AN-020 IoT Radar Sensor with WiFi Interface

The OPS243 radar sensor with WiFi wireless connectivity is a complete IoT (Internet of Things) solution for traffic and object monitoring. All that is needed to access remote data is power for the sensor and a WiFi network to transport the data to the cloud for visualization, processing, or storage. The sensor is pre-configured to seamlessly pass data through the WiFi network to the cloud. This application note describes the use of the OPS243 network architecture, how to connect the sensor to a WiFi network, and visualizing the sensor data.

Network Architecture

The WiFi interface on OPS243 is designed for simple connection to a WiFi network and viewing of sensor data. A WiFi connection is made using the API of the OPS243 sensor. Alternatively, a Radar Sensor app for smartphones and tablets is available on the Google Playstore here. **Note this app only works with Andorid OS 12 and earlier.** An iOS app from the WiFi module provider is available as well here.

The overall network architecture is shown in Figure 1. The OPS243 uses the local WiFi network to pass data to an MQTT (Message Queuing Telemetry Transport) Broker in the cloud. The MQTT Broker is pre-configured by OmniPreSense to receive data from the sensor. Data from the sensor is published to the MQTT Broker as it becomes available and is passed on by the broker to any device which is a subscriber to that data. The owner of the sensor uses unique MQTT username and password to subscribe to the sensor data. In addition, API commands can be passed back to the sensor to configure it as desired or query its status.

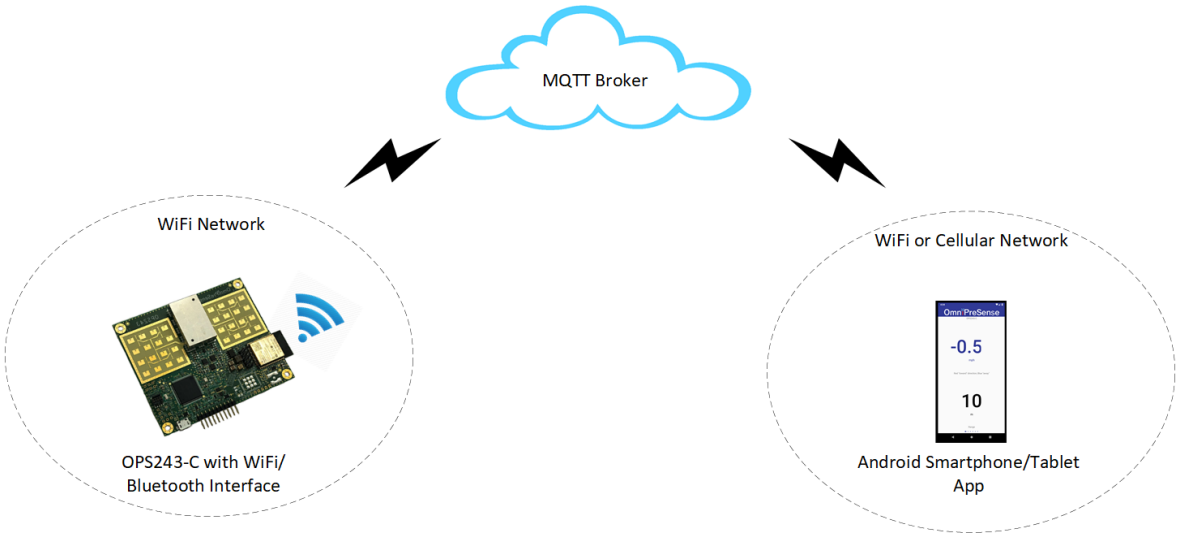


Figure 1. OPS243 Wireless Network Architecture

There are several methods for connecting a WiFi enabled sensor to a WiFi network. The newest method is via the standard API which now allows for scanning and connecting to a WiFi network. In addition, the legacy approach of using the OmniPreSense Android app is available along with the WiFi module provider iOS app. The Android app only works with Android OS 12 or earlier. Longer term this option will be replaced with a different app.

API WiFi Network Connection

Starting with OPS243-A code v1.2.4 and OPS243-C code v1.2.7, the API has been updated to scan and connect to a WiFi network.

The API provides several commands to enable the WiFi connection and query the WiFi status. The API commands are summarized in Table 1.

Table 1. WiFi Control API

Command	Name	R/W	Value
$V	WiFi Version	Write	Provides the WiFi code version. Response: {"name": "mqtt_blufi_uart","version": "v2.0.4","build":"Sep 22 2025 21:46:53"}
$G	WiFi Network	Write	Returns currently set WiFi network ssid and password {"wifi":"values","ssid":"OmniPreSense","password":"****"}
$W	WiFi Scan	Write	Returns list of top WiFi networks detected by order of signal strength. {"results":[{"ssid": "network1","channel":6,"rssi": -48}, {"ssid": "network2",….
$C=cccccc	WiFi Network SSID Select	Write	Set the desired WiFi network WiFi to join, cccccc=SSID. (max 32 characters) Response: {"values":{"ssid": "network1"}}
$P=pppppp	WiFi Network Password	Write	Set the password of the WiFi network to connect to, ppppp=password. Response: {"values":{"password":"xxxx"}}
$I	WiFi IP Status	Write	Returns the IP address assigned and the MAC address of the WiFi module. Use to confirm connection to WiFi network. Reponse: {"ip": {"MAC": "08A6F7AB8F6C", "address":"10.0.0.97","netmask":"255.255.255.0", "gateway":"10.0.0.1"}}

The process to connect the sensor to a WiFi network is as follows:

1. Scan for available WiFi networks with the $W command
2. Set the SSID for the desired network with the $C=ccccc command
3. Set the network password if required with the $P=ppppp command
4. Confirm connection with the $I command, showing an IP address has been assigned

In addition to these commands, the version of WiFi code on the sensor is returned with the $V command. Future updates will provide over-the-air (OTA) updates of the WiFi code and ability to change the MQTT broker that data is sent to.

Android App WiFi Network Connection

An OmniPreSense Android app is available for connecting the sensor to a WiFi network. The app is found on the Google Playstore here. As noted above, this app only works with Android OS 12 or older OS versions.

The steps to connect to a WiFi network are:

1. Open WiFi Radar Sensor app
2. Connect smartphone/tablet with OPS243 over Bluetooth
3. Scan and choose WiFi network
    a. Provide password if protected
4. Receive data within the app

A Bluetooth connection is only used for the initial connection and WiFi provisioning. Once connected into the WiFi network, all data is sent over the WiFi network from the sensor. Once connected, the user does not have to be within range of the sensor and can receive sensor data anywhere the app is on a device with internet connection (cellular, WiFi).

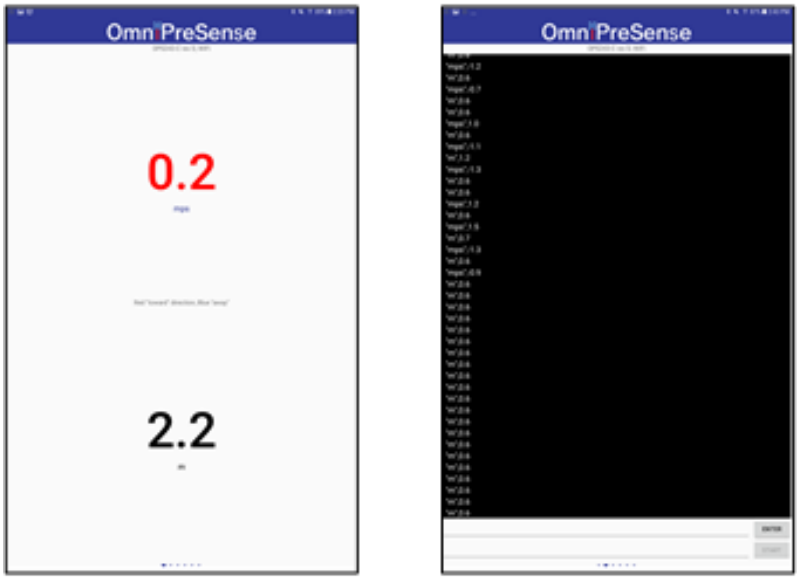
The following describes the connection sequence for the Radar Sensor app and the OPS243 WiFi enabled sensor.

1. First, make sure the OPS243 is powered up via USB or other connection.
2. Upon opening the app the first time, several introductory screens will provide an overview of the apps features. At the end, the app will provide the 5th screen for network connectivity (Figure *2*).

Figure 2. WiFi Radar Sensor App Bluetooth Pairing

3. The user should press the Scan Bluetooth button to scan for available Bluetooth devices. The radar sensor will be labeled OPS_RADAR_xxxx with xxxx the last digits of the MAC address. Select the OPS243 sensor and the Bluetooth connection will be completed.
4. Next, select and connect to the WiFi network. Press the Scan WiFi button and a list of available WiFi networks within range of the radar sensor will appear.
5. Select the desired WiFi network and if required, enter the password.

At the end of step 5, the OPS243 will be connected to the WiFi network of choice and start streaming data to the cloud. The data can be visualized on either the 1st or 2nd screen of the app (Figure 3).



1st Screen 2nd Screen

Figure 3. WiFi Radar Sensor App Data Visualization

The 1st screen provides the strongest signal speed and range information while the 2nd screen provides more detailed flow of data. The 2nd screen mimics a command terminal program like Putty and Teraterm and allows for entry of API commands at the bottom. Entering a command will be transferred back to the sensor which will adjust its output or processing accordingly.

iOS App WiFi Network Connection

An iOS app is available from the WiFi module supplier. The app is called EspBlufi and is available from the Apple store here (https://apps.apple.com/sg/app/espblufi/id1450614082).

The instructions for using the app are as follows:

1) Connect power to the OPS243 WiFi sensor and make sure it's within range of your desired WiFi network.
2) Download and install the EspBluFi app on your phone or tablet and make sure Bluetooth is enabled on your device.
3) Open up the app and you'll either see a mostly blank screen or a list of Bluetooth devices as in the figure below (Figure 4). If blank, use your finger to pull down on the screen to refresh or view the list of devices available. Locate the OPS_RADAR_xxxx in the list of devices and select it.

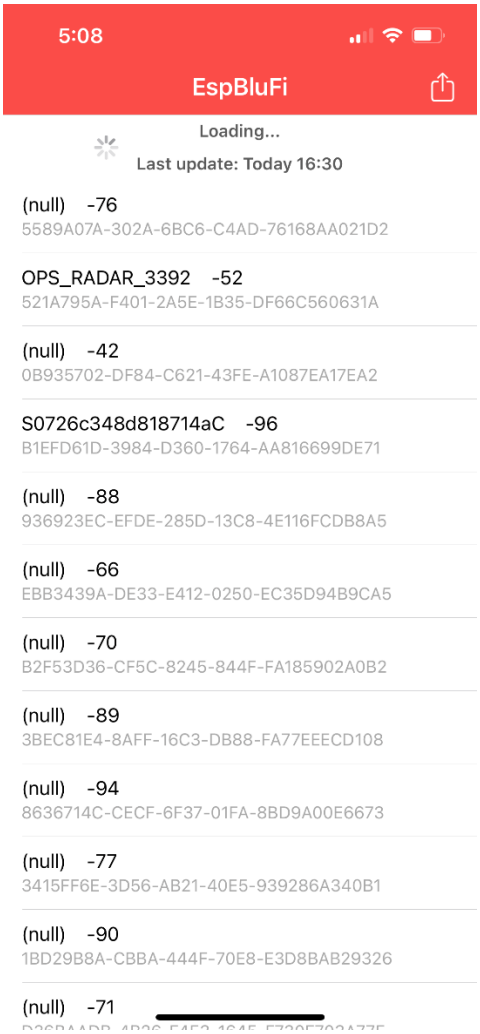


Figure 4. EspBlufi Start Screen with Bluetooth Devices

4) Press the Connect button in the lower left (Figure 5) to connect the app to your OPS243 WiFi sensor. Upon connecting, the other buttons on the app will change color and may be selected (right side Figure 5).

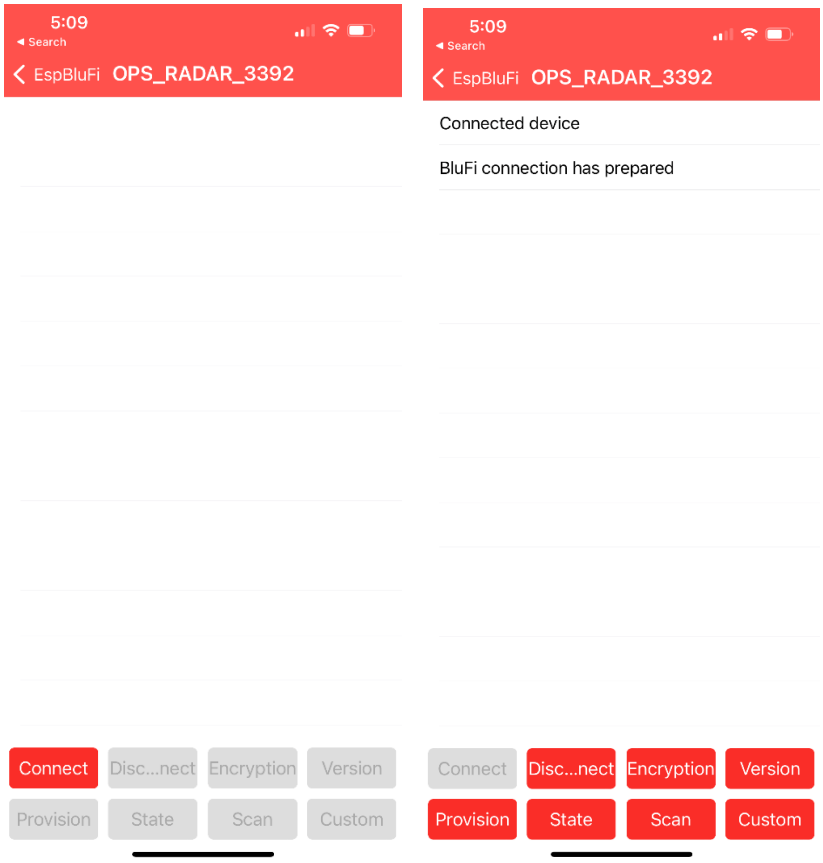


Figure 5. EspBlufi Connect Screen

5) If the app does not detect your sensor you may have to clear a filter in the app. As shown in Figure 6, press the box in the upper right corner to get the Setting pop-up and click on it.

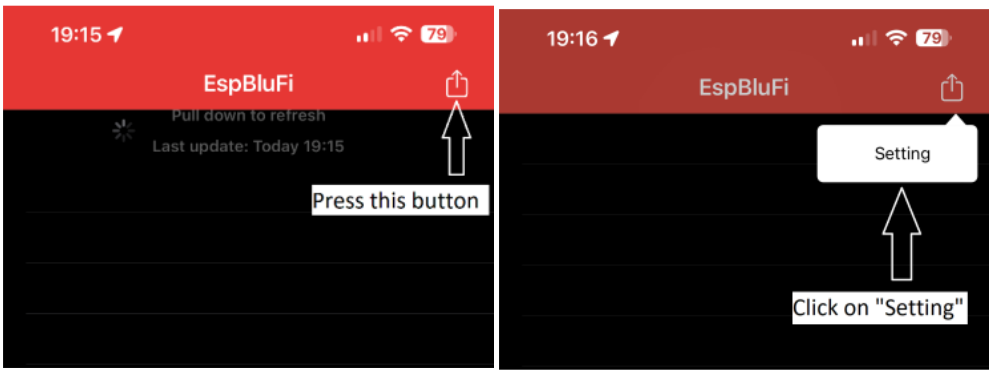


Figure 6. Clearing BluFi Bluetooth Filter

6) Next, a new screen as shown in Figure 7 will appear. Select BluFi and a pop window appears (right side Figure 7). Make sure this is clear and enter Ok. The cleared Setting will look like Figure 8.

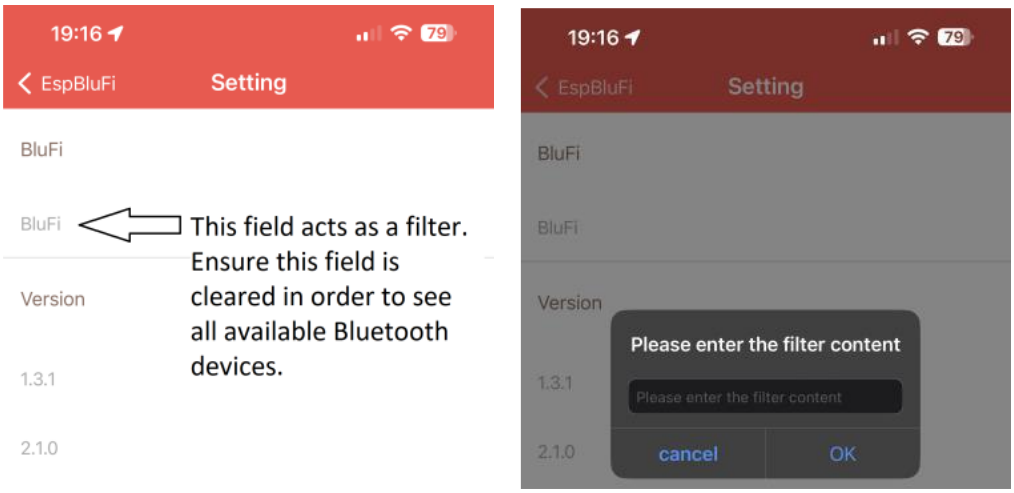


Figure 7. Clear BluFi Filter

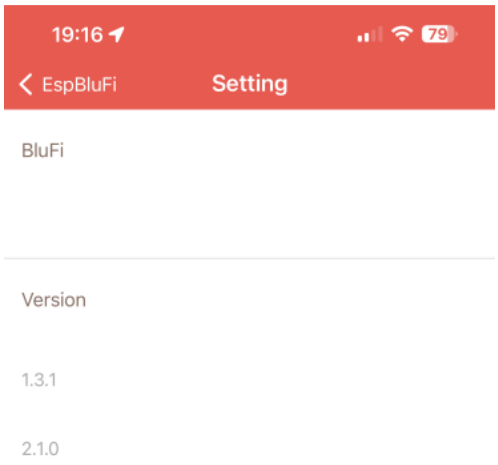


Figure 8. Cleared BluFi Filter

7) Next, select the Provision button on the lower left which will change to the screen show in Figure 9.
8) Press the Null next to OpMode and a scroll down menu will appear. Change to STA for station as in Figure 10 and select Ok.

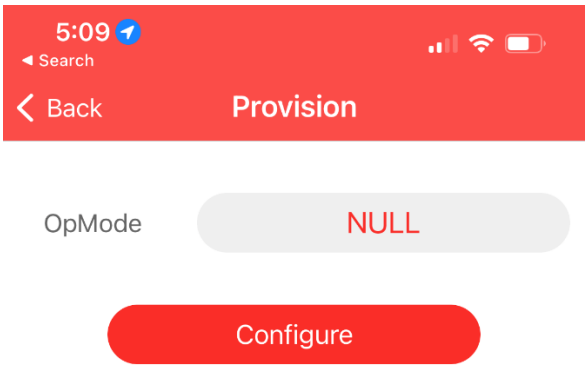


Figure 9. Provision Main Screen

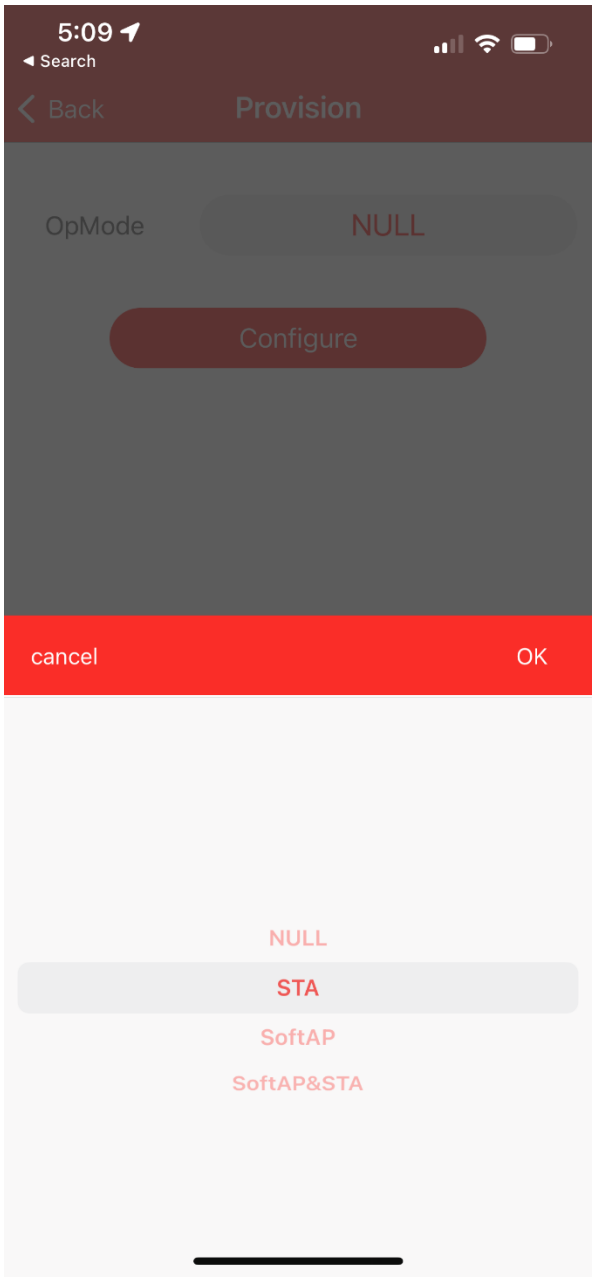


Figure 10. Provisioning STA (station)

9) At this point the screen will show where you can enter your network name and password (Figure 11) and hit Configure. Make sure they are spelled correctly. At this point the OPS243 WiFi sensor should be connected to your WiFi network. You can confirm the sensor is connected by pressing the State button (Figure 12).

   The sensor is factory programmed to send the data to the cloud via MQTT as discussed earlier and no further action is required. To see your data, you'll want to use MQTT X,

MQTT Explorer, or similar to see the data. See the prior section for how to make that type of connection.

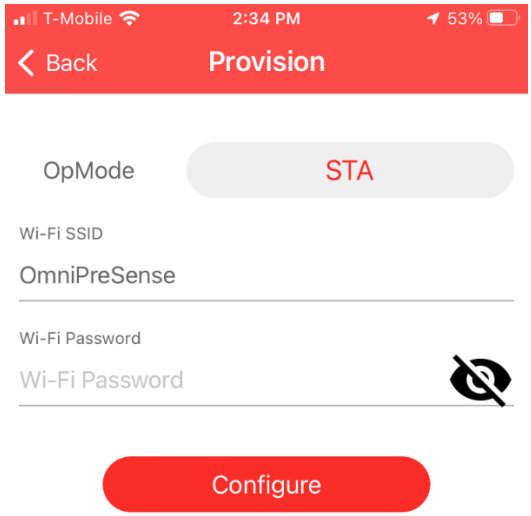


Figure 11. Provisioning WiFi Network Name and Password

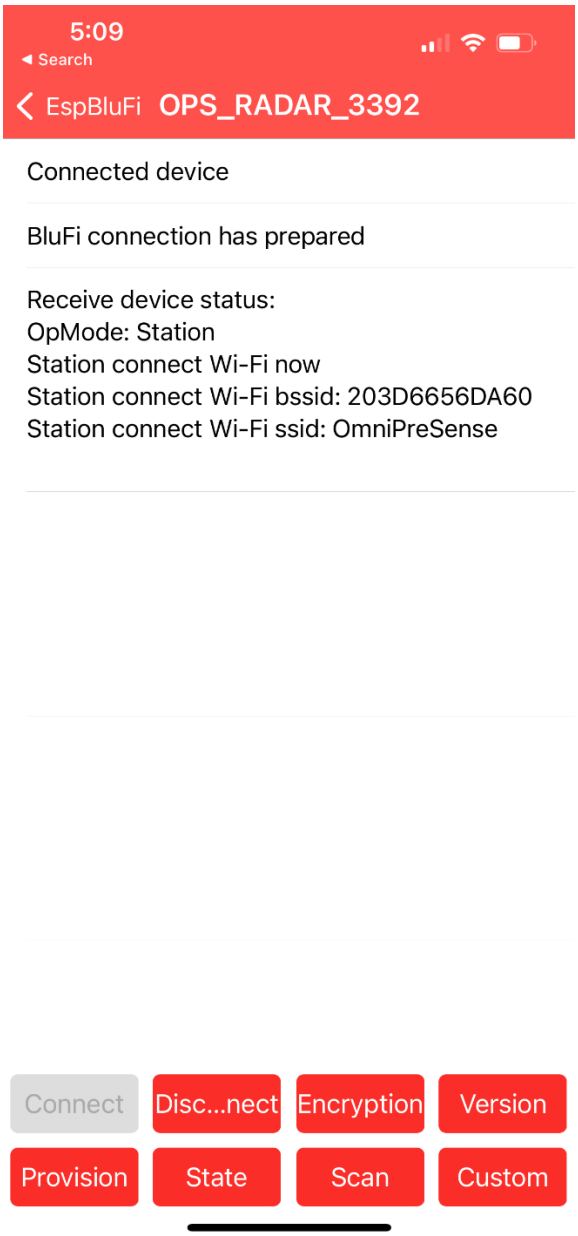


Figure 12. Connected OPS243 to WiFi Network

Data Visualization

Your OPS243 sensor data can be visualized on any device that can subscribe to MQTT data. A processor board can run code to accept the data for visualization, processing, or storage. A useful PC based tool is MQTT Explorer. The login screen for MQTT Explorer is shown in Figure 13. Contact OmniPreSense customer service to request the MQTT username and password for your sensor. Once connected, a new screen will show the data sent by the sensor and you're able to send API commands back to the sensor.

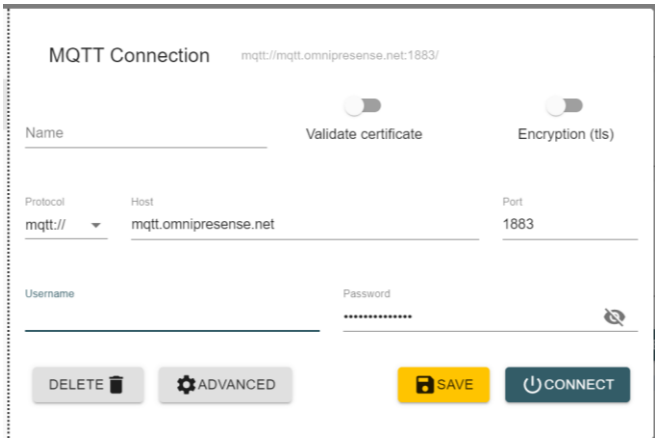


Figure 13. MQTT Explorer Log-In Screen

Revision History

Version	Date	Description
A	February 3, 2021	Initial release.
B	January 7, 2022	Added information for connecting to sensor with MQTT X.
C	February 15, 2023	Added information for using iOS BluFi app to connect OPS243 to WiFi network.
D	September 29, 2025	Added WiFi connection method using API in OPS243-A v1.2.4 and OPS243-C v1.2.7 code releases. Added updates on Android app usage. Added viewing using MQTT Explorer, removed MQTT X information.