkWMIrKbyQ

Example JSON Response:


```
{
  "message": "Call successful",
  "data": [
    "admin@gmail.com",
    "user@company.com"
  ],
  "error": null
}
```

2.5 ProtoCast Methods

2.5.1 CreateOEMUser

Description: Creates a new OEM User account under the provided OEM

HTTP Verb: GET

URL: /rest/method/fieldpop-api/createOemUser

Query String Parameters:

- **happn_token** - Token obtained from login request. Contact MSA Safety support for credentials with access this method.
- **email** - the email address of the user for which an account should be created.
- **company** - the name of the SMC Cloud OEM under which the account should be created.

Example GET URL:
/rest/method/fieldpop-api/createOemUser?email=joe@soap.com&company=OEM_A&happn_token=eyJ0eXAiOiJKV1QiLC

Example JSON Response:

```
{
  "message": "Call successful",
  "data": "FieldPoP account requested. Please check your email for further action.",
  "error": null
}
```

Example Error Response:

```
{
  "message": "Call failed",
  "data": null,
  "error": {
    "message": "Invalid product company. Your FieldServer provider [OEM_A] has not yet registered with FieldPoP."
  }
}
```

2.5.2 GetFieldCastOEMList

Description: Returns a list of OEMs supported by the ProtoCast app

HTTP Verb: GET

URL: /rest/method/fieldpop-api/getFieldCastOemList

Query String Parameters:

- **happn_token** - Token obtained from login request. Contact MSA Safety support for credentials with access this method.

Example GET URL:
/rest/method/fieldpop-api/getFieldCastOemList?happn_token=eyJ0eXAiOiJKV1QiLC

Example JSON Response:

```
{
  "message": "Call successful",
  "data": [
    {
      "name": "Sierra Monitor Corporation",
      "display_name": "SMC"
    }
  ],
  "error": null
}
```

2.5.3 GetDeviceData

Description: Retrieve the data available under the specified ProtoCast device.

HTTP Verb: POST

URL: /rest/method/fieldpop-api/getDeviceData

Parameters:

- **happn_token** - token obtained from the login request.
- **deviceID** - the IIoT gateway ID (see **Appendix A.1.1**).

Example GET URL:
/rest/method/fieldpop-api/getDeviceData?happn_token=eyJ0eXAiOiJKV1QiLC

Example 3 – JSON Payload:

```
{
  "parameters": {
    "deviceID": "FB4CC9BD94ADE5D"
  }
}
```

Example JSON Response:


```
{
  "message": "Call successful",
  "data": [
      {
          "ProtoCast": {
              "type": "ProtoCast",
              "name": "Socomec",
              "path": "/Fieldcast/devices/Socomec",
              "status": "Normal",
              "status_class": "status_normal",
              "events": {
                  "alarm": false,
                  "warning": false,
                  "trouble": false
              },
              "points_count": 2,
              "Volts (V)": 123.9,
              "Current (mA)": 120
          }
      }
          ],
  "error": null
}
```

2.5.4 GetDeviceDataWithTime

Description: Retrieve the data available under the specified ProtoCast device with time stamp.

HTTP Verb: POST

URL: /rest/method/fieldpop-api/getDeviceDataWithTime

Parameters:

- **happn_token** - token obtained from the login request.
- **deviceID** - the IIoT gateway ID (see **Appendix A.1.1**).

Example GET URL:
/rest/method/fieldpop-api/getDeviceDataWithTime?happn_token=eyJ0eXAiOiJKV1QiLC

Example 3 – JSON Payload:

```
{
  "parameters": {
    "deviceID": "FB4CC9BD94ADE5D"
  }
}
```

Example JSON Response:


```
{
  "message": "Call successful",
  "data": [
    {
        "ProtoCast": {
              "type": "ProtoCast",
              "name": "Socomec",
              "path": "/Fieldcast/devices/Socomec",
              "status": "Normal",
              "status_class": "status_normal",
              "events": {
                  "alarm": false,
                  "warning": false,
                  "trouble": false
              },
              "points_count": 2,
              "Volts (V)": {
                  "value": 123.55,
                  "time": 1550187148
              },
              "Current (mA)": {
                  "value": 130,
                  "time": 1550187305
              }
          }
    }
          ],
  "error": null
}
```

2.6 Firmware Methods

2.6.1 GetDeviceFirmwareVersions

Description: Returns a list of available firmware versions for a given IIoT gateway

HTTP Verb: GET or POST

Query String Parameters:

- **happn_token** - Token obtained from the login request.
- **deviceID** - the IIoT gateway ID (see **Appendix A.1.1**).

Body Parameters (JSON):

- **parameters** - object containing the following fields:
  - **deviceID** - the IIoT gateway ID (see **Appendix A.1.1**).

Example 1 – GET URL:
/rest/method/fieldpop-api/getDeviceFirmwareVersions?happn_token=eyJ0eXAiOiJKV1QiLC&deviceID=cybersparrow_NkWMIrKbyQ

Example 1 – JSON Response:


```
{
  "message": "Call successful",
  "data": {
    "SMC": {
      "SMC ProtoCast": {
        "model": "SMC ProtoCast",
        "versions": [
          "5.4.0",
          "5.4.1",
          "5.4.2"
        ]
      }
    }
  },
  "error": null
}
```


Example 3 – POST URL:
/rest/method/fieldpop-api/getDeviceFirmwareVersions?happn_token=eyJ0eXAiOiJKV1QiLC

Example 3 – JSON Payload:


```
{
  "parameters": {
    "deviceID": "cybersparrow_NkWMIrKbyQ"
  }
}
```


Example 3 – JSON Response:


```
{
  "message": "Call successful",
  "data": {
    "SMC": {
      "SMC ProtoCast": {
        "model": "SMC ProtoCast",
        "versions": [
          "5.4.0",
          "5.4.1",
          "5.4.2"
        ]
      }
    }
  },
  "error": null
}
```

2.6.2 UpgradeFirmware

Description: Initiates a firmware upgrade for a given deviceID and version

HTTP Verb: POST

URL: /rest/method/fieldpop-api/upgradeFirmware

Query String Parameters:

- **happn_token** - Token obtained from the login request.

Body Parameters (JSON):

- **parameters** - object containing the following fields:
  - **deviceID** - the IIoT gateway ID (see **Appendix A.1.1**)
  - **firmwareVersion** - the semantic versioning (or semver) of firmware to load on the IIoT Gateway

Example 3 – POST URL:
/rest/method/fieldpop-api/upgradeFirmware?happn_token=eyJ0eXAiOiJKV1QiLC

Example 3 – JSON Payload:

```
{
  "parameters": {
    "deviceID": "cybersparrow_NkWMIrKbyQ",
    "firmwareVersion": "5.4.2"
  }
}
```

Example 3 – JSON Response:

```
{
  "message": "Call successful",
  "data": null,
  "error": null
}
```

Appendix A. Additional Information

Appendix A.1. General Notes

Appendix A.1.1. How to Obtain an IIoT Device ID (DeviceID Parameter Value)

This can be obtained from:

- LAN access to the local SMC Cloud GUI on the device (e.g. http://192.168.1.20/fieldpop/client/).
- SMC Cloud tunnel access to the SMC Cloud GUI on the device.
- From the SMC Cloud website, by clicking on the device pin on the Device Management map.

Appendix A.1.2. Device API Version

The API available to communicate directly to a device is versioned by the "deviceAPIVersion" parameter. A device may support multiple device API versions at once (e.g. version 1 and 2 being supported). The highest supported device API versions on the device is specified as an array of semver formatted strings, e.g. ["1.2.3", "2.3.4"] – it can be obtained by:

- Viewing the "About" page on the device.
- Doing a getUserDevices request – under the "packageInfo" property of the response per device – see **Section 2.4.2**.

The desired "deviceAPIVersion" should be specified as part of the request:

- Example: /rest/method/fieldpop-api/getDeviceData?happn_token=eyJ0eXAiOiJKV1QiLC&deviceAPIVersion=2
- If JSON body parameters are used in the request, then deviceAPIVersion should also specified in the JSON body, and not as a query parameter as shown in the example.
- If not specified, request version 1 is assumed.
- If deviceAPIVersion=2 is specified, the requests will only be sent to devices with supported versions greater than or equal to 2 and less than 3.
- If deviceAPIVersion=2.3 is specified, the requests will only be sent to devices with supported versions greater than or equal to 2.3 and less than 3.
- If deviceAPIVersion=2.3.4 is specified, the requests will only be sent to devices with supported versions greater than or equal to 2.3.4 and less than 3.

Appendix A.1.3. Data Retention

The RESTful API will provide access to all data retained on the SMC Cloud. The duration for which data is retained on the SMC Cloud is an operation decision outside the scope of this document.

Appendix A.1.4. Data Structure

All data will be returned in a self-describing JSON format.

Appendix A.2. Using CURL to Generate SMC Cloud RESTful API Requests

When using CURL to generate REST API requests, the URL must be in quotes when specifying URL parameters. Here are some examples:

deviceDataLog

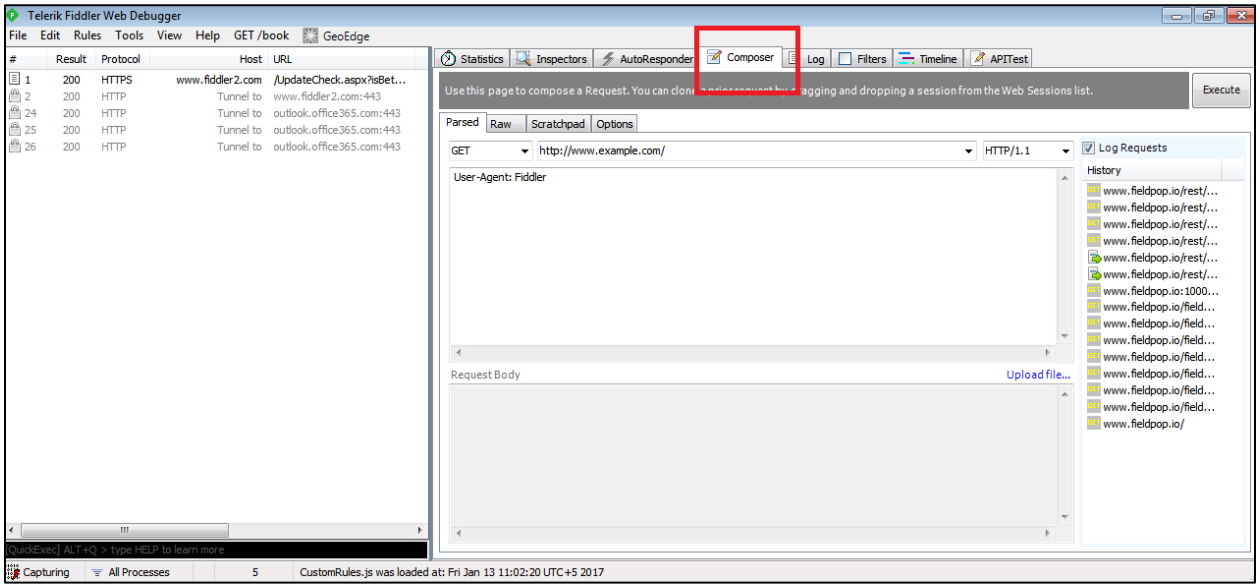
curl -v -L 'https://www.fieldpop.io/rest/method/fieldpop-api/deviceDataLog?deviceID=enter-device-id-here&happn_token=enter-token-here'

getDeviceData

curl -v -L -H "Content-Type: application/json" -X POST -d '{"parameters": {"deviceID": "enter-device-id-here","path": "1001/analog-input:1/present-value","options": {"asn1": true}}}' 'https://www.fieldpop.io/rest/method/fieldpop-api/getDeviceData?happn_token=enter-token-here'

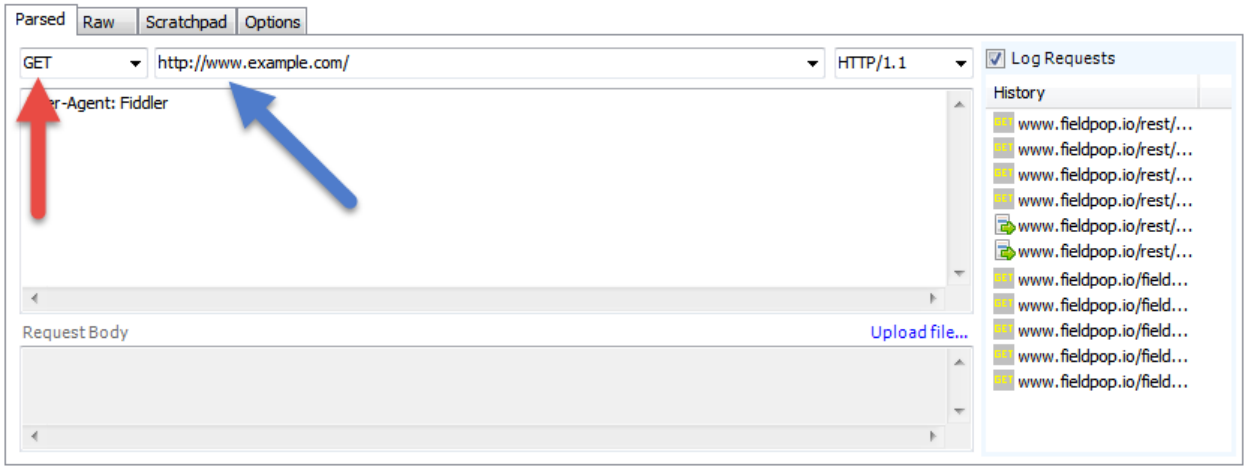
Appendix A.3. How to Access the RESTful API Through Fiddler

1. Install Fiddler from URL: https://www.telerik.com/download/fiddler
2. Open Fiddler4.
3. Click on the Composer tab.

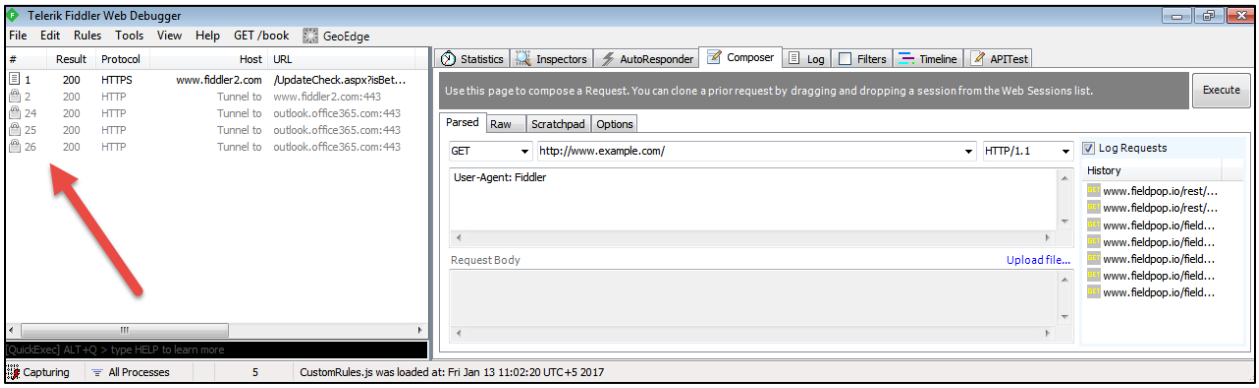


4. Click on the Parsed tab.
5. Set the HTTP verb to GET from the dropdown menu and enter the following HTTPS address: https://www.fieldpop.io/rest/login?username=<*username*>&password=<*password*>

NOTE: For the italisized portions of the address above ("<username>" and "<password>"), the authenticated username and password must be added. No brackets should be included ("<" or ">").



6. Before pressing the Execute button, check the left most window for logs of all URL's and clear them if any exist.

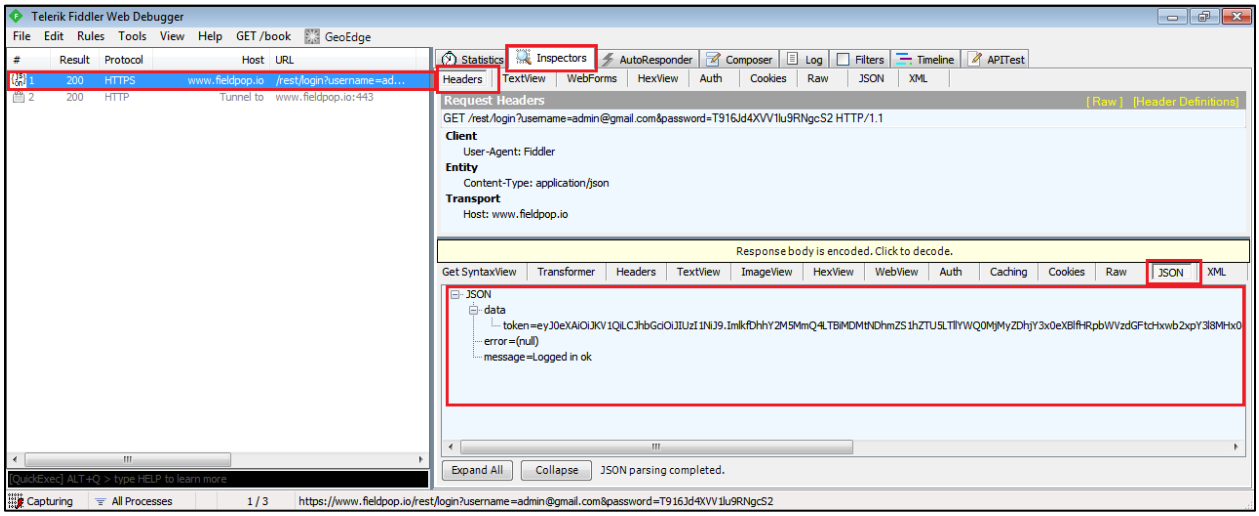


7. Click on the Execute button.

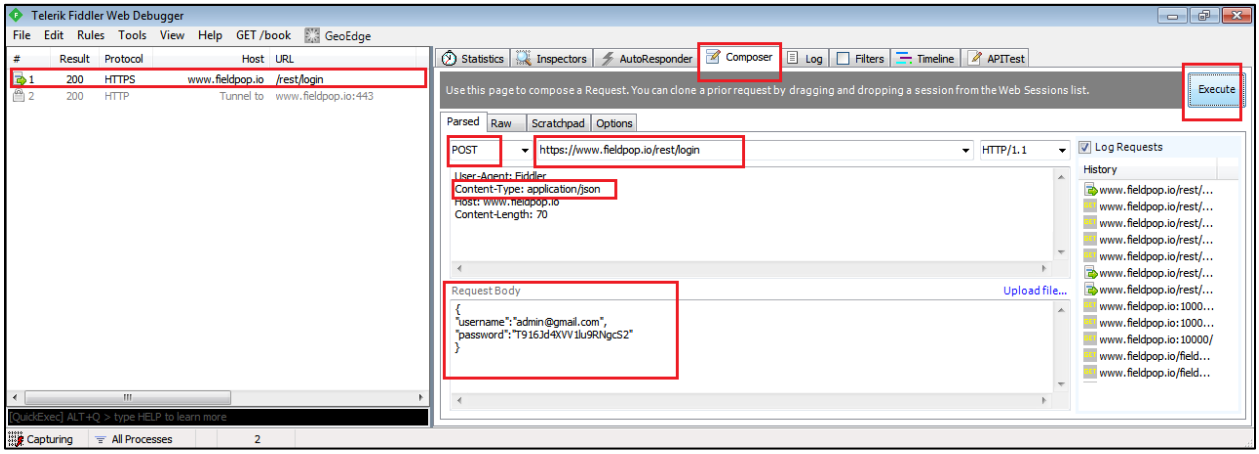
NOTE: The HTTP response is logged on the left part of the window. HTTP code 200 indicates the call is successful.

8. Double click on the URL logged on the left side of the screen.
9. Select the Inspector tab.
10. Select the Header sub-tab.
11. Select the JSON box below to see the JSON data response from the RESTful API.
12. Record the data.token value.

NOTE: The token is the authentication token needed to get the device data logs.



13. Go back to the Composer-Parse tab and set the HTTP verb to POST from the dropdown menu.
14. Enter the URL of the REST endpoint.
15. Enter the attribute “Content-Type: application/json” in the request header.
16. Enter the input to be passed to the RESTful service in JSON format.
17. Click on the “Execute” button to make a POST request.



18. Double click the URL logged.
19. Click on the Inspectors tab.
20. Select JSON to see the response in JSON format.
21. Copy the token.
22. Compose the URL again to get the device data logs (https://www.fieldpop.io/rest/method/fieldpop-api/deviceDataLog).
23. Pass the copied token as a query parameter (deviceID=*deviceID*&happn_token=*authtoken*&startUTCsec=*1477388259*&endUTCsec=*1500000000*).

Appendix A.4. Generic Information for Access to SMC Cloud RESTful API

NOTE: In the tables that follow, italicized characters indicate placeholder text that must be filled in by the user with the correct information.

Appendix A.4.1. Authenticate via GET Command

Method	GET
URL	https://www.fieldpop.io/rest/login?username=*username*&password=*password*
Request Header	Content-Type: application/json
Request Body	NA

Response	{
	message: 'Logged in ok',
	data: { token: tokenvalue},
	error: null
	}

Appendix A.4.2. Authenticate via POST Command

Method	POST
URL	https://www.fieldpop.io/rest/login
Request Header	Content-Type: application/json

Request Body	{
	“username” : “*username*”
	“Password” : “*password*”
	}

Response	{
	message: 'Logged in ok',
	data: { token: tokenvalue},
	error: null
	}

Appendix A.4.3. Run GET Command to Deliver Device Data Logs

Method	GET
URL	https://www.fieldpop.io/rest/method/fieldpop-api/deviceDataLog?deviceID=*deviceID*&happn_token=*authtoken*&startUTCsec=*1477388259*&endUTCsec=*1500000000*
Request Header	Content-Type: application/json
Request Body	NA

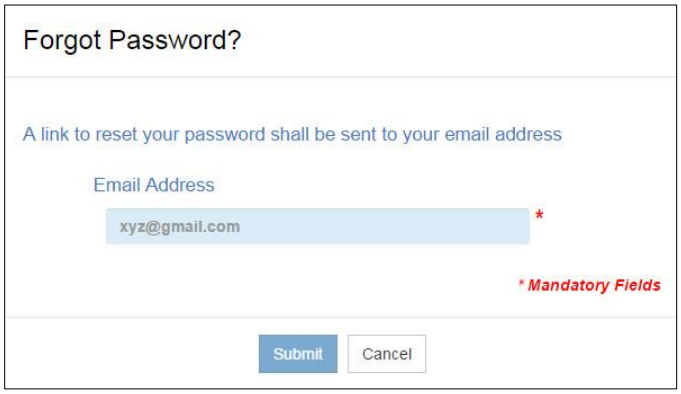
Response	{
	"message": "Call successful",
	"data": {},
	"error": null
	}

Appendix B. Troubleshooting

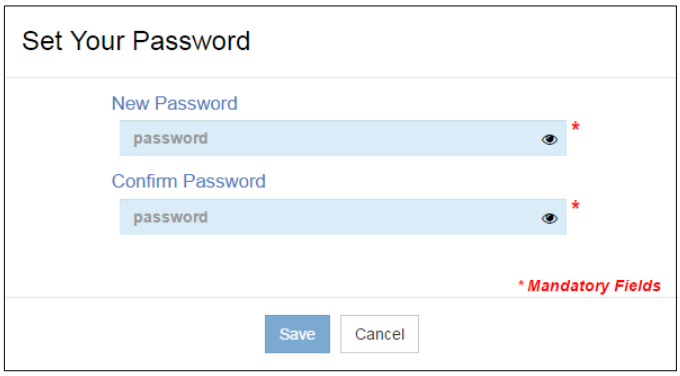
Appendix B.1. Lost SMC Cloud Login Password

If the password is lost, follow the below instructions:

- Click “Forgot Password” on the SMC Cloud Login Screen.



- Once the Password Reset Window appears, enter the email address of the SMC Cloud account and click Submit.
- Click on the “Reset Your Password” button in the email sent from notification@fieldpop.io to reset the SMC Cloud password.



- Enter and confirm the new password then click Save.

Appendix C. Useful Features

Appendix C.1. Security

SMC Cloud to FieldServer and FieldServer to browser connections are secured with HTTPS, which uses TLS/SSL (Transport Layer Security/Secure Sockets Layer). The HTTPS certificate is issued by SSL.com. Details are viewable via any local PC browser by following the instructions found in **Appendix C.1.3**.

NOTE: SMC Cloud keeps information private between individual OEMs and individual enterprise users. There is no bleed between different OEMs and different enterprise users.

Appendix C.1.1. PC to SMC Cloud

To browse SMC Cloud via PC, type the following into the PC internet browser: www.smccloud.net (port 80 and 443).

Appendix C.1.2. FieldServer to SMC Cloud

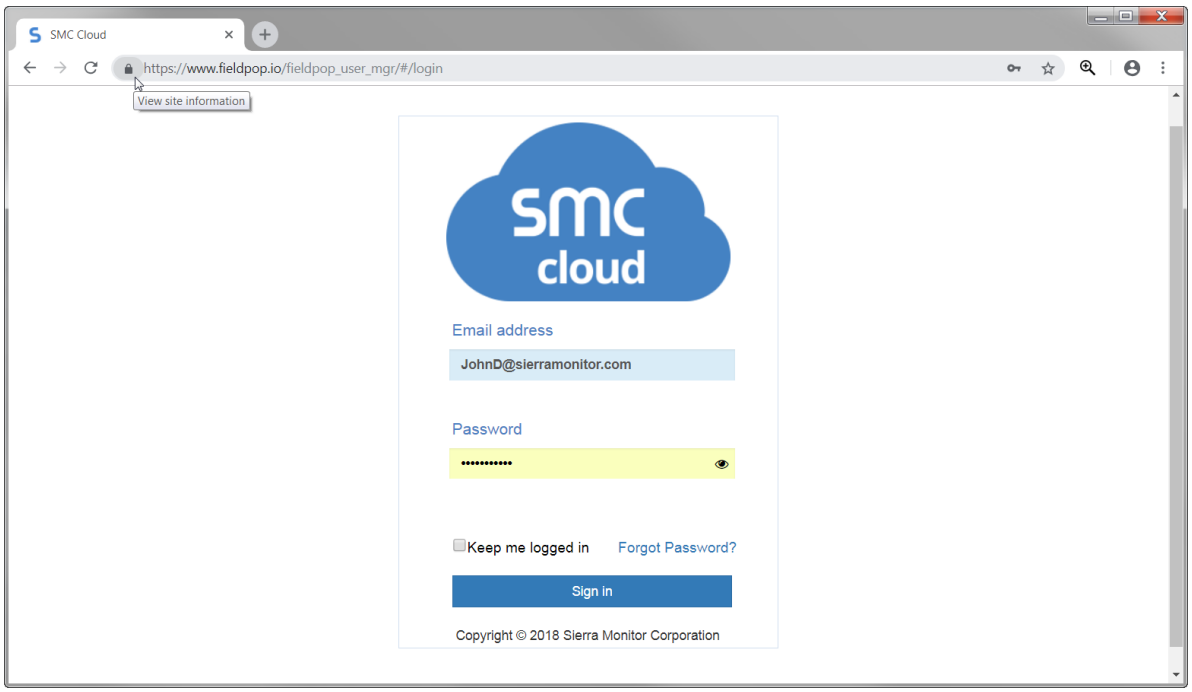
To allow the FieldServer to connect to SMC Cloud, use the following domain: www.fieldpop.io (port 80 and 443).

To connect to a ProtoNode via SMC Cloud, a device tunnel is created that has a unique subdomain in the URL. The best way to configure a firewall rule with this in mind is to use a wildcard domain: *.tunnel.fieldpop.io (port 443).

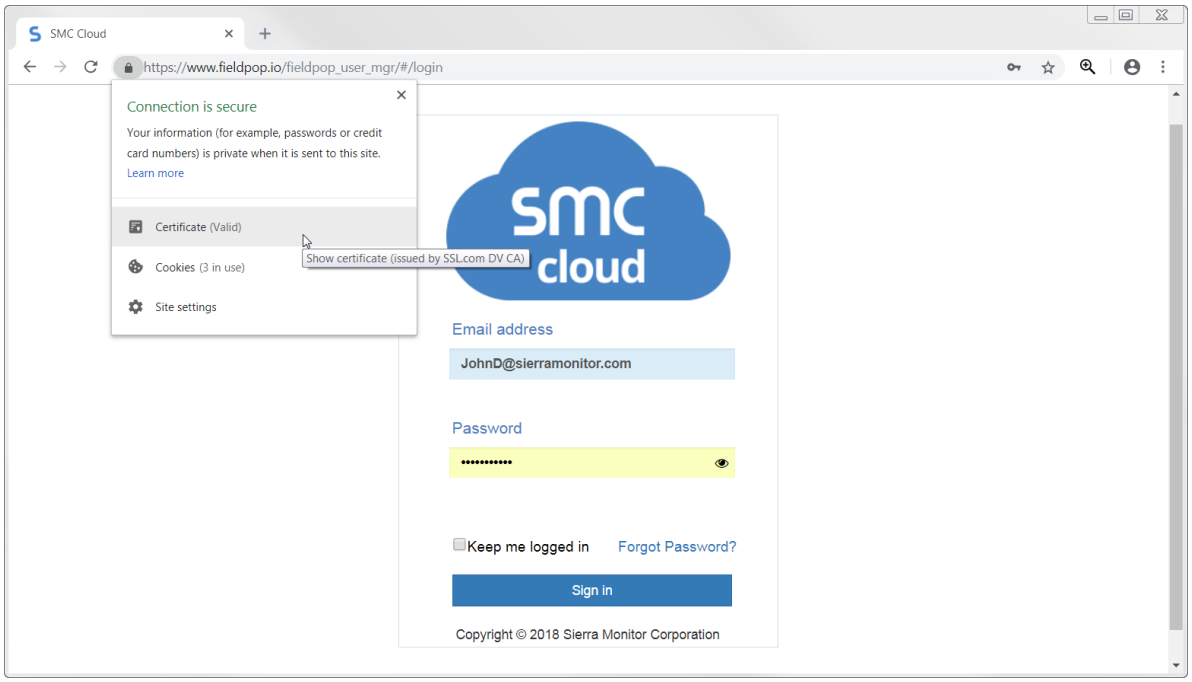
Additional security can be added by allowing the FieldServer to exclusively access the *.fieldpop.io. This provision can be set up in the customer's firewall.

Appendix C.1.3. Viewing the Certificate

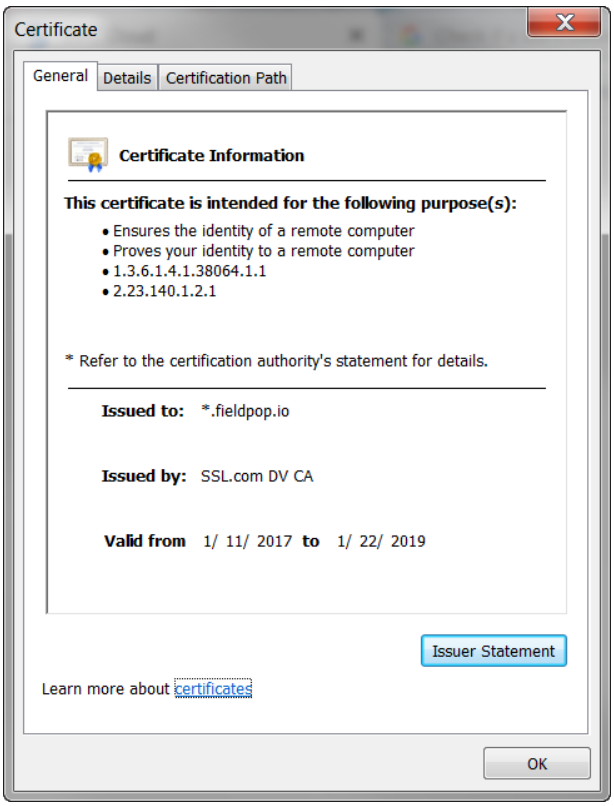
- Open a web browser on the local PC and go to www.smccloud.net.
- Move the cursor to the padlock icon (🔒) next to the website address.



- Click the padlock icon to open a dropdown menu for website information and browser settings.



- Review the information and click the Certificate button.



- Examine the certificate as needed.

NOTE: To download the certificate, click the Details tab and click the 'Copy to File' button.

Appendix D. Warranty

MSA Safety warrants its products to be free from defects in workmanship or material under normal use and service for two years after date of shipment. MSA Safety will repair or replace any equipment found to be defective during the warranty period. Final determination of the nature and responsibility for defective or damaged equipment will be made by MSA Safety personnel.

All warranties hereunder are contingent upon proper use in the application for which the product was intended and do not cover products which have been modified or repaired without MSA Safety's approval or which have been subjected to accident, improper maintenance, installation or application, or on which original identification marks have been removed or altered. This Limited Warranty also will not apply to interconnecting cables or wires, consumables or to any damage resulting from battery leakage.

In all cases MSA Safety's responsibility and liability under this warranty shall be limited to the cost of the equipment. The purchaser must obtain shipping instructions for the prepaid return of any item under this warranty provision and compliance with such instruction shall be a condition of this warranty.

Except for the express warranty stated above, MSA Safety disclaims all warranties with regard to the products sold hereunder including all implied warranties of merchantability and fitness and the express warranties stated herein are in lieu of all obligations or liabilities on the part of MSA Safety for damages including, but not limited to, consequential damages arising out of/or in connection with the use or performance of the product.