Introduction

This application note provides a step-by-step setup to connect iGS01S/iGS02E 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.

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 security token, create device with authentication type “Symmetric key”.

<table>
<thead>
<tr>
<th>Device ID *</th>
<th>igs01s</th>
</tr>
</thead>
</table>

**Authentication type**

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

**Primary key**

Enter your primary key

**Secondary key**

Enter your secondary key

**Auto-generate keys**

- [ ]
After device created, you can get the symmetric key in device detail.

<table>
<thead>
<tr>
<th>Device ID</th>
<th>igs01s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primary Key</td>
<td>MOS9pHMRZ0P+0So78uhjhEQEvx8nHlBucSv41mB74gQ=</td>
</tr>
<tr>
<td>Secondary Key</td>
<td>..........................................................</td>
</tr>
<tr>
<td>Primary Connection String</td>
<td>..........................................................</td>
</tr>
<tr>
<td>Secondary Connection String</td>
<td>..........................................................</td>
</tr>
</tbody>
</table>

Then we need to generate the SAS token from the symmetric key for the MQTT connection. Please refer Security Token for detail about the token structure. And the link above provide 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+0So78uhjhEQEvx8nHlBucSv41mB74gQ= 1440

SharedAccessSignature
sr=igs-test.azure-devices.net%2Fdevices%2Figs01s&sig=12bF1jJ%2FQWg8XYJPzfne1vWN5HEp02VvIBDEcc1wx7I%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 “hello” message to Azure IoT using mosquitto 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%2FQWg8XYJPzfne1vWN5HEp02VvIBDEcc1wx7I%3D&se=1577324521"
--capath /etc/ssl/certs/ --tls-version tlsv1 -d -V mqttv311 -q 0 -m "hello"
Then, 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`
- MQTT PASSWORD: `<SAS Token of the Device>`
- Enable MQTT
- Select Azure-IoT-Hub RootCA
- Disable use certificate

![Configuration Interface](image-url)
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
```

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

```
$ 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'
2A276C509956C50BCEAE46D2B6D112D38BC95056E
```

Create 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**
  - 2A276C509956C50BCEAE46D2B6D112D38BC95056E

- **Connect this device to an IoT hub**
  - [Enable] [Disable]
Use mosquitto tool on PC to verify the Azure configuration. Whatever using the primary key or the secondary key should pass the authentication.

```
$ mosquitto_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'
```

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

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`
- Enable MQTTS
- Select Azure-IoT-Hub RootCA
- Enable use certificate
### Application

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Application</strong></td>
<td>MQTT Client</td>
</tr>
<tr>
<td><strong>Host/IP</strong></td>
<td>igs-test.azure-devices.net</td>
</tr>
<tr>
<td><strong>Port</strong></td>
<td>8883</td>
</tr>
<tr>
<td><strong>Publish Topic</strong></td>
<td>devices/igs01s/messages/ev</td>
</tr>
<tr>
<td><strong>Client ID</strong></td>
<td>igs01s</td>
</tr>
<tr>
<td><strong>Username</strong></td>
<td>igs-test.azure-devices.net/igs01s</td>
</tr>
<tr>
<td><strong>Password</strong></td>
<td>password</td>
</tr>
<tr>
<td><strong>MQTTSS</strong></td>
<td>Enable</td>
</tr>
<tr>
<td><strong>Root CA</strong></td>
<td>Azure-IoT Hub</td>
</tr>
<tr>
<td><strong>Use Certificate</strong></td>
<td>Enable</td>
</tr>
<tr>
<td><strong>Format Type</strong></td>
<td>plain-text</td>
</tr>
<tr>
<td><strong>Request Interval (in secs)</strong></td>
<td>1</td>
</tr>
<tr>
<td><strong>Drop reports while cache full</strong></td>
<td>✅</td>
</tr>
<tr>
<td><strong>Throttle Control (filter out redundant records)</strong></td>
<td>✅</td>
</tr>
</tbody>
</table>

[Save]  [Cancel]
Verify Configuration

Use Azure IoT Explorer to verify if the configuration works fine. [https://docs.microsoft.com/zh-tw/azure/iot-pnp/howto-install-iot-explorer](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

<table>
<thead>
<tr>
<th>Device Identity</th>
<th>Device twin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Telemetry</td>
<td></td>
</tr>
<tr>
<td>Direct method</td>
<td></td>
</tr>
<tr>
<td>Cloud-to-device</td>
<td></td>
</tr>
</tbody>
</table>

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

```
[10:09:43 AM, July 08, 2020:
{
  "body": "5GPRP_291A76093961,F008D1798C84,-74.1EFF06080109200229198508ED8EA61A7EFC0E3E14F1046403B93E
  "enqueuedTime": "2020-07-08T02:09:43.3152",
  "properties": {}
}
```

```
[10:09:43 AM, July 08, 2020:
{
  "body": "5GPRP_A061FC14596F,F008D1798C84,-63.02810612F060803BC2D0180A3FFFF80019046000\n"
  "enqueuedTime": "2020-07-08T02:09:43.2992",
  "properties": {}
}
```
# Revision History

<table>
<thead>
<tr>
<th>DATE</th>
<th>REVISION</th>
<th>CHANGES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feb 11, 2019</td>
<td>1</td>
<td>Initial release</td>
</tr>
<tr>
<td>Mar 28, 2019</td>
<td>2</td>
<td>Update reference documents</td>
</tr>
<tr>
<td>Dec 24, 2019</td>
<td>3</td>
<td>Update example for using X-509 certificates</td>
</tr>
</tbody>
</table>