

# Azure IoT Hub Integration User Guide

From 8.8 Aruba Controller and IAPs start to support Azure IoT Hub integration, so far we support BleData and serialData related IoT devices. the serialData support both south bound and north bound messages.

Our Aruba devices are using DPS (Device Provisioning Service) to get ithub connection info like IoT Hub URL, device ID, and device Key. which device connect to which IoT Hub is controlled by DPS, and the DPS works to add device into the related IoT Hub, which give more flexible to our customer to deploy IoT solution, and simplify the process and configuration model.

## Pre-preparation:

this User Guide assume user already configured DPS and IoT Hub, if not, please refer to Microsoft guide to create DPS and IoT Hub.

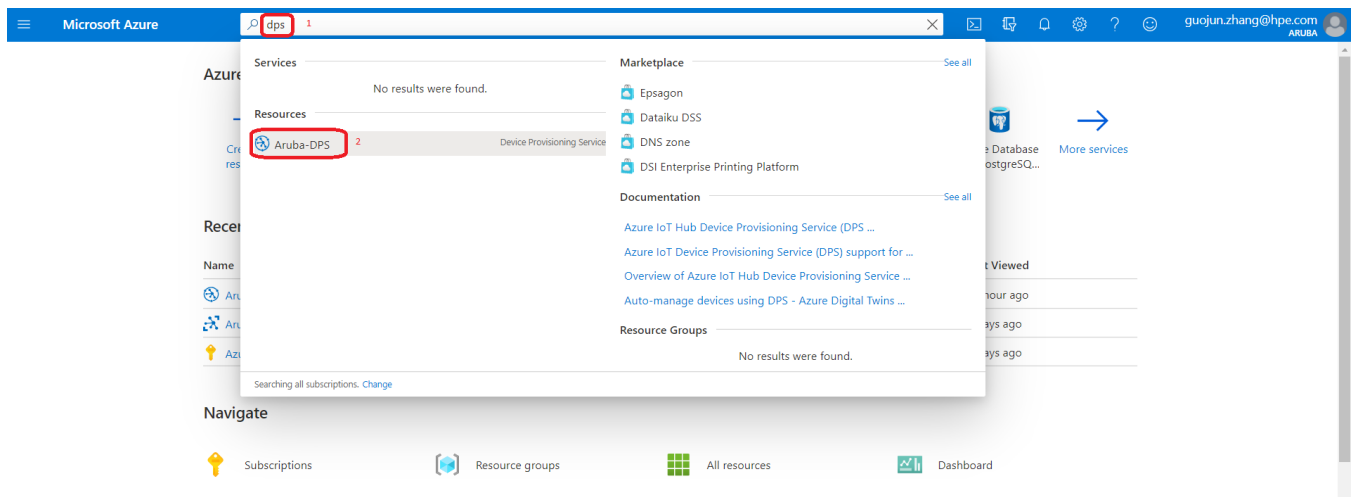
<https://docs.microsoft.com/en-us/azure/iot-dps/quick-setup-auto-provision>

And make sure your Controller/IAP are enabled NTP to get accurate clock. without the accurate clock the device will not be able to establish the connection with Azure IoT Hub.

## Create group device enrollment in the Azure DPS service:

Go to Azure portal <https://portal.azure.com>, you can search dps by enter the "dps" in the search bar, and you can see your configured DPS resource. otherwise you have to create your DPS service by previous pre-prepare step.

If the DPS is in your "recent resources" list, you can directly click it.



After enter the DPS service, in this example the DPS name is "Aruba-DPS", you need to record the "ID Scope" as highlight in below pictures. then click the left side "Manage enrollments" to start the device enrollment configuration.

Microsoft Azure

Search resources, services, and docs (G+ /)

Home > Aruba-DPS Device Provisioning Service

Search (Ctrl+/) Move Delete Refresh

Overview

- Activity log
- Access control (IAM)
- Tags
- Diagnose and solve problems
- Settings
  - Quick Start
  - Shared access policies
  - Linked IoT hubs
  - Certificates
  - Manage enrollments** 2
  - Manage allocation policy
  - Networking
  - Properties

Azure IoT Hub and the Azure Device Provisioning Service are updating their TLS certificates starting October 5, 2020 with a new Microsoft Certificate Authority (CA) chained under the existing Baltimore root. If your devices pin certificates, you may need to take action to ensure your devices can continue to connect. [Learn more](#)

### Essentials

Resource group (change)	: Kaushal-RG	Service endpoint	: Aruba-DPS.azure-devices-provisioning.net
Status	: Active	Global device endpoint	: global.azure-devices-provisioning.net
Location	: West Central US	ID Scope	: 0ne00190C2D 1
Subscription (change)	: Azure for Yogendra	Pricing and scale tier	: S1
Subscription ID	: 2bf1e338-5361-470d-bc8a-78c50b2b7f16		
Tags (change)	: <a href="#">Click here to add tags</a>		

### Quick Links

- [Azure IoT Hub Device Provisioning Service Documentation](#)
- [Learn more about IoT Hub Device Provisioning Service](#)

In the top of manage enrollments page, click the "Add enrollment group" to add a enrollment group for your group of device.

Microsoft Azure

Search resources, services, and docs (G+ /)

Home > Aruba-DPS

### Aruba-DPS | Manage enrollments

Device Provisioning Service

Search (Ctrl+/) + Add enrollment group + Add individual enrollment Refresh Delete

Overview

- Activity log
- Access control (IAM)
- Tags
- Diagnose and solve problems
- Settings
  - Quick Start
  - Shared access policies

You can add or remove individual device enrollments and/or enrollment groups from this page

### Enrollment Groups

Individual Enrollments

Search group enrollment by group name (name has to be exact match)

GROUP NAME
<input type="checkbox"/> Aruba-Group1

In the "Add Enrollment Group" page enter the "Group Name" and choose "Symmetric Key", leave the others options as default, and save it.

Microsoft Azure

Search resources, services, and docs (G+/)

[Home](#) > [Aruba-DPS](#) >

## Add Enrollment Group

Save <sup>3</sup>

Group name \*

Aruba-group <sup>1</sup>

Attestation Type ⓘ

Certificate

Symmetric Key <sup>2</sup>

Auto-generate keys ⓘ

☒

Primary Key ⓘ

Enter your primary key

Secondary Key ⓘ

Enter your secondary key

IoT Edge device ⓘ

True

False

Select how you want to assign devices to hubs ⓘ

Select the IoT hubs this group can be assigned

[Home](#) > [aruba-beijing](#) >

## IAP

...

Enrollment Group Details

Save Refresh Regenerate keys

IoT Edge device ⓘ

True

False

Select how you want to assign devices to hubs


Evenly weighted distribution

Select the IoT hubs this group can be assigned to: ⓘ

beijingtest.azure-devices.net


[Link a new IoT hub](#)




Click the group you create in the last step, copy the primary key, for later config in the Controller or IAP.

 **Microsoft Azure**


Search resources, services, and docs (G+)

Home > Aruba-DPS >


 **Aruba-Group1**  
Enrollment Group Details


 Save  Refresh  Regenerate keys

**Settings** Registration Records

 You can view and update attestation information, set how you want to assign devices to hubs, define the re-provisioning policy and set the initial twin state of provisioning devices.

**Attestation Type**  
Symmetric Key

Primary Key  
yMapHQcCRM5k6xW1yDRWOBHSMY26Ds7D9aVNvuJHiBLD2Z0LdYIdCp2tFaqU90Rer7DOnOrUsw2tjbMnCFje6w== 

Secondary Key  
\*\*\*\*\* 

IoT Edge device ⓘ  
☐ True ☒ False

## Config IAP

In the "create group device enrollment in the Azure DPS service" section, we already got "ID Scope" and "group enrollment key", we will need these two parameters to be configured in "iot transportProfile"

Following is a example. note, we introduce a new endpointType "Azure-IoTHub".

```
iot transportProfile azure-iothub-test
endpointType Azure-IoTHub
payloadContent serial-data
azure-dps-id-scope 0ne00190XXX -- Your ID Scope
azure-dps-auth-type symmetric-key group-enroll-key
yMapHQcCRM5k6xW1yDRWOBHSMY26Ds7D9aVNvuJHiBLD2Z0LdYIdCp2tFaqU90Rer7DOnOrUsw2tjbMnCFje6XXX -- Your group key
```

## Check Azure connection

### iot-profile in ble-daemon

```
70:3a:0e:cc:ed:f8# show ap debug ble-daemon iot-profile azure-iot-hub-test
```

```
BLE IoT Transport Context Config ID: 2  
Last Sync Time: 2020-10-27 03:59:57
```

```
-----Profile[azure-iot-hub-test]-----
```

```
ServerType :Azure IoT Hub  
Last Update Sent :No Update Sent  
ReportingInterval :600 second  
DeviceClassFilter :Serial Data(22),All(5)  
RSSI Reporting :Average  
EnvironmentType :office  
CustomFadingFactor :20  
DataFilter :00 00  
Server Connection State  
-----  
DPS Id Scope :0ne00190C2D  
DPS group key :*****  
IoT Hub provision type :DPS  
IoT Hub provision status :Provision success  
DPS provision retry times :0  
Last Provision Time :2020-10-27 03:59:42  
IoT Hub register id :703A0ECCEDF8  
IoT Hub URL :Aruba-IoT-Hub.azure-devices.net  
IoT Hub key :*****  
IoT Hub connection status :Connect success  
IoT Hub connect retry times :0  
Last connect Time :2020-10-27 03:59:47  
Current Time :2020-10-28 00:07:58
```

## iot-profile in ble-relay

```
70:3a:0e:cc:ed:f8# show ap debug ble-relay iot-profile
```

```
ConfigID : 3
```

```
-----Profile[azure-iot-hub-test]-----
```

```
serverType : Azure IoT Hub  
deviceClassFilter : Serial Data,All  
reportingInterval : 600 second  
authentication-mode : none  
rssiReporting : Average  
environmentType : office  
azure-dps-id-scope : 0ne00190C2D  
azure-dps-group-key : *****  
Server Connection State  
-----  
TransportContext : Ready  
Last Data Update : 2020-10-27 03:59:47  
Last Send Time : 1969-12-31 17:00:00  
TransType : Azure AMQP Over Websocket  
Transport count : 1  
-----  
-----Azure Transport[Aruba-IoT-Hub.azure-devices.net]-----  
Transport url : Aruba-IoT-Hub.azure-devices.net  
Transport created : Yes  
-----  
Device MAC : 70:3a:0e:cc:ed:f8  
Device id : 703A0ECCEDF8  
Device created : Yes  
Status : Connect success  
-----
```

## Debugging log

### ble-daemon log

```
70:3a:0e:cc:ed:f8# ble-init-action log-level-str Azure-IoTHub
```

```
log level parsed. log_level = 0x400000
```

```
70:3a:0e:cc:ed:f8# show ap debug ble-daemon
```

```
2020-10-27 03:55:15 (azure-iot-hub-test): Sending azure DPS provision request msg to ble_relay@localhost msglen:121
```

```
2020-10-27 03:55:15 az_prov_start_connect_timer:80 start azure prov connect timer.
```

```
2020-10-27 03:55:19 ble_ap_handle_az_iothub_prov:370 azure-iot-hub: Received Provision Message!
```

### ble-relay log

```
70:3a:0e:cc:ed:f8# ble-init-action ble_relay set-attr br-loglvl 2048
```

```
70:3a:0e:cc:ed:f8# show ap debug ble-relay
```

```
[9856]2020-10-27 03:59:07 ble_parse_clid_message:1055 new name:azure-iot-hub-test
```

```
[9856]2020-10-27 03:59:07 Azure connect: Azure iot-hub init...
```

```
[9856]2020-10-27 03:59:07 Azure connect: Azure iot-hub init done
```

### azure-iot-hub-log

```
70:3a:0e:cc:ed:f8# show ap debug ble-relay azure-iot-hub-log
```

```
2020-10-27 03:51:07 Azure connect: Azure iot-hub init done
```

```
2020-10-27 03:51:34 Azure connect: az_iot_connecting(114) Start azure iot-hub connecting thread.
```

```
2020-10-27 03:51:34 Azure connect: New connecting, iot-hub url Aruba-IoT-Hub.azure-devices.net, register id 703A0ECCEDF8, key *****
```

```
2020-10-27 03:51:34 az_iot_hub_creat_client_hash_entry(99): Created AP mac<70:3a:0e:cc:ed:f8>, id <703A0ECCEDF8>.
```

```
2020-10-27 03:51:34 Azure connect: az_iot_hub_create_device(265) creating client 703A0ECCEDF8
```

```
2020-10-27 03:51:34 Azure connect: az_iot_connecting(154) Create device handler done.
```

```
2020-10-27 03:51:34 Azure connect: No more Azure IoT-Hub connecting request in the queue, stop connecting.
```

```
2020-10-27 03:51:34 Azure connect: Azure iot-hub connecting thread end.
```

```
2020-10-27 03:51:35 Azure connect: The device client 703A0ECCEDF8 is connected to iot-hub
```

```
2020-10-27 03:51:35 Azure connect: Update azure connection status to daemon
```

## Troubleshooting command

### azure-clear-flash-provisioning

70:3a:0e:cc:ed:f8# ble-init-action azure-clear-flash-provisioning azure-iot-hub-test

70:3a:0e:cc:ed:f8# show ap debug ble-daemon iot-profile

BLE IoT Transport Context Config ID: 3  
Last Sync Time: 2020-10-28 00:10:20

-----Profile[azure-iot-hub-test]-----

ServerType :Azure IoT Hub  
Last Update Sent :No Update Sent  
ReportingInterval :600 second  
DeviceClassFilter :Serial Data(22),All(5)  
RSSI Reporting :Average  
EnvironmentType :office  
CustomFadingFactor :20  
DataFilter :00 00  
Server Connection State

-----  
DPS Id Scope :0ne00190C2D  
DPS group key :\*\*\*\*\*  
IoT Hub provision type :DPS  
IoT Hub provision status :Provisioning  
DPS provision retry times :0  
Last Provision Time :2020-10-28 00:21:44  
IoT Hub register id :  
IoT Hub URL :  
IoT Hub key :Null  
IoT Hub connection status :Connect fail  
IoT Hub connect retry times :0  
IoT Hub failure reason :IOTHUB\_CLIENT\_CONNECTION\_OK  
Last failure Time :2020-10-28 00:21:44  
Last connect Time :2020-10-27 03:59:47  
Current Time :2020-10-28 00:21:47  
70:3a:0e:cc:ed:f8#  
70:3a:0e:cc:ed:f8#  
70:3a:0e:cc:ed:f8# show ap debug ble-daemon iot-profile

BLE IoT Transport Context Config ID: 3  
Last Sync Time: 2020-10-28 00:10:20

-----Profile[azure-iot-hub-test]-----

ServerType :Azure IoT Hub  
Last Update Sent :No Update Sent  
ReportingInterval :600 second  
DeviceClassFilter :Serial Data(22),All(5)  
RSSI Reporting :Average  
EnvironmentType :office  
CustomFadingFactor :20  
DataFilter :00 00  
Server Connection State

-----  
DPS Id Scope :0ne00190C2D  
DPS group key :\*\*\*\*\*  
IoT Hub provision type :DPS  
IoT Hub provision status :Provision success  
DPS provision retry times :0  
Last Provision Time :2020-10-28 00:21:44  
IoT Hub register id :703A0ECCEDF8  
IoT Hub URL :[Aruba-IoT-Hub.azure-devices.net](https://Aruba-IoT-Hub.azure-devices.net)  
IoT Hub key :\*\*\*\*\*  
IoT Hub connection status :Connect success  
IoT Hub connect retry times :0  
Last connect Time :2020-10-28 00:21:50  
Current Time :2020-10-28 00:21:53

## azure-dps-provisioning

70:3a:0e:cc:ed:f8# ble-init-action azure-dps-provisioning azure-iot-hub-test

70:3a:0e:cc:ed:f8# show ap debug ble-daemon iot-profile

BLE IoT Transport Context Config ID: 3  
Last Sync Time: 2020-10-28 00:10:20

-----Profile[azure-iot-hub-test]-----

ServerType :Azure IoT Hub  
Last Update Sent :No Update Sent  
ReportingInterval :600 second  
DeviceClassFilter :Serial Data(22),All(5)  
RSSI Reporting :Average  
EnvironmentType :office  
CustomFadingFactor :20  
DataFilter :00 00  
Server Connection State

-----  
DPS Id Scope :0ne00190C2D  
DPS group key :\*\*\*\*\*  
IoT Hub provision type :DPS  
IoT Hub provision status :Provisioning  
DPS provision retry times :0  
Last Provision Time :2020-10-28 00:24:15  
IoT Hub register id :703A0ECCEDF8  
IoT Hub URL :[Aruba-IoT-Hub.azure-devices.net](https://Aruba-IoT-Hub.azure-devices.net)  
IoT Hub key :\*\*\*\*\*  
IoT Hub connection status :Connect success  
IoT Hub connect retry times :0  
Last connect Time :2020-10-28 00:21:50  
Current Time :2020-10-28 00:24:17  
70:3a:0e:cc:ed:f8#  
70:3a:0e:cc:ed:f8# show ap debug ble-daemon iot-profile

BLE IoT Transport Context Config ID: 3  
Last Sync Time: 2020-10-28 00:10:20

-----Profile[azure-iot-hub-test]-----

ServerType :Azure IoT Hub  
Last Update Sent :No Update Sent  
ReportingInterval :600 second  
DeviceClassFilter :Serial Data(22),All(5)  
RSSI Reporting :Average  
EnvironmentType :office  
CustomFadingFactor :20  
DataFilter :00 00  
Server Connection State

-----  
DPS Id Scope :0ne00190C2D  
DPS group key :\*\*\*\*\*  
IoT Hub provision type :DPS  
IoT Hub provision status :Provision success  
DPS provision retry times :0  
Last Provision Time :2020-10-28 00:24:15  
IoT Hub register id :703A0ECCEDF8  
IoT Hub URL :[Aruba-IoT-Hub.azure-devices.net](https://Aruba-IoT-Hub.azure-devices.net)  
IoT Hub key :\*\*\*\*\*  
IoT Hub connection status :Connect success  
IoT Hub connect retry times :0  
Last connect Time :2020-10-28 00:21:50  
Current Time :2020-10-28 00:24:24

**azure-iot-hub-connect without connect parameters**



```

70:3a:0e:cc:ed:f8# ble-init-action azure-iot-hub-connect azure-iot-hub-test

70:3a:0e:cc:ed:f8# show ap debug ble-daemon iot-profile

BLE IoT Transport Context Config ID: 3
Last Sync Time: 2020-10-28 00:10:20

-----Profile[azure-iot-hub-test]-----

ServerType :Azure IoT Hub
Last Update Sent :No Update Sent
ReportingInterval :600 second
DeviceClassFilter :Serial Data(22),All(5)
RSSI Reporting :Average
EnvironmentType :office
CustomFadingFactor :20
DataFilter :00 00
Server Connection State
-----
DPS Id Scope :0ne00190C2D
DPS group key :*****
IoT Hub provision type :DPS
IoT Hub provision status :Provision success
DPS provision retry times :0
Last Provision Time :2020-10-28 00:24:15
IoT Hub register id :703A0ECCEDF8
IoT Hub URL :Aruba-IoT-Hub.azure-devices.net
IoT Hub key :*****
IoT Hub connection status :Connect success
IoT Hub connect retry times :0
Last connect Time :2020-10-28 00:26:29
Current Time :2020-10-28 00:26:30

```

## azure-iot-hub-connect with connection parameters

```

70:3a:0e:cc:ed:f8# ble-init-action azure-iot-hub-connect azure-iot-hub-test Aruba-IoT-Hub.azure-devices.net 703A0ECCEDF8
fouDREhrzSjSAYM+ud906lKZchwJC9946tduHwsxkZs=

70:3a:0e:cc:ed:f8# show ap debug ble-daemon iot-profile azure-iot-hub-test

BLE IoT Transport Context Config ID: 3
Last Sync Time: 2020-10-28 00:10:20

-----Profile[azure-iot-hub-test]-----

ServerType :Azure IoT Hub
Last Update Sent :No Update Sent
ReportingInterval :600 second
DeviceClassFilter :Serial Data(22),All(5)
RSSI Reporting :Average
EnvironmentType :office
CustomFadingFactor :20
DataFilter :00 00
Server Connection State
-----
DPS Id Scope :0ne00190C2D
DPS group key :*****
IoT Hub provision type :DPS
IoT Hub provision status :Provision success
DPS provision retry times :0
Last Provision Time :2020-10-28 00:24:15
IoT Hub register id :703A0ECCEDF8
IoT Hub URL :Aruba-IoT-Hub.azure-devices.net
IoT Hub key :*****
IoT Hub connection status :Connect success
IoT Hub connect retry times :0
Last connect Time :2020-10-28 00:28:12
Current Time :2020-10-28 00:28:14

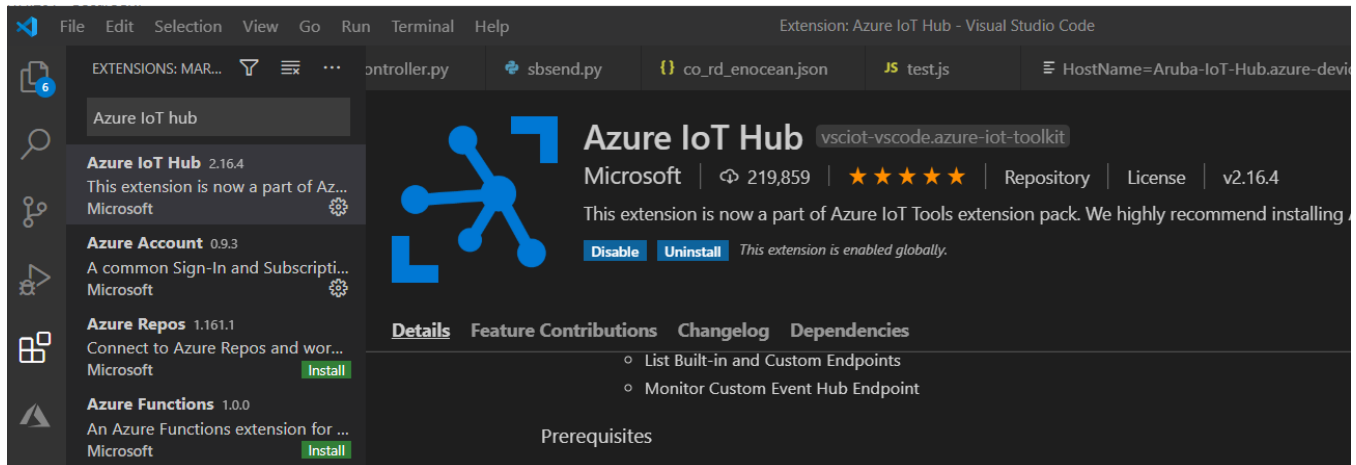
```

## Back-end Application

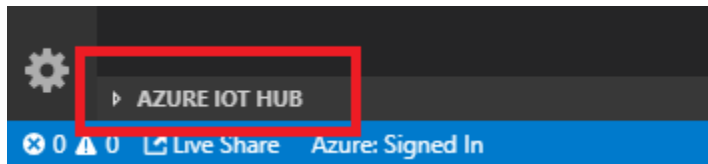
For the back-end application, we can use VSC Azure IoT Hub extension or python script.

## Back-end Application-VSC Extension

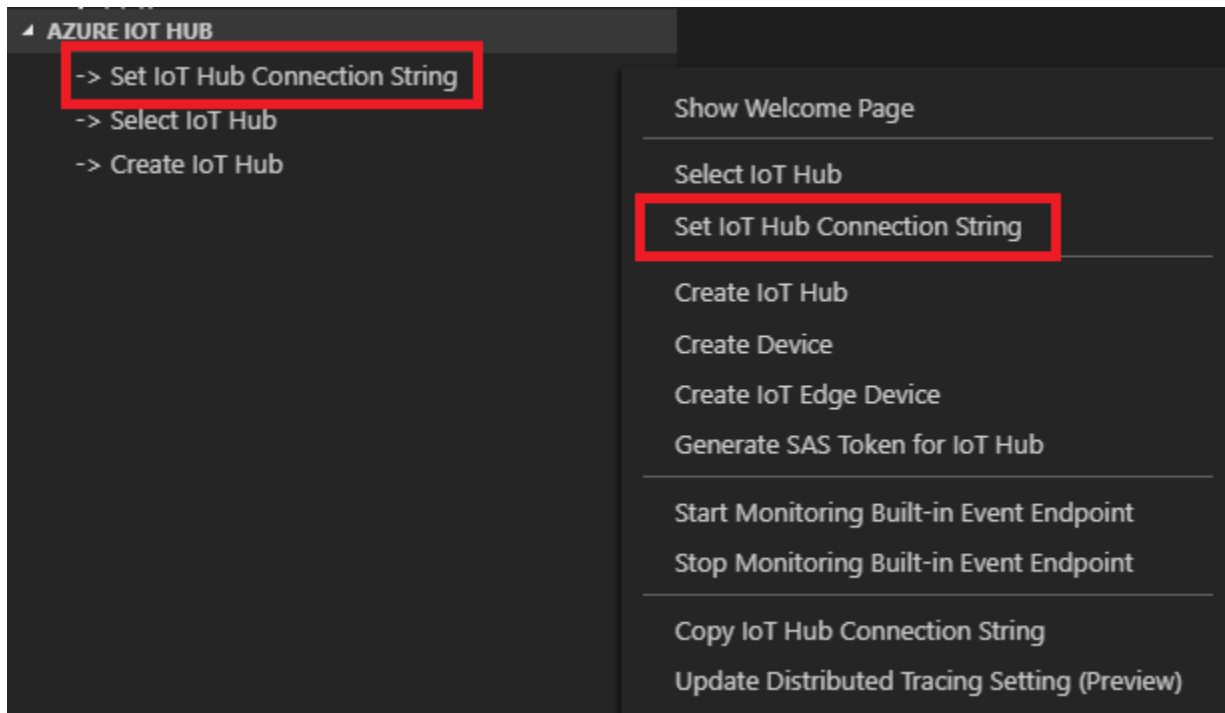
In the extensions of the VSC, search "Azure IoT Hub", and install it.



In Explorer of VS Code, click "Azure IoT Hub" in the bottom left corner.

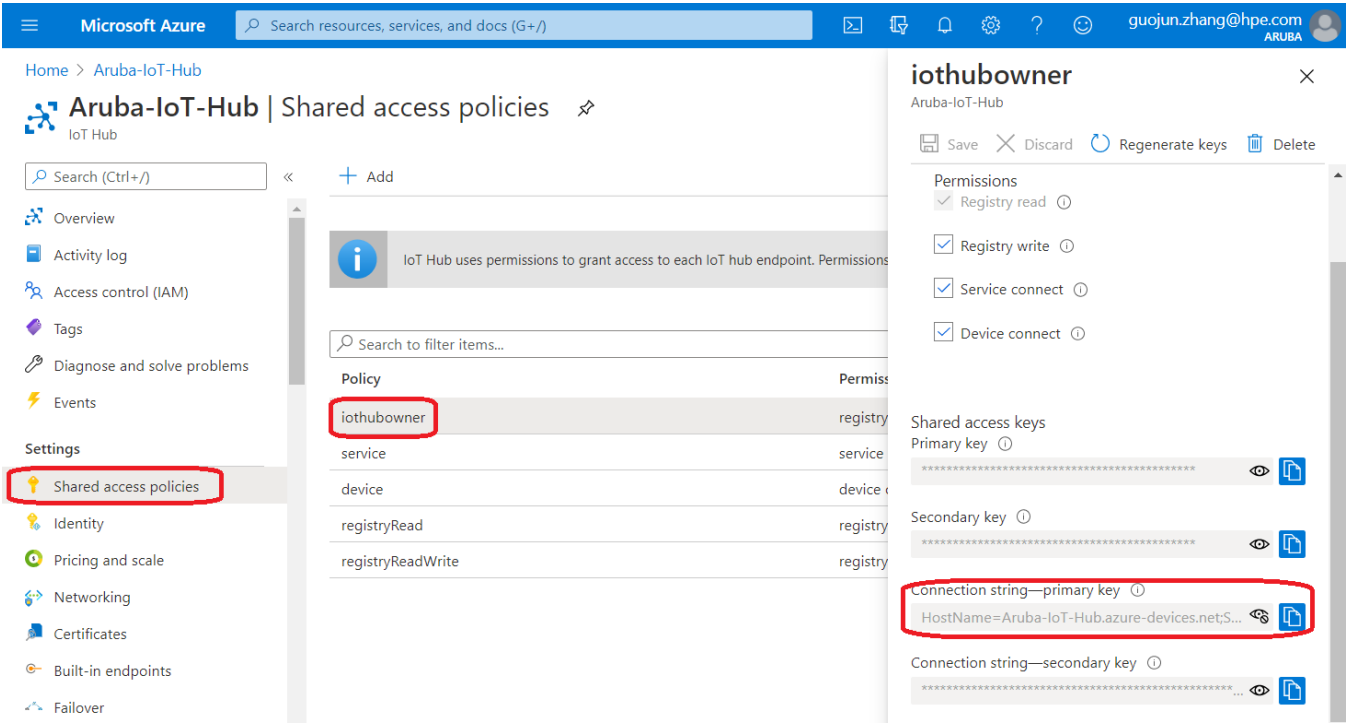


Click "Set IoT Hub Connection String" in context menu.

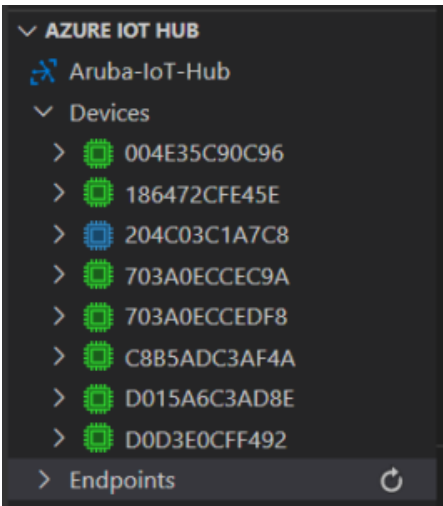


An input box will pop up, then enter your IoT Hub Connection String (It is one-time configuration, and please make sure it is IoT Hub Connection String not Device Connection String. The format is `HostName=<my-hub>.azure-devices.net;SharedAccessKeyName=<my-policy>;SharedAccessKey=<my-policy-key>`).

you can get iotHub conntction string from your iotHub



The devices list will be shown.

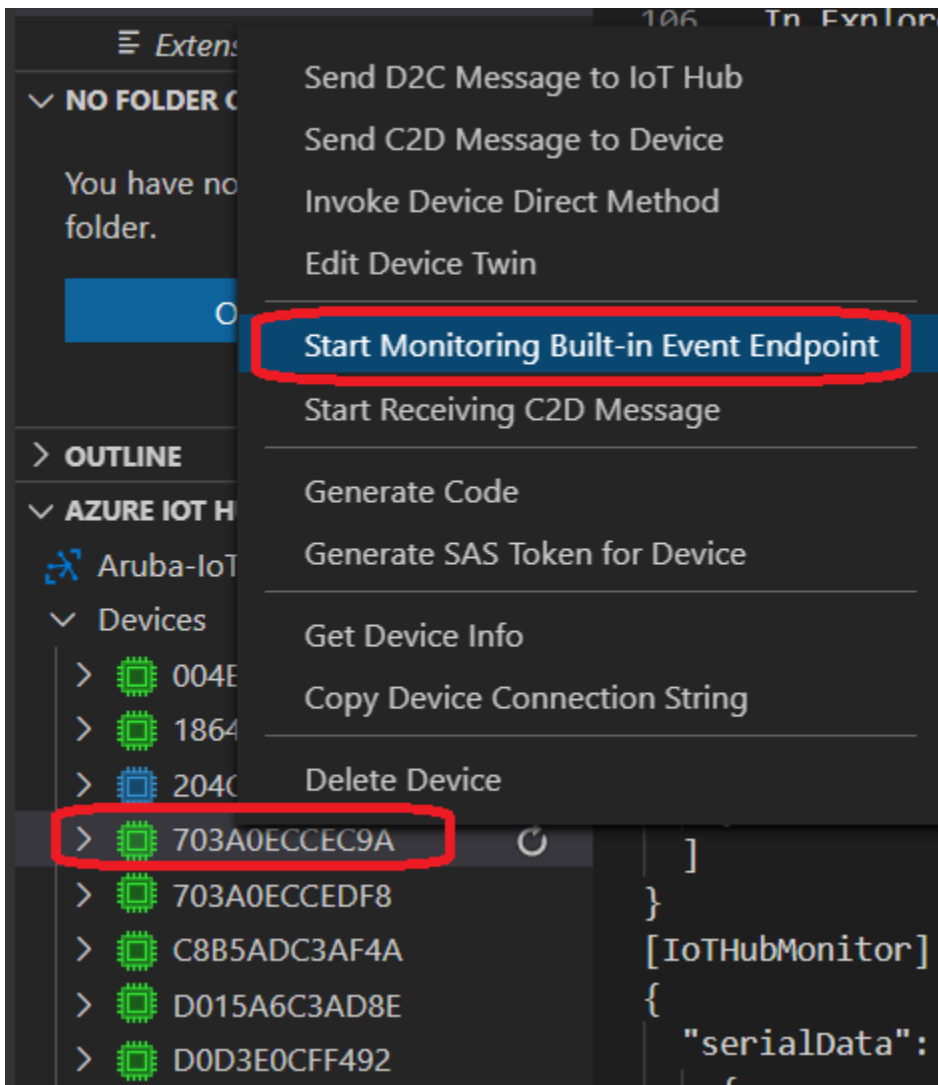


monitoring global message for your iotHub

The screenshot displays the Visual Studio Code interface with the Azure IoT Hub extension installed. The left sidebar shows the Explorer view with a file list including `upload_ble.py`, `upload_ble_controller.py`, `sbsend.py`, `co_rd_enocean.json`, `test.js`, and a folder named `HostName=Aruba-IoT-Hu`. Below the file list, a message states "NO FOLDER OPENED" with a button labeled "Open Folder". The Outline view on the left shows the "AZURE IOT HUB" section expanded, displaying the "Aruba-IoT-Hub" folder and a list of devices: `004E35C90C96`, `186472CFE45E`, and `204C03C1A7C8`. A red circle highlights the three-dot menu icon next to the "AZURE IOT HUB" header. The right sidebar shows the "Azure IoT Hub" extension menu, which includes options like "Show Welcome Page", "Send D2C Message to IoT Hub", "Select IoT Hub", "Set IoT Hub Connection String", "Create IoT Hub", "Create Device", "Create IoT Edge Device", "Generate SAS Token for IoT Hub", "Start Monitoring Built-in Event Endpoint" (highlighted with a red rectangle), "Stop Monitoring Built-in Event Endpoint", "Copy IoT Hub Connection String", and "Update Distributed Tracing Setting (Preview)". The bottom right corner shows a log output with the following text:

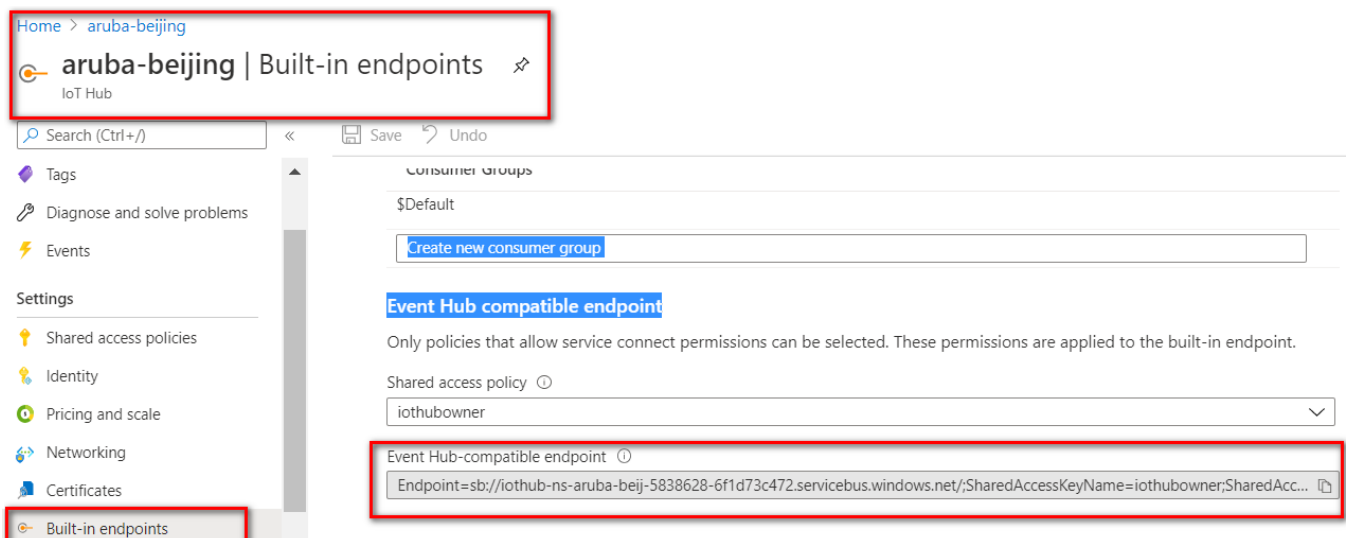
```
[IoTHubMonitor] Created partition
[IoTHubMonitor] [1:01:11 PM] Message
{
  "serialData": [
    {
      "data": "VQAHBwF69gAAKZyJoAH"
```

You can also monitor the specific device's message



Event Hub compatible endpoint

home----iot Hub-----Built-in endpoints



## Output example

```
[IoTHubMonitor] Start monitoring message arrived in built-in endpoint for device [703A0ECCEC9A] ...
[IoTHubMonitor] Created partition receiver [0] for consumerGroup [$Default]
[IoTHubMonitor] Created partition receiver [1] for consumerGroup [$Default]
[IoTHubMonitor] Created partition receiver [2] for consumerGroup [$Default]
[IoTHubMonitor] Created partition receiver [3] for consumerGroup [$Default]
[IoTHubMonitor] [1:01:11 PM] Message received from [703A0ECCEC9A]:
{
  "serialData": [
    {
      "data": "VQAHBwF69gAAKZyJoAH/////NABV"
    }
  ]
}
[IoTHubMonitor] [1:01:11 PM] Message received from [703A0ECCEC9A]:
{
  "serialData": [
    {
      "data": "VQAHBwF69IAAKZyJsAH/////LQAM"
    }
  ]
}
[IoTHubMonitor] Stopping built-in event endpoint monitoring...
[IoTHubMonitor] Built-in event endpoint monitoring stopped.
```

## Back-end Application-Python

We will use python samples to receive device-to-cloud messages and send cloud-to-device message.

download following samples from github to start with.

<https://github.com/Azure-Samples/azure-iot-samples-python>

you need to install your python in your system, require Python 3.6.x (or higher).

you also required to install "azure-iot-hub" and "azure-eventhub" module.

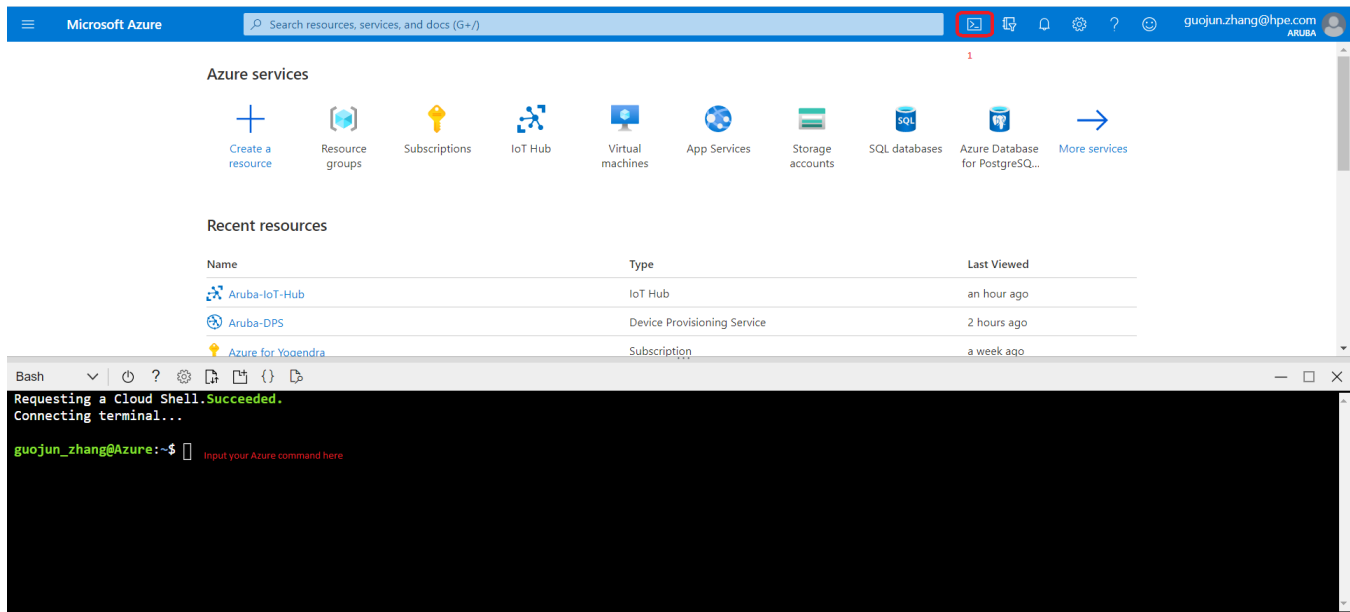
```
python3 -m pip install azure-iot-hub
```

```
python3 -m pip install azure-eventhub
```

## Receive device to cloud messages

You also need the Event Hubs-compatible endpoint, Event Hubs-compatible path, and service primary key from your IoT hub to enable the back-end application to connect to your IoT hub and retrieve the messages.

You can open the Azure CLI from your azure account home page as displayed by following screenshot.



The following commands retrieve these values for your IoT hub:

Replace the "{YourIoTHubName}" below with the name you choose for your IoT hub.

```
az iot hub show --query properties.eventHubEndpoints.events.endpoint --name {YourIoTHubName}
az iot hub show --query properties.eventHubEndpoints.events.path --name {YourIoTHubName}
az iot hub policy show --name service --query primaryKey --hub-name {YourIoTHubName}
```

Navigate to the root folder of the sample Python project. Then navigate to the `iot-hub\Quickstarts\read-d2c-messages` folder.

Open the `read_device_to_cloud_messages_sync.py` file, and replace following three variables "EVENTHUB\_COMPATIBLE\_ENDPOINT", "EVENTHUB\_COMPATIBLE\_PATH", "IOTHUB\_SAS\_KEY".

Run the following command to receive the device to cloud message.

```
python3 read_device_to_cloud_messages_sync.py
```

The output example

```
Received event from partition: 0.
Telemetry received: {"serialData": [{"data": "VQAHBwF69gAAKZyJoAH/////NgB/"}]}
Properties (set by device): {b'version': b'1', b'messageType': b'serialDataNb'}
System properties (set by IoT Hub): {b'message-id': b'MSG_ID', b'correlation-id': b'CORE_ID', b'content-type': b'application%2fjson', b'content-encoding': b'utf-8', b'iothub-connection-device-id': b'703A0ECCEC9A', b'iothub-connection-auth-method': b'{"scope": "device", "type": "sas", "issuer": "iothub", "acceptingIpFilterRule": null}', b'iothub-connection-auth-generation-id': b'637388596578496697', b'iothub-enqueuedtime': 1603904565349, b'iothub-message-source': b'Telemetry', b'x-opt-sequence-number': 103, b'x-opt-offset': b'53576', b'x-opt-enqueued-time': 1603904565341}
```

## Send cloud to device messages

To use cloud to device message, you need need a service connection string to enable the back-end application to connect to your IoT hub. The following command retrieves the service connection string for your IoT hub:

Replace the "{YourIoTHubName}" below with the name you choose for your IoT hub.

```
az iot hub show-connection-string --policy-name service --name {YourIoTHubName} --output table
```

The example give blow is based on Enocean usb device.

```

import random
import sys
from azure.iot.hub import IoTHubRegistryManager

MESSAGE_COUNT = 1

JSON_TXT = "{\"sbDeviceId\":\"ENOCAN_USB\", \"serialData\":{\"data\":\"VQABAAVwAwk=\"}}\"

CONNECTION_STRING = "{CONNECTION_STRING}"

DEVICE_ID = "{DEVICE_ID}"

def iot_hub_messaging_sample_run():
    try:
        # Create IoTHubRegistryManager
        registry_manager = IoTHubRegistryManager(CONNECTION_STRING)

        for i in range(0, MESSAGE_COUNT):
            print ( 'Sending message: {0}'.format(i) )
            data = JSON_TXT
            print ('data: %s' % data)

            props={}
            # optional: assign system properties
            props.update(messageId = "message_%d" % i)
            props.update(correlationId = "correlation_%d" % i)
            props.update(contentType = "application/json")

            # optional: assign application properties
            #prop_text = "PropMsg_%d" % i
            #props.update(testProperty = prop_text)
            props.update(messageType = "serialDataSb")
            props.update(deviceIdentifier = "11111111")
            print ('props: %s' % props)

            registry_manager.send_c2d_message(DEVICE_ID, data, properties=props)

    except:
        # Try Python 2.xx first
        raw_input("Press Enter to continue...\n")
    except:
        pass
        # Use Python 3.xx in the case of exception
        input("Press Enter to continue...\n")

except Exception as ex:
    print ( "Unexpected error {0}" % ex )
    return
except KeyboardInterrupt:
    print ( "IoT Hub C2D Messaging service sample stopped" )

if __name__ == '__main__':
    print ( "Starting the Python IoT Hub C2D Messaging service sample..." )

    iot_hub_messaging_sample_run()

```

Run SendCloudToDeviceMessage example:

```

swatch@swatch-XPS-L412Z:Send-Cloud-To-Device-Message$ python3 SendCloudToDeviceMessage.py
Starting the Python IoT Hub C2D Messaging service sample...
Sending message: 0
data: {'sbDeviceId': 'ENOCAN_USB', 'serialData': {'data': 'VQABAAVwAwk='}}
props: {'messageId': 'message_0', 'correlationId': 'correlation_0', 'contentType': 'application/json', 'messageType': 'serialDataSb', 'deviceIdentifier': '11111111'}
Press Enter to continue...

```

## Trace messages on the IAP

We need to enable ble\_daemon and ble\_relay logs for "Azure-IoTHub" specifically



```
70:3a:0e:cc:ec:9a# ble-init-action log-level-str Azure-LoTHub  
log level parsed. log_level = 0x400000  
70:3a:0e:cc:ec:9a# ble-init-action ble_relay set-attr br-loglvl 2048
```

## North bound example:

Run "show ap debug ble-daemon" to check if ble-daemon forward data to Azure IoT Hub

```
2020-10-28 21:03:29 ble_ap_fw_serial_data:5374 forward serial reporting to Azure for azure-iot-hub-test profile  
2020-10-28 21:03:29 Send message to Azure IoT Hub messageType serialDataNb, ID Null, data: {"serialData": [{"data": "VQAHBwF69gAAKZyJoAH  
/////KgDU"}]}.
```

Run "show ap debug ble-relay" to check if ble\_relay received the data to Azure IoT Hub

```
[6346]2020-10-28 21:03:29 ble_relay_handle_azure_iot-hub_data_msg:135 Aruba Telemetry :1  
[6346]2020-10-28 21:03:29 br_azure_iot-hub_msg:67 msg from 127.0.0.1:8514, len: 142.
```

Run "show ap debug ble-relay azure-iot-hub-log" to check if we send the data to Azure IoT Hub

```
2020-10-28 21:03:29 Sending serialDataNb message (id ) to IoT Hub
```

## South bound example:

Run "show ap debug ble-relay azure-iot-hub-log" to check if the message is received by azure iot-hub library.

```
2020-10-28 19:07:07 Received Binary message  
Message ID: message_0  
Correlation ID: correlation_0  
Data: <<<{"sbDeviceId":"ENOCLEAN_USB","serialData":[{"data":"VQABAAVwAwk="}]}>>> & Size=67  
2020-10-28 19:07:07 Message Properties:  
2020-10-28 19:07:07 Key: messageType Value: serialDataSb  
2020-10-28 19:07:07 Key: deviceId Value: 11111111  
2020-10-28 19:07:07  
2020-10-28 19:07:07 Azure SB MSG: send sb message to ble_relay messageType serialDataSb, deviceId 11111111..
```

Run "show ap debug ble-relay" to check if the ble\_relay forward the SB message to ble\_daemon.

```
[6346]2020-10-28 19:07:07 azure_iot-hub_forward_sbsdata_to_ble_daemon:665 Azure-LoTHub-SB-Data: receive sb data, messageType serialDataSb,  
deviceId 11111111.  
[6346]2020-10-28 19:07:07 azure_iot-hub_forward_sbsdata_to_ble_daemon:742 Azure-LoTHub-SB-Data: send sb data to 127.0.0.1, messageType  
serialDataSb, deviceId 11111111.
```

Run "show ap debug ble-daemon" to check if the ble\_daemon handled the SB message.

```
2020-10-28 19:07:07 Received SB message  
Message ID: serialDataSb  
device ID: 11111111  
Data: <<<{"sbDeviceId":"ENOCLEAN_USB","serialData":[{"data":"VQABAAVwAwk="}]}>>> & Size=67  
  
2020-10-28 19:07:07 ble_ap_handle_az_iot-hub_sb_data:629 start to write message to device ENOCLEAN_USB, data: VQABAAVwAwk=(base64 format).  
2020-10-28 19:07:07 ble_ap_send_data_to_serial_device:5444 Send serial data to ENOCLEAN_USB.
```