Azure step by step

Introduction and limitation

1. For this demo, we are using mqtt protocol to communicate with Azure IOT Hub. You can access Communicate with your IoT hub using the MQTT protocol for more details.
2. We secure IOT Hub with X.509 certificate authentication, and we need to create X.509 CA Signed device in IOT Hub. Other security access methods like security token will not covered in this demo and not recommended by us. For more information, you can visit Set up X.509 security in your Azure IoT hub .
3. The demo requires two independent function block libraries: IT_Library and IIOT_Library . IT_Library will provide the DNS translation function and IIOT_Library will provide the mqtt connection, publish, subscribe, disconnection, reconnection and so on functions.
4. For the certificate concerned files, users need to put it to a proper place in plcnext device, and the basic design idea is like this:
1). Core function is provided by MQTT_Client function block, which will be used to build the fundamental mqtt communication path. The SSH related stuff will also be handled by it.
2). Though MQTT_Client plays a very important role, its configuration element is not a direct stuff we can get from Azure. So Azure_CertificateInfo function block is provided to work as a translator, who will translate the crude connection information we get from Azure to a internal one, which can be understood by MQTT_Client function block.

Steps

1. Create Azure IOT HUB on cloud side.

2. Get X.509 CA certificates, Register X.509 CA certificates to your IoT hub, Create an X.509 device for your IoT hub.

For more details of this step, you can refer to Set up X.509 security in your Azure IoT hub .

3. Download Visual studio code, and install Azure IoT Hub extension.

In this extension browse to the IOT HUB you created. And you can find the device you created in the cloud. And with this extension, you can either monitor D2C data or send C2D data to real device.

For more information, please refer to Use Azure IoT Tools for Visual Studio Code to send and receive messages between your device and IoT Hub.

4. Use winscp to transfer the certificate related files to /opt/plcnext/certs/.

In this demo, we have established the default certificate folder to /opt/plcnext/certs/, so if you want to try another place, do not forget to change the default path in PLCNEXT engineer project.

5. Download the demo project to device and enter the debug mode, there are two variable waiting for you to manually set true: config_en and Run.

6. Monitor data in the Visual Studio Code extension. Or send C2D data to the device and device should get this message.