

iGS01S/iGS02E/iGS03 Azure IoT Hub Guide

Introduction

This application note provides a step-by-step setup to connect iGS01S/iGS02E/iGS03 with Azure IoT Hub. Azure-IoT allows Symmetric Key or X.509 Certificates for internal authorization (<https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-devguide-security>). Both should work with IGS01S/iGS02E/iGS03.

Prerequisites

Before using iGS01s/iGS02E with Azure IoT, you have to create an IoT hub with your Azure subscription. See [Create an IoT hub](#) for detailed steps.

Example By Using Symmetric Key

IoT Hub can use security tokens to authenticate devices and services to avoid sending keys on the wire. To use a security token, create a device with authentication type "Symmetric key".

Device ID * ⓘ

Authentication type ⓘ

Symmetric key X.509 Self-Signed X.509 CA Signed










Primary key ⓘ

Secondary key ⓘ

Auto-generate keys ⓘ



After the device is created, you can get the symmetric key in device detail.

Device ID ⓘ	igs01s	
Primary Key ⓘ	MOS9pHMRZ0P+0So78uhjhEQeVx8nHlBucSv4lmB74gQ=	 
Secondary Key ⓘ	 
Primary Connection String ⓘ	 
Secondary Connection String ⓘ	 

Then we need to generate the SAS token from the symmetric key for the MQTT connection. Please refer to [Security Token](#) for details about the token structure. And the link also provides some sample code for how to generate the SAS token for device use.

Or, you can use a tool provided by Intel

<https://github.com/intel-iot-devkit/iot-samples-cloud-setup/releases>

Usage: sastoken <IoT Hub Name>.azure-devices.net/devices/<Device ID> <Device Primary Key> 1440

Here is the example:

```
$ ./sastoken igs-test.azure-devices.net/devices/igs01s
MOS9pHMRZ0P+0So78uhjhEQeVx8nHlBucSv4lmB74gQ= 1440

SharedAccessSignature
sr=igs-test.azure-devices.net%2Fdevices%2Figs01s&sig=12bF1jJ%2FQWg8XYJPzfne1vWN5HEp02VvIBDEcc1
wx7I%3D&se=1577324521
```

We also suggest users to test your configurations on PC first to confirm your settings are correct.

You can download the mosquitto tool to test connecting Azure IoT Hub with mqtt.

<https://mosquitto.org/download/>

Below shows an example to publish a "hello" message to Azure IoT using mosquitto command line tool.

```
$ mosquitto_pub -h igs-test.azure-devices.net -p 8883 -i igs01s -t
devices/igs01s/messages/events/ -u igs-test.azure-devices.net/igs01s -P
"SharedAccessSignature
sr=igs-test.azure-devices.net%2Fdevices%2Figs01s&sig=12bF1jJ%2FQWg8XYJPzfne1vWN5HEp02VvIBDEcc1
wx7I%3D&se=1577324521"
--capath /etc/ssl/certs/ --tls-version tlsv1.2 -d -V mqttv311 -q 0 -m "hello"
```

Configuration for iGS01S/iGS02E

Here is the iGS01S/iGS02E applications configurations:

- MQTT HOST: <IoT Hub Name>.[azure-devices.net](#)
- MQTT PORT: 8883
- MQTT PUBTOPIC: devices/<Device ID>/messages/events/
- MQTT CLIENTID: <Device ID>
- MQTT USERNAME: <IoT Hub Name>.[azure-devices.net](#)/<Device ID>
- MQTT PASSWORD: <SAS Token of the Device>
- Enable MQTTS
- Select Azure-IoT-Hub RootCA
- Disable use certificate

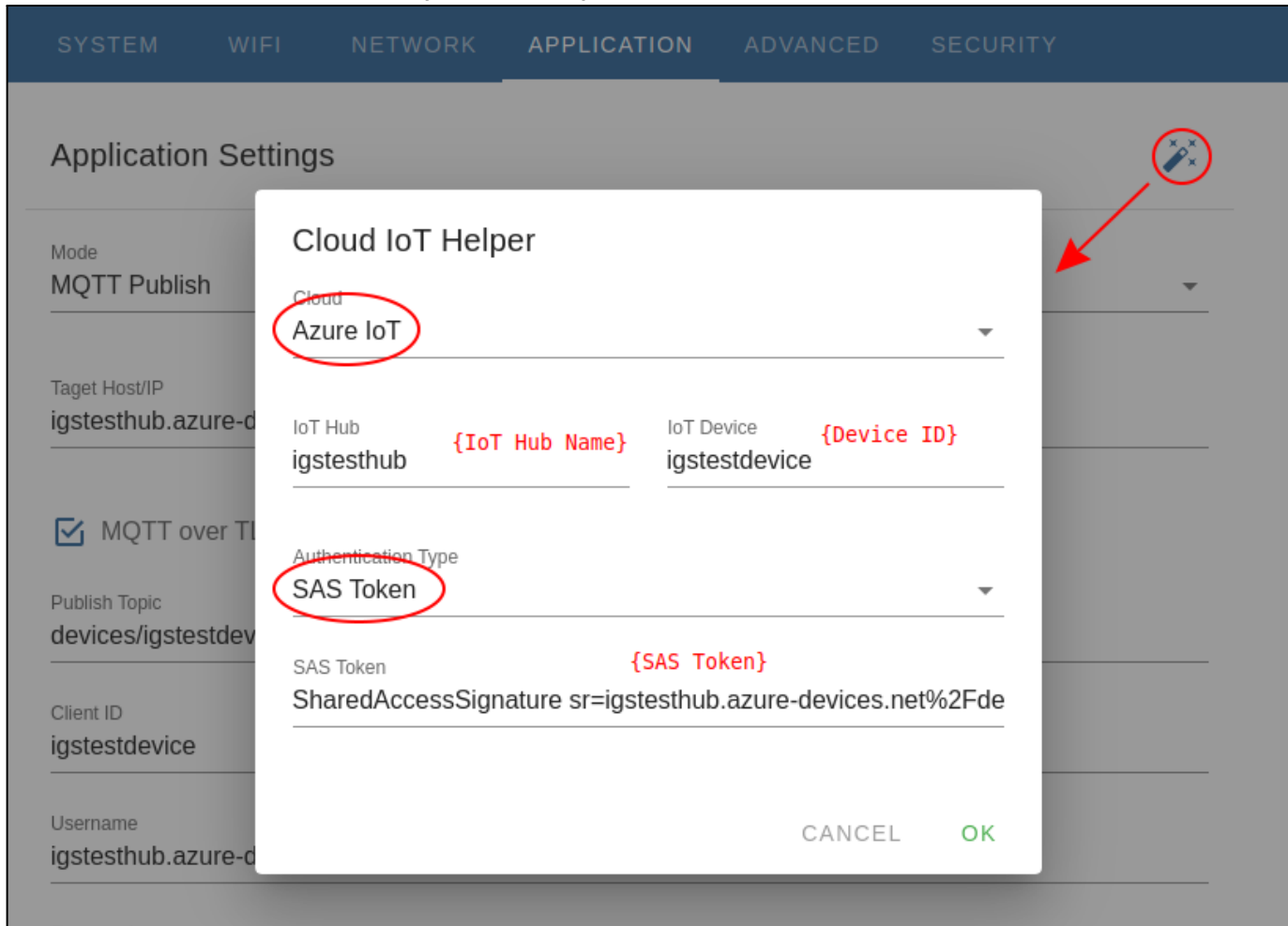
The screenshot shows the configuration interface for the iGS01S/iGS02E device. The top navigation bar includes tabs for BLE-WIFI, Wi-Fi, Network, Applications (selected), Advanced, System, and Reboot. The Applications tab is active, displaying the following configuration fields:

Application	MQTT Client ▾
Host/IP	igs01-iot.azure-devices.n
Port	8883
Publish Topic	devices/igs01-iot/messaç
Client ID	igs01-iot
Username	igs01-iot.azure-devices.n
Password	SharedAccessSignature
MQTTS	Enable ▾
Root CA	Azure-IoT Hub ▾
Use Certificate	Disable ▾
Request Interval (in secs)	1
Throttle Control (filter out redundant records)	<input type="checkbox"/>

At the bottom of the configuration form, there are two buttons: Save and Cancel.

Configuration for iGS03 Series


Users can use the 'Cloud IoT Helper' to set up the iGS03.



The application settings will be filled by 'Cloud IoT Helper', or users can fill the configuration manually.

SYSTEM WIFI NETWORK APPLICATION ADVANCED SECURITY

Application Settings

Mode
MQTT Publish 

Target Host/IP **{IoT Hub Name}.azure-devices.net**
igstesthub.azure-devices.net


Port **8883**

MQTT over TLS (MQTTS)


Publish Topic **devices/{Device ID}/messages/events/**
devices/igstestdevice/messages/events/

Client ID **{Device ID}**
igstestdevice

Username **{IoT Hub Name}.azure-devices.net/{Device ID}**
igstesthub.azure-devices.net/igstestdevice

Password **{SAS Token}**
SharedAccessSignature sr=igstesthub.azure-devices.net%2Fdevices%2Figstestdevice&sig=3ssf1Rl 

Use Client Certificate

Server Root CA **Azure-IoT Hub** 

Example By Using X.509 Self-Signed Certificates

You can create your own self-signed certificates for device authentication. Azure requires two (primary & secondary) keys for a single IoT self-signed certificate device. Here is the example for creating primary certificate:

```
$ openssl genrsa -out primary.key 2048
$ openssl req -new -key primary.key -sha256 -out primary.csr
$ openssl x509 -req -days 365 -in primary.csr -signkey primary.key -sha256 -out primary.cert
$ openssl x509 -in primary.cert -out primary.cert.pem -outform PEM
$ openssl rsa -in primary.key -out primary.key.pem -outform PEM
```

INGICS TECHNOLOGY

Repeat the same commands to create the secondary certificates.
And then get the fingerprint of both certificates for creating IoT devices.

```
$ openssl x509 -text -fingerprint -in primary.cert.pem | grep Fingerprint | cut -d '=' -f 2 |  
sed 's/[[::]]//g'  
5400819C589D43EABDFA32CEE5F26124E1353A16  
$ openssl x509 -text -fingerprint -in secondary.cert.pem | grep Fingerprint | cut -d '=' -f 2  
| sed 's/[[::]]//g'  
2A276C509956C50BCAE46D2B6D112D38BC95056E
```

Create an IoT device with the fingerprints.

Device ID * ⓘ

igs02e ✓

Authentication type ⓘ

Symmetric key X.509 Self-Signed X.509 CA Signed

Primary Thumbprint * ⓘ

5400819C589D43EABDFA32CEE5F26124E1353A16 ✓

Secondary Thumbprint * ⓘ

2A276C509956C50BCAE46D2B6D112D38BC95056E ✓

Connect this device to an IoT hub ⓘ

Enable Disable

Use mosquito tool on PC to verify the Azure configuration. Whatever using the primary key or the secondary key should pass the authentication.

```
$ mosquito_pub -h <hub_name>.azure-devices.net -p 8883 -t  
devices/<device_id>/messages/events/ -i <device_id> --cert primary.cert.pem --key  
primary.key.pem -u <hub_name>.azure-devices.net/<device_id> -d -V mqttv311 -q 0 --capath  
/etc/ssl/certs/ -m 'hello'
```

Configuration for iGS01S/iGS02E

The application configurations:

- MQTT HOST: <IoT Hub Name>.azure-devices.net
- MQTT PORT: 8883

- MQTT PUBTOPIC: devices/<Device ID>/messages/events/
- MQTT CLIENTID: <Device ID>
- MQTT USERNAME: <IoT Hub Name>.azure-devices.net/<Device ID>
- Enable MQTTS
- Select Azure-IoT-Hub RootCA
- Enable use certificate

BLE-WIFI	Wi-Fi	Network	Applications	Advanced	System	Reboot
Application						
Application	MQTT Client ▾					
Host/IP	igs-test.azure-devices.net					
Port	8883					
Publish Topic	devices/igs01s/messages/ev					
Client ID	igs01s					
Username	igs-test.azure-devices.net/ig:					
Password	password					
MQTTS	Enable ▾					
Root CA	Azure-IoT Hub ▾					
Use Certificate	Enable ▾					
Format Type	plain-text ▾					
Request Interval (in secs)	1					

INGICS TECHNOLOGY

Upload the primary.key.pem as key and primary.cert.pem as certificate via webUI advanced page.

Device Key/Certificate Update

Existing Brief

```
-----BEGIN CERTIFICATE-----  
MIIFuTCCA6GgAwIBAgIBAzANBgkqhkiG9w0BAQsFADAqMS  
gwJgYDVQQDDDB9BenVy  
ZSBJb1QgSHViIENBIENlcnQgVGVzdCBPbm ...
```

選擇檔案 未選擇任何檔案

Certificate

Upload Certificate

Clear Certificate

Existing Brief

```
-----BEGIN RSA PRIVATE KEY-----  
MIJKAIBAAKCAgEAlIn44PUonPT4dTWXhVcO5J+oHJaXJzx  
0me0gwyw7PHmlj7/m  
yjFNsDhx0HulWqCiRzL/9Q5HP12ky9 ...
```

選擇檔案 未選擇任何檔案

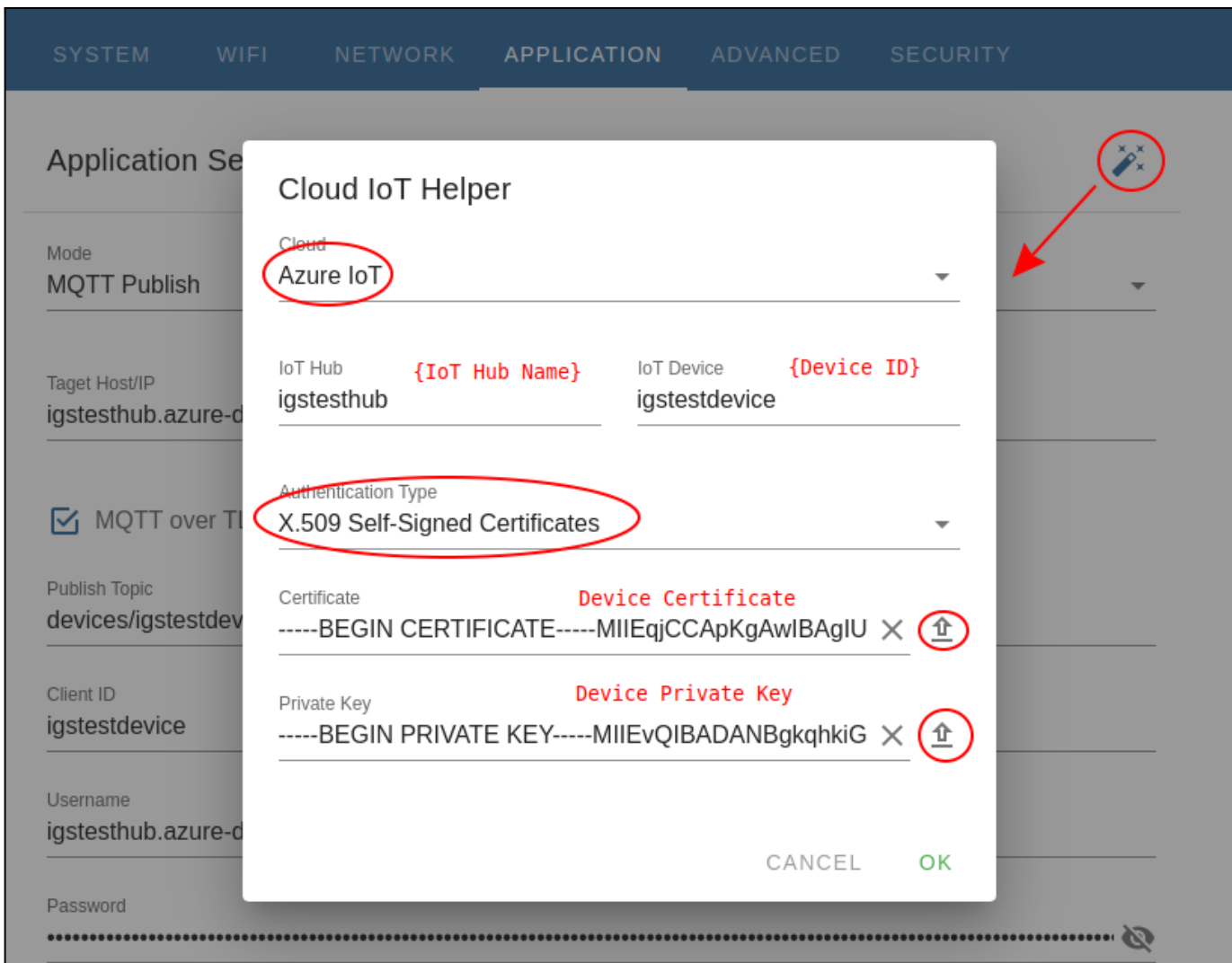
Key

Upload Key

Clear Key

Configuration for iGS03 Series


Users can use the 'Cloud lot Helper' to set up the iGS03.



The application settings will be filled by 'Cloud IoT Helper', or users can fill the configuration manually.

SYSTEM WIFI NETWORK APPLICATION ADVANCED SECURITY

Application Settings

Mode
MQTT Publish 

Target Host/IP {Iot Hub Name}.azure-devices.net
igstesthub.azure-devices.net


Port 8883

MQTT over TLS (MQTTS)


Publish Topic devices/{Device ID}/messages/events/
devices/igstestdevice/messages/events/

Client ID {Device ID}
igstestdevice

Username {Iot Hub Name}.azure-devices.net/{Device ID}
igstesthub.azure-devices.net/igstestdevice

Password
--- 

Use Client Certificate

Server Root CA Azure-IoT Hub 

Don't forget to upload the device certificate and private key in the SECURITY tab (Using the 'Cloud IoT Helper', the files should already be uploaded).


```
tim@tim-X411UA: ~  
tim@tim-X411UA: ~ 116x21  
tim@tim-X411UA:~$ az iot hub monitor-events --output table -p all -n igstesthub  
Starting event monitor, use ctrl-c to stop...  
event:  
  annotations:  
    iothub-connection-auth-generation-id: '637219317146240614'  
    iothub-connection-auth-method: '{"scope":"device","type":"sas","issuer":"iothub","acceptingIpFilterRule":null}'  
    iothub-connection-device-id: igstestdevice  
    iothub-enqueuedtime: 1657867836023  
    iothub-message-source: Telemetry  
    x-opt-enqueued-time: 1657867836029  
    x-opt-offset: '498216902272'  
    x-opt-sequence-number: 39396  
  component: ''  
  interface: ''  
  module: ''  
  origin: igstestdevice  
  payload: "$GPRP,F88A5EB8D1E3,F008D1798BA4,-33,02010612FF0D0083BC290100AAAA3475630313040000,1657867835\r\\\n"  
  properties:  
    application: {}
```

Use Azure IoT Explorer

Or, Use Azure IoT Explorer to verify if the configuration works fine.

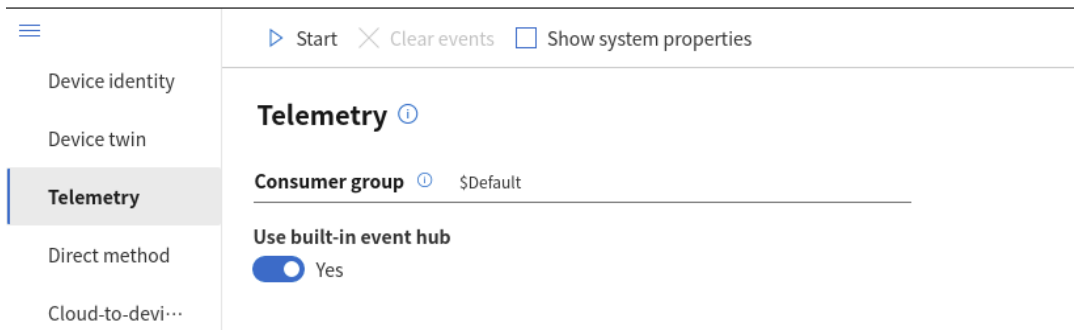
<https://docs.microsoft.com/zh-tw/azure/iot-pnp/howto-install-iot-explorer>

Use default EventHub to check the messages from devices to Cloud.

1. Login use IoT Hub connect string
2. Choice your IoT Hub -> IoT device -> Telemetry
Make sure "Use built-in event hub" is YES, click "Start" button to monitor the events



[Hubs](#) > [igstesthub](#) > [Devices](#) > [igstestdevice](#) > [Telemetry](#)



3. If everything works fine, the message will show up

- Device identity
- Device twin
- Telemetry
- Direct method
- Cloud-to-device message
- Module identity
- IoT Plug and Play comp...

▶ Start
✕ Clear events
 Show system properties

Telemetry ⓘ

Consumer group ⓘ \$Default

Use built-in event hub
 Yes

10:09:43 AM, July 08, 2020:

```
{
  "body": "$GPRP,291A78036961,F008D1798C84,-74,1EFF0600010920022919850BED8EA61A7FFC0E1E14F1640403BF95E",
  "enqueuedTime": "2020-07-08T02:09:43.315Z",
  "properties": {}
}
```

10:09:43 AM, July 08, 2020:

```
{
  "body": "$GPRP,0C61CFC1459F,F008D1798C84,-63,02010612FF0D0083BC2D0100AAAAFFFF000019040000\r\n",
  "enqueuedTime": "2020-07-08T02:09:43.299Z",
  "properties": {}
}
```

Revision History

DATE	REVISION	CHANGES
Feb 11, 2019	1	Initial release
Mar 28, 2019	2	Update reference documents
Dec 24, 2019	3	Update example for using X-509 certificates
Jul 22, 2021	4	Update for iGS03 series support
Jul 15, 2022	5	Update Azure-CLI usage for verification