# u-connectXpress

## **Azure Connectivity**

**Application note** 

### Abstract

This Application note provides information on how to configure and setup the connection for the Azure cloud services, using u-connectXpress software.



UBX-22039667 - R01 C1-Public

www.u-blox.com



## **Document information**

Title	u-connectXpress	
Subtitle	Azure Connectivity	
Document type	Application note	
Document number	UBX-22039667	
Revision and date	R01	24-Mar-2023
Disclosure restriction	C1-Public	

#### This document applies to the following products:

Product name	Software version
NINA-W131	2.1.x onwards
NINA-W132	2.1.x onwards
NINA-W151	All
NINA-W152	All
NINA-W156	3.1.x onwards
ODIN-W260	7.0.x onwards
ODIN-W262	7.0.x onwards
ODIN-W263	7.0.x onwards

u-blox or third parties may hold intellectual property rights in the products, names, logos, and designs included in this docum ent. Copying, reproduction, or modification of this document or any part thereof is only permitted with the express written permission of u-blox. Disclosure to third parties is permitted for clearly public documents only.

The information contained herein is provided "as is" and u-blox assumes no liability for its use. No warranty, either express or implied, is given, including but not limited to, with respect to the accuracy, correctness, reliability, and fitness for a particular purpose of the information. This document may be revised by u-blox at any time without notice. For the most recent documents, visit www.u blox.com.

Copyright © u-blox AG.



## Contents

Doc	umen	it information	2
Con	tents		3
1	Config	guring Azure IoT Hub	4
1.	1 Crea	ate a new IoT Hub	4
1.2	2 Con	figure a module with a X.509 CA Signed certificate	4
	1.2.1	Obtain the CA certificate	4
	1.2.2	Configure cloud server	4
	1.2.3	Prepare module certificates	7
	1.2.4	Install certificates on the module	8
	1.2.5	Connect to the cloud	9
1.:	3 Con	figure a module with a Symmetric Key/SAS Token	10
	1.3.1	Create the first device	10
	1.3.2	Connect to the cloud	11
1.4	4 Dev	rice Provisioning Service (DPS)	13
	1.4.1	Create and Prepare the Azure IoT Hub Device Provisioning Services	13
	1.4.2	Configure enrollment group	13
	1.4.3	Configure IoT device connection to Device Provisioning Service (DPS)	14
	1.4.4	Monitoring DPS status	16
1.	5 Inte	gration with Azure IoT Explorer	19
	1.5.1	IoT Hub connection	19
	1.5.2	View Devices	20
	1.5.3	Interact with a device	20
Арр	endix	<	
Α	Gloss	ary	
В	Monit	toring messages to/from the cloud	
в.	1 In A	zure IoT Hub	23
	B.1.1	Device-to-cloud	24
	B.1.2	Cloud-to-device	24
Rela	ated d	locuments	25
Rev	ision ł	history	
Con	tact		



## 1 Configuring Azure IoT Hub

Due to limitations in the maximum length of an AT command for NINA-W13 and NINA-W15 in u-connectXpress software prior to version 3.0, keep all names and identifiers as short as possible.

Identifiers and names for an IoT Hub should typically include:

- The name of the IoT Hub
- The Device Id of the devices
- The names of the certificates (CA and CC) as well as the client private key (PK).

## 1.1 Create a new IoT Hub

Unless one already exists, create a new IoT Hub.

- 1. Navigate to portal.azure.com and select More Services.
- 2. Search for IoT Hub and select **Create** in the pop-up window.
- 3. Create new or select an existing Resource group.
- 4. Choose your IoT hub name, such as "iot-uconnectxpress-dev". Microsoft suggest the naming convention "iot-<App Name>-<Environment>". The given name is publicly visible, so private metadata should not be included. Keep the IoT hub name as short as possible.
- 5. Step through the configuration wizard. Tags are optional private metadata which you may want or need to manage your hub especially as your company and your Azure cloud grows. If you are unsure, just add the tag "MyTag", and value "MyValue".
- 6. Once created, go to your IoT Hub (which Microsoft refer to in more general terms as a resource) and check that it is available under "All resources" from the home screen.

## **1.2** Configure a module with a X.509 CA Signed certificate

### 1.2.1 Obtain the CA certificate

Create a CA certificate to use when signing the certificates for each module. The procedures for generating a CA certificate using OpenSSL are described in u-connectXpress Wi-Fi security application note [4].

### 1.2.2 Configure cloud server

In the IoT Hub resource, you are required to verify that you have the necessary permissions to generate and sign certificates. To configure the cloud server, you must:

- Create the first device
- Sign a client side-certificate for the device
- Upload the CA used to sign the client-side certificate
- Verify the certificate used to sign the client-side certificate



## 1.2.2.1 Create the first device

- 1. Go to the IoT Devices tool in the IoT hub Explorer section. Select **New** or **Add** to register a new device.
- 2. Enter a unique Device ID. For example, enter the serial number of the device as "device1", or enter the module response to the AT+UMLA=2 command. If the NINA-W13/W15 module is configured with u-connectXpress software version 3.0 or earlier, keep the device ID as short as possible.
- 3. Set the Authentication type to "X.509 CA Signed".





4. Select **Save** to add your new device to the displayed list of IoT devices



### 1.2.2.2 Sign a client-side certificate for the module

Generate a client-side certificate for your module. See also the u-connectXpress Wi-Fi security application note [4]), where Common Name (CN) is set to the Device ID of the device.

This client-side certificate (CC) and its corresponding private key (PK) are also used later in the configuration.

### 1.2.2.3 Upload and verify the CA used to sign the client-side certificate

1. To upload the CA certificate created in Obtain the CA certificate, go to "Certificates" and select Add.



- 2. Select the certificate added in the previous step. To generate the verification code needed to generate a new certificate, select **Generate Verification Code** in the "Certificate Details" pane.
- 3. Generate the new client-side certificate using the same CA used in Obtain the CA certificate, but this time use the Verification Code as the Common Name (CN). To generate a client-side certificate, see the u-connectXpress Wi-Fi security application note, reference [4].





4. Upload the certificate generated in the previous step to the Verification Certificate in the Certificate Details pane and select "Verify". The STATUS of the CA certificate changes to Verified.

Micros	Soft Azure $P$ Search resource	s, services,	and docs			>_ 🛱	Q	ŝ	?	©			
»	Home > •••••••••••••••••••••••••••••••••••	- Certificat	tes										
+	Jot Hub	ıb-1 - (	Certificat	es								\$	×
		«	🕂 Add	Columns	Refresh								
	Zevents	*											
·= - ★	Settings			You can use this	tool to upload	and manage yo	our certifi	cates.				Z	
	📍 Shared access policies		_										
	O Pricing and scale		NAME	STAT	US	EXPIRY		SUBJECT	т	HUMBPRINT	CREATED		
8	Operations monitoring		rootCA	0 V	erified	Wed Nov 03	2021 1	ubx	6	1D1669396132C7D	Thu Jan 24 2019	12:03	
<i>«</i>	∃ <mark>→</mark> IP Filter												
2	🔎 Certificates												

## 1.2.3 Prepare module certificates

- Obtain the latest Azure IoT Hub server certificate. The certificate can be found in a .c-file in the certs directory of the Azure-iot-sdk-c repository: https://github.com/Azure/azure-iot-sdk-c/blob/master/certs/certs.c
- 2. Convert the c-file to a local Azure certificate file containing the DigiCert Baltimore Root certificate by copying the certificate information from certs.c to a text editor. Include the lines: ----BEGIN CERTIFICATE----- and -----END CERTIFICATE----- and then remove the quotation marks (") at the beginning and end of every line. Also remove the \r\n characters at the end of every line.
- 3. Save the file as, for example, azure-iot-baltimore-cert.pem.

azure-iot-baltimore-cert.pem - Notepad				_		×
<u>F</u> ile <u>E</u> dit F <u>o</u> rmat <u>V</u> iew <u>H</u> elp						
BEGIN CERTIFICATE						~
MIIDdzCCAl+gAwIBAgIEAgAAuTANBgkqhkiG9w0BAQUFADBaMQs	wCQYDVQQGEwJJ					
RTESMBAGA1UEChMJQmFsdGltb3JlMRMwEQYDVQQLEwpDeWJlclR	ydXN0MSIwIAYD					
VQQDEx1CYWx0aW1vcmUgQ31iZXJUcnVzdCBSb290MB4XDTAwMDU	xMjE4NDYwMFoX					
DTI1MDUxMjIzNTkwMFowWjELMAkGA1UEBhMCSUUxEjAQBgNVBAo	TCUJhbHRpbW9y					
ZTETMBEGA1UECxMKQ3liZXJUcnVzdDEiMCAGA1UEAxMZQmFsdG1	tb3JlIEN5YmVy					
VHJ1c3QgUm9vdDCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQo	CggEBAKMEuyKr					
mD1X6CZymrV51Cni4eiVgLGw41uOKymaZN+hXe2wCQVt2yguzmK	iYv60iNoS6zjr					
IZ3AQSsBUnuId9Mcj8e6uYi1agnnc+gRQKfRzMpijS3ljwumUNK	oUMMo6vWrJYeK					
mpYcqWe4PwzV9/1SEy/CG9VwcPCPwBLKBsua4dnKM3p31vjsufF	oREJIE9LAwqSu					
XmD+tqYF/LTdB1kC1FkYmGP1pWPgkAx9XbIGevOF6uvUA65ehD5	t/xXtabz50TZy					
dc93Uk3zyZAsuT31ySNTPx8kmCFcB5kpvcY670duhjpr13RjM71	oGDHweI12v/ye					
JIOQHQQNKNWNGJKCAWEAAANFMEMWHQYDVROOBBYEFOWdWICCR1J	MrPolVDaGezq1					
BE3WIBIGATUGEWEB/WQIMAYBAT8CAQMWDgYDVR0PAQH/BAQDAge	GMAUGCSQGSID3					
DQEBBQUAA4IBAQCFDF20509RdE1F0N27TyCIIA0992T9Lucw46Q						
jhrsewEQ7pE1YadTcp0w/y7cbMPEbbaE/Y++7PGiD8AC+DbSNzk	Flakvebi /oCr0					
Enn300WC47va072atciafC7Tn150CBPLhf1wbWa2V71k5h237vD	vnv6767fvllThz					
ksli4xaNmiTCa44Y3ek0Ee5+Nau0rz4w]Hr0Mz2n70/1/T6eYs9	HRCwRXhsdtTLS					
R9T41 tD+gdwyab617izV/OeBHRnDJEL gYzmp	Intemproba et Eb					
END CERTIFICATE						
4						>
	In 1 Col 1	100%	Windows (CDLD)	LITE	0	
	LIFT, COLT	100%	windows (CRLF)	UIF-	0	



## 1.2.4 Install certificates on the module

1. Upload the local Azure certificate file to the module as a trusted root (CA) certificate using s-center, or the command AT+USECMNG=0,0.

usic Connection 🤮 Bluetooth Settings 🌍 ۱ Connection Setup Data Mode Serial Settings	Wi-Fi Settings 🚯 Advanced Connection and Settings 🚛 User Defines Miscellaneous GPIO WI-Fi Certificate NFC	Ublox
Certificate and Private key import Trusted root (CA) azure-iot-baltimore-cert.pem Client Certificate (CC) Client Private Key (PK) List Use Certificate MD5 Upload Distructed root CA Erase Password	Filesystem list	Connection Status Bluetooth: - Address: - Wi-Fi - SSID - Channet: - RSSI - Network Status State: - Interface: - IP Address: - Subnet: - Gateway: -
ansole Window Log Window		Prim. DNS: - Sec. DNS: - IPv6: -
nntole Window Log Window I - GMR 2.1.0-017 N - UNLL:04CASEFD96E0 X W USECMNG-0, "azure-iot-baltimore-cert.pem", "ACI M - USECMNG-3	:6694A59C17E0D7915296B19706A6E4*	Prim. DNS - Sec. DNS - IPv6: - Store Factory Default Enable GL EVIC-ODIN-W2 via ST-LINK Software Update

For further information about AT+USECMNG, see the u-connect AT commands manual [1] and u-connectXpress user guide [3].

- 2. Upload the client certificate (CC) and the client private key (PK) generated in Sign a client-side certificate for the module using either:
  - AT commands AT+USECMNG=0,1 (client certificate) and AT+USECMNG=0,2 (private key)
  - s-center

ic Connection 🤪 Bluetooth Settings 🎧 WI-Fi Settings 🚯 Advanced Connection and Settings 🊛 User Defines		blox
Certificate and Private key import Trusted root (CA) azure-iot-baltimore-cert.pem Client Certificate (CC) Client Private Key (PK) Client.key.pem Ust Ust Use Certificate MDS Upload C.Client Private Key Erase Password Log Window Log Window	Connection St Bluetooth: - Address: - W-Fe - SSID: - BSSID: - BSSI	atus
c + USECMNG= 3	Store	Factory
A", "azure-iot-baltimore-cert.pem" C" "dient cer"	Default	Enable GU
, / unineen K JSECMNG:0.2. "client.kev.pem" "25E91BDDCS065411B5149A8E762E0324"	EVK-ODIN-	W2 via ST-LINK
( +USECMNG=3	Softwa	ire Update
A","azure-iot-baltimore-cert.pem"	_	A

As s-center requires a file suffix .crt, .cer, or .pem, the files might need to be renamed.



## 1.2.5 Connect to the cloud

- 1. Use s-center or the command AT+UWSC and AT+UWSCA to set up a network connection to the internet on the module. See the u-connectXpress user guide [3] for details.
- 2. To publish and subscribe to events, connect to Azure IoT Hub using the commands AT+UDCP and MQTT. For further information about the mqtt scheme, see the u-connectXpress MQTT Application With reference to the u-connectXpress MQTT application note [2], the format of the AT+UDCP command for Azure is as follows:

```
AT+UDCP=mqtt://iothubhostname:8883/?client=device_id&user=iothubhostname/device_id
&ca=server_cert_name&cert=device_cert_name&privKey=device_key_name
&pt=devices/device_id/messages/events/
&st=devices/device_id/messages/devicebound/#
```

		In the URL,	replace the	marked section	ns above, a	s shown below
--	--	-------------	-------------	----------------	-------------	---------------

Item	Description
iothubhostname	From the Hostname section in the Overview page in Azure IoT Hub
device_id	The Device ID set to the device in Azure IoT Hub
server_cert_name	The internal name given to the Azure loT Hub server certificate when uploaded to the module
device_cert_name	The internal name given to the client certificate when uploaded to the module
device_key_name	The internal name given to the client private key when uploaded to the module

#### Example

77

```
at+udcp=mqtt://iot-xxxx-test-hub-1.azure-devices.net:8883/?client=device1
&user=iot-xxxx-test-hub-1.azure-devices.net/device1
&ca=azure-iot-baltimore-cert.pem&cert=client.cer&privKey=client.key.pem
&pt=devices/device1/messages/events/
&st=devices/device1/messages/devicebound/#
```

- 3. Switch to Data mode and send data to/from the Azure IoT server; monitor the transferred data as described in Appendix B.1.
- The #wildcard is not supported by ODIN-W26-7.0.0, or NINA-W13/NINAW15 prior to v3.0.



## 1.3 Configure a module with a Symmetric Key/SAS Token

**SAS** Tokens are supported from on NINA-W13 and NINA-W15 starting with version 3.0.

If you have not earlier, Create a new IoT Hub.

## 1.3.1 Create the first device

- 1. Go to the IoT Devices tool in the IoT hub Explorer section and select **New** or **Add** to register a new device.
- 2. Enter a unique Device ID, for example "device2", the device's serial number, or the result of AT+UMLA=2 for your module.
- 3. Set the Authentication type to "Symmetric Key"

Create a device     Image: Create a device           <
Find Certified for Azure IoT devices in the Device Catalog
Find Certified for Azure IoT devices in the Device Catalog
Device ID * ①
device?
deviced v
Authentication type ①
Symmetric key X.509 Self-Signed X.509 CA Signed
Primary key 🛈
Enter your primary key
Secondary key ①
Enter your secondary key
Auto-generate keys ①
Connect this device to an IoT hub ①
Enable Disable
Parent device ①
No parent device
Set a parent device
Save

4. Select **Save** to include your new device in the list of IoT devices.

The device page on IoT Hub now contains the auto-generated primary and secondary keys, including the connection strings to use.



### 1.3.2 Connect to the cloud

1. Generate a SAS token using the algorithm defined by the code snippets included in the Microsoft Azure documentation at:

https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-devguide-security#security-tokens. The returned string is the complete SAS Token.

A python program which generates a SAS Token could be written as:

```
from base64 import b64encode, b64decode
from hashlib import sha256
from time import time
from urllib import parse
from hmac import HMAC
IOTHUBHOSTNAME = "iothubhostname"
PRIMARY_KEY = "primary_key"
def generate sas token(uri, key, policy name, expiry=3600):
    ttl = time() + expiry
    sign_key = "%s\n%d" % ((parse.quote_plus(uri)), int(ttl))
signature
    signature
                      =
                           b64encode(HMAC(b64decode(key), sign key.encode('utf-8'),
               sha256).digest())
    rawtoken = {
        'sr' : uri,
'sig' : signature,
'se' : str(int(tt
        'se'
               : str(int(ttl))
    }
    if policy_name is not None:
        rawtoken['skn'] = policy_name
    return 'SharedAccessSignature ' + parse.urlencode(rawtoken)
res = generate_sas_token(IOTHUBHOSTNAME, PRIMARY_KEY, "")
print(res)
```

In the above program, replace the *marked* items as follows:

Item	Description				
iothubhostname	From the Hostnar	ne section in the Overview page in Azure loT Hub.			
primary_key	Home > lagger and the device 2     A       device 2     A       Image: Some Image to Device for the device 2	Tield on the Device page in the Azure IoT Hub.		×	~
	Device ID	device2		Ð	
	Primary Key	bH/VjrcdSOJJtcwx94hePfV8EJ8iryZ/cXQ7e3i6gcDU=	-	D	I
	Secondary Key 🌘	•••••	٩	D	I
	Primary Connection String		۲	D	I
	Secondary Connection String		۲	D	I
	Enable connection to IoT Hub	Enable      Disable			
	Parent device 🔹	No parent device			
	Distributed Tracing (preview) 🕚 Learn more	Not configured			
	Module Identities Configurations	3			~ ~



#### Example

Invocation of:

```
generate_sas_token("iot-xxxx-test-hub-2.azure-devices.net",
"bH/VjcdSOJJtcwx94hePfV8EJ8ivyZ/cXQ7e3i6gcDU=", "")
```

#### Returns the SAS Token:

```
SharedAccessSignature sr=iot-xxxx-test-hub-2.azure-devices.net &sig=ix4nP1LaU%2FZg2Nk3OaBwJ0MikCPIxGxz2uiZ3KUxecM%3D&se=1591876214&skn=
```

The SAS Token contains both spaces, ampersands, equals, and percent signs.

- 2. Use s-center or the commands AT+UWSC and AT+UWSCA to set up a network connection to the internet on the module. For further information, see the u-connectXpress user guide [3].
- 3. Use the command AT+UDUV to set an URL Value to the SAS Token:

```
AT+UDUV=0, "SharedAccessSignature sr=iot-xxxx-test-hub-2.azure-devices.net
&sig=ix4nP1LaU%2FZq2Nk3OaBwJ0MikCPIxGxz2uiZ3KUxecM%3D&se=1591876214&skn="
```

- 4. Enable Wi-Fi and connect to an internet-connected AP.
- 5. Use the command AT+UDCP command and MQTT to connect to Azure IoT Hub, publish and subscribe to events. For further information about the mqtt: scheme, see the u-connectXpress MQTT application note [2]. The format of the AT+UDCP command for Azure using a SAS Token is as follows:

```
AT+UDCP="mqtt://iothubhostname:8883/?ca=<ca certificate>
&client=device_id
&user=iothubhostname/device_id
&passwd=%%url_value_index
&pt=devices/device_id/messages/events/
&st=devices/device_id/messages/devicebound/#"
```

#### T

#### In the URL, replace the *marked* sections above as follows:

Item	Description
iothubhostname	From the Hostname section in the Overview page in Azure IoT Hub.
device_id	The Device ID set to the device in Azure IoT Hub.
url_value_index	The URL Value index given as parameter to AT+UDUV.

#### Example

```
at+udcp="mqtt://iot-xxxx-test-hub-2.azure-devices.net:8883/
?ca=<ca certificate>
&client=device2
&user=iot-xxxx-test-hub-2.azure-devices.net/device2
&passwd=%%0
&pt=devices/device2/messages/events/
&st=devices/device2/messages/devicebound/#"
```

6. Switch to Data mode and send data to/from the Azure IoT server. Monitor the transferred data as described in Appendix B.1.



## 1.4 Device Provisioning Service (DPS)

To get started with the Azure IoT service, it is possible to use the IoT Hub device provisioning service (DPS) that enables zero-touch provisioning and configures the device connection to the cloud without requiring human intervention, allowing customers to configure multiple devices in a secure and scalable manner.

## 1.4.1 Create and Prepare the Azure IoT Hub Device Provisioning Services

When connected to the Azure DPS, the device is assigned to the IoT Hub created in Create a new IoT Hub. To create a new IoT Hub Device Provisioning Service using the Azure Portal:

- 1. In the Azure portal, select + Create a resource.
- 2. From the Categories menu, select Internet of Things then IoT Hub Device Provisioning Service.
- 3. Select **Create** and fill the required information accordingly to your Azure plan and resources.
- 4. Select **Review + Create** to validate your provisioning service.
- 5. Select Create.

To Link the IoT hub and your Device Provisioning Service:

- 1. In the **Settings** menu, select **Linked IoT hubs**.
- 2. Select + Add.
- 3. On the Add link to IoT hub panel, provide the following information:
- 4. Access Policy: Select iothubowner as the credentials for establishing the link with the IoT hub.
- 5. Select Save.

For mor information please visit Azure Quick Start [6].

### 1.4.2 Configure enrollment group

An enrollment group is a group of devices that share a specific attestation mechanism. Enrollment groups support X.509 certificate or symmetric key attestation. Devices in an X.509 enrollment group present X.509 certificates that have been signed by the same root or intermediate Certificate Authority (CA). The common name (CN) of each device, end-entity (leaf) certificate becomes the registration ID for that device.

The registration ID is used to uniquely identify a device registration with the Device Provisioning Service. The registration ID must be unique in the provisioning service ID scope. Each device must have a registration ID. The registration ID is a case-insensitive string (up to 128 characters long) of alphanumeric characters plus the special characters: '-', '.', '\_', ':'. The last character must be alphanumeric or dash ('-').

- 1. From your DPS in Azure portal, select the "Manage enrollments" tab, and then select "Add enrolment group" at the top.
- 2. Enter the following information in the "Add Enrolment Group" panel, and then select **Save**. The device certificate can be signed with the root CA or an intermediate CA, as shown below.



Il services > Azure IoT Hub Device Provisioning Services > IeoDPSs >
🕉 Add Enrollment Group 👘
🚽 Save
Group name *
dpsGroupName
ttestation Type ① Certificate Symmetric Key
oT Edge device ① True False
CA Certificate       Intermediate Certificate         rimary Certificate       ①
rootCA.pem
econdary Certificate ①
No certificate selected
elect how you want to assign devices to hubs ① Evenly weighted distribution
elect the IoT hubs this group can be assigned to: ①
leoDPS.azure-devices.net
Link a new IoT hub elect how you want device data to be handled on re-provisioning * ①
Re-provision and migrate data
levice Twin is only supported for standard tier IoT hubs. Learn more about standard tier. nitial Device Twin State
<pre>{     "tags": {},     "properties": {         "desired": {}     } }</pre>

See also "Create an enrolment group" in the Provision multiple X.509 devices tutorial [5].

## 1.4.3 Configure IoT device connection to Device Provisioning Service (DPS)

For X.509-based attestation, the registration ID is set to the common name (CN) of the device certificate. For this reason, the common name must adhere to the registration ID string format.

1. Create client private key and CSR:

\$ openssl req -newkey rsa:2048 -nodes -keyout nina.key.pem -out nina.csr

- The common name (CN) of each device's end-entity (leaf) certificate becomes the registration ID for that device.
- 2. Sign the CSR with the client root CA registered on Azure as described in Upload and verify the CA used to sign the client-side certificate, for example:

```
$ openssl x509 -req -in nina.csr -CA rootCA.pem -CAkey rootCA.key -CAcreateserial -out
nina.pem -days 500 -sha256
```



- 3. Upload the client Private Key and the client certificate created above to the module as described in section Install certificates on the module.
- 4. Use the command AT+UDCP command and MQTT to connect to Azure IoT Hub Device Provisioning Service (DPS), publish and subscribe to events.

For further information about the mqtt: scheme, see the u-connectXpress MQTT application note [2]. The format of the AT+UDCP command for Azure using a SAS Token is as follows:

```
AT+UDCP="mqtt://iothubhostname:8883/?
client=registration_id
&user=idScope/registrations/registration_id/api-version=2019-03-31
&pt=$dps/registrations/PUT/iotdps-register/?$rid={request_id}
&st=$dps/registrations/res/#"
&encr=3
&cca=<ca certificate>
&ccrt=<device certificate>
&privKey=<device private key>
```

In the URL, replace the *marked* sections above as follows:

Item	Description
iothubhostname	From the Hostname section in the Overview page in Azure IoT Hub.
registration_id	The Device ID set to the device in Azure IoT Hub.
idScope	is the ID Scope from the Azure IoT Hub Device Provisioning Service.

#### Example

The example pseudo-code below shows the DPS flow using u-blox u-connectXpress. It is assumed that the module is provisioned with the certificates and is already connected to the Wi-Fi with internet connection:

```
AT+UDCP=mqtt://global.azure-devices-provisioning.net:8883/?
client=ninadps
&user=OneOO67E45C/registrations/ninadps/api-version=2019-03-31
&pt=$dps/registrations/PUT/iotdps-register/?$rid={request id}
&st=$dps/registrations/res/#
&keepAlive=60
&encr=3
&ca=ca.pem
&cert=ninadps.pem
&privKey=ninadps.key.pem
+UDCP:1
OK
+UUDPC:1,2,6,0.0.0,0,40.113.176.170,8883
ATO1
OK
// Publish the following message
"{\"payload\":\"\",\"registrationId\":\"registration id\"}"
{"operationId":"5.b084dab098f0a900.5f5b44b2-ae20-4957-bd06-
```

40f9e8dcec3f", "status": "assigning"}





## 1.4.4 Monitoring DPS status

To poll the DPS status, it is possible to publish to the following topic, where <code>operationId</code> is returned on the previous message. For example:

```
$dps/registrations/GET/iotdps-get-
operationstatus/?$rid={request_id}&operationId=5.b084dab098f0a900.5f5b44b2-ae20-4957-bd06-
40f9e8dcec3f
```

This example demonstrates the process using u-connectXpress Data Mode. Since it is not possible to start a new peer while in Data Mode, it is necessary to change to Command mode by sending the escape sequence, +++, and create a new peer. Because the topic has an escape character "&", the url needs to be used with a placeholder:

```
AT+UDUV=0,"$dps/registrations/GET/iotdps-get-
operationstatus/?$rid={request_id}&operationId=5.b084dab098f0a900.5f5b44b2-ae20-4957-bd06-
40f9e8dcec3f"
AT+UDCP=mqtt://global.azure-devices-
provisioning.net:8883/?client=ninadps&user=0ne0067E45C/registrations/ninadps/api-
version=2019-03-
31&pt=%%0&st=$dps/registrations/res/#&keepAlive=60&encr=3&ca=ca.pem&cert=ninadps.pem&privK
ey=ninadps.key.pem
+UUDPC:1,2,6,0.0.0.0,0,40.113.176.170,8883
AT01
```

```
OK
```

1. Publish the following message "get operationstatus" to the status topic.

```
{\"operationId\":\"5.54dcdee4e6de2c9a.26a7ed00-28fd-4e98-bb5c-
b464a00262fd\",\"status\":\"assigned\",\"registrationState\":{\"x509\":{\"enrollmentGr
oupId\":\"enrolGroup\"},\"registrationId\":\"dpsdev\",\"createdDateTimeUtc\":\"2022-
07-01T16:46:50.830016Z\",\"assignedHub\":\"leoDPS.azure-
devices.net\",\"deviceId\":\"dpsdev\",\"status\":\"assigned\",\"substatus\":\"initialA
ssignment\",\"lastUpdatedDateTimeUtc\":\"2022-07-
01T16:46:51.0919787Z\",\"etag\":\"IjhkMGJIMTE3LTAwMDAtMDEwMC0wMDAwLTYyYmYyNGZiMDAwMCI=
\"}}
```

2. When the message "status": "assigned" is received, check that the device has been transferred to the configured IoT Hub in **Devices** under the **Azure IoT Hub**.



In this example, "assignedHub": "leoDPS.azure-devices.net":

Home > Recent > leoDPS								
leoDPS   Devices	×	۶ ···						
₽ Search (Ctrl+/)	~	View, create, delete, and update	devices in your IoT Hub.					
X Overview	^	Device name						
Activity log		enter device ID						
😤 Access control (IAM)		Find devices	Find devices					
🗳 Tags		🕂 Add Device 💍 Refresh	+ Add Device 🕐 Refresh 📋 Delete					
Diagnose and solve problems								
🗲 Events		Device ID	Status		Last Status Update	Authentication Type	Cloud to Device Message Count	
O Pricing and scale		ninados	Enabled			SelfSigned	0	
Device management			LINDICO			Schoighea	•	
Devices								
IoT Edge								
😤 Configurations								
🧼 Updates								
🔎 Queries								
Hub settings								
<ul> <li>Built-in endpoints</li> </ul>								
😢 Message routing								
🕒 File upload								
- Failover								
Properties								
🔒 Locks								

The following example includes the log from the DPS process, using NINA-W15 and the python script example found at https://github.com/u-blox/u-connectXpress\_azure\_device\_provisioning\_services.

```
$ python .\azure_dps.py COM44
com44 open
18ms -> AT+UFACTORY
 400ms <- AT+UFACTORY
OK
401ms -> AT+CPWROFF
416ms <- AT+CPWROFF
OK
2804ms -> AT+USECMNG=0,0,ca.pem,1261
 3836ms <- AT+USECMNG=0,0,ca.pem,1261
>
3836ms -> ----BEGIN CERTIFICATE----
. . .
----END CERTIFICATE----
5859ms <- +USECMNG:0,0,"ca.pem","ACB694A59C17E0D791529BB19706A6E4"
OK
5860ms -> AT+USECMNG=0,1,cert.pem,1224
 6923ms <- AT+USECMNG=0,1,cert.pem,1224
>
6923ms -> ----BEGIN CERTIFICATE----
. . .
----END CERTIFICATE----
8938ms <- +USECMNG:0,1,"cert.pem","9A1F95B8D512FA0ED075DD2B794ECB2A"
OK
8941ms -> AT+USECMNG=0,2,key.pem,1704
10033ms <- AT+USECMNG=0,2,key.pem,1704
>
10034ms -> ----BEGIN PRIVATE KEY-----
. . .
----END PRIVATE KEY-----
12057ms <- +USECMNG:0,2,"key.pem","695D5D1EBB1DDBD0C6F2C128BAEB7F34"
OK
12057ms -> ATO2
12067ms <- ATO2
```



```
OK
14076ms -> 0xAA00120044 'AT+UWSC=0,2,SSID\r' 0x55
 14089ms AT response: OK
 14091ms -> 0xAA00100044 'AT+UWSC=0,5,2\r' 0x55
 14105ms AT response: OK
14106ms -> 0xAA001F0044 'AT+UWSC=0,8,PASSWORD\r' 0x55
14121ms AT response: OK
 14122ms -> 0xAA000F0044 'AT+UWSCA=0,3\r' 0x55
14153ms AT response: OK
 17017ms AT event: +UUWLE:0,802AA8035ADE,11
 17065ms AT event: +UUNU:0
19092ms AT event: +UUNU:0
22100ms -> 0xAA011F0044 'AT+UDCP=mqtt://global.azure-devices-
provisioning.net:8883/?client=ninadps&user=OneOO67E45C/registrations/ninadps/api-
version=2019-03-31&pt=$dps/registrations/PUT/iotdps-
register/?$rid={request id}&st=$dps/registrations/res/#&encr=3&keepAlive=60&ca=ca.pem&cert
=cert.pem&privKey=key.pem\r' 0x55
22440ms AT response: +UDCP:2
OK
24712ms Connect event IPv4
       Channel id: 0
 25726ms AT event: +UUDPC:2,2,6,0.0.0.0,0,20.43.44.164,8883
 25726ms -> 0xAA002C0036 '\x00{"payload":"","registrationId":"ninadps"}' 0x55
 26168ms Data event:
        Channel id: 0
Data: {"operationId":"5.b084dab098f0a900.076859d0-7180-40b8-9d06-
bf611e398ec0","status":"assigning"}
26169ms -> 0xAA00950044 'AT+UDUV=0,$dps/registrations/GET/iotdps-get-
operationstatus/?$rid={request id}&operationId=5.b084dab098f0a900.076859d0-7180-40b8-9d06-
bf611e398ec0\r' 0x55
 26197ms AT response: OK
 26197ms -> 0xAA00CD0044 'AT+UDCP=mqtt://global.azure-devices-
provisioning.net:8883/?client=ninadps&user=OneOO67E45C/registrations/ninadps/api-
version=2019-03-31&pt=%%0&encr=3&keepAlive=60&ca=ca.pem&cert=cert.pem&privKey=key.pem\r'
0x55
26234ms AT response: +UDCP:4
OK
 26235ms Connect event IPv4
       Channel id: 1
 27245ms AT event: +UUDPC:4,2,6,0.0.0.0,0,20.43.44.164,8883
 27245ms -> 0xAA00160036 '\x01get operationstatus' 0x55
 27466ms Data event:
        Channel id: 0
        Data: {"operationId":"5.b084dab098f0a900.076859d0-7180-40b8-9d06-
bf611e398ec0","status":"assigned","registrationState":{"x509":{"enrollmentGroupId":"enrolG
roup"}, "registrationId": "ninadps", "createdDateTimeUtc": "2022-07-
04T08:28:35.8694014Z", "assignedHub": "leoDPS.azure-
devices.net", "deviceId": "ninadps", "status": "assigned", "substatus": "initialAssignment", "las
tUpdatedDateTimeUtc":"2022-07-
04T08:28:36.154464Z", "etag": "IjYwMDA4OTJjLTAwMDAtMDEwMCOwMDAwLTYyYzJhNGI0MDAwMCI="}}
Successfully Assigned
 27468ms -> 0xAA000D0044 'AT+CPWROFF\r' 0x55
27482ms AT response: OK
```



## 1.5 Integration with Azure IoT Explorer

IoT Explorer is a tool provided by Azure to set up the connection and monitoring of the module to Azure Cloud applications. The software is provided as a free download and is available at https://github.com/Azure/azure-iot-explorer/releases.

The main features of the Azure IoT Explorer include:

- Simple Azure connection configuration
- Device twin and direct method functions
- Real-time Azure monitoring and alarm management
- Advanced diagnostics tools

For more information about IoT Explorer functions, see the official Azure IoT Explorer page.

## 1.5.1 IoT Hub connection

1. The first time that Azure IoT Explorer is executed, the application is prompted for the user's IoT hub connection string. Provide the connection string.

Azure IoT Explorer (preview)			- 🗆 ×
File Edit View Window Help			
Azure IoT Explorer (previ	ew)	Notifications	🔅 Settings
Home > IoT hubs			
=	+ Add connection		
윰 loT hubs			
	iothub-cel-tm 🖍 🗎		
Notification Center	Host name		
La rocalcation center	iothub-cel-tm.azure-devices.net		
	Shared access policy name		
	iothubowner		
	Shared access policy key		
	••••••		
	Connection String		
	·······		
	ightarrow View devices in this hub		

2. Select Connect.



## 1.5.2 View Devices

After the tool connects to the IoT hub, it displays the **Devices** list page that lists the device identities registered with the IoT hub. The user can select any entry in the list to see more information.

To view devices from the Devices list page:

- 1. Select **New** to register a new device with the Azure IoT hub. Then enter a **Device ID**. Use the default settings to automatically generate authentication keys and enable the connection to the IoT hub.
- 2. Select a device and then select **Delete** to delete a device identity. Review the device details before completing this action to check if this is the correct device identity to delete.

	evices				Notrications	💿 Seti
New ORefresh 📄 Delete	,c → (¥ Add	query parameter)				
Device ID	Status	Connection st	Authenticatio	Last status up	loT Plug and	Edge devi
—	Enabled	Disconnected	SelfSigned			

## 1.5.3 Interact with a device

To interact with a device:

1. On the **Devices** list page, select a value in the **Device ID** column to view the detail page for the registered device. For each device, there are two main sections: **Device** and **Digital Twin**.

	🖾 Save – 🔍 Manage knys 🗸	
Device identity	Device identity	
Device twin	Device identity	
Telemetry	Device ID ©	
( Discourses)		
> Direct method	This device is being authenticated using an X.509 certificate.	
Cloud-to-device messa	Primary thumbprint 0	
Module identities		•
6	Secondary thumbprint 💿	
P IoT Plug and Play com		۵ (
	Connection string (0)	
		•



2. The user can access the device twin information on the Device twin tab





## Appendix

## A Glossary

Abbreviation	Definition			
API	Application Programming Interface			
CA	Certificate Authority			
CN	Common Name			
CSR	Certificate Signing Request			
loT	Internet of Things			
MQTT	Message Queuing Telemetry Transport			
TLS	Transport Layer Security			

Table 1: Explanation of the abbreviations and terms used



## B Monitoring messages to/from the cloud

## **B.1** In Azure IoT Hub

The Device Explorer tool is used to monitor messages between the device and the Azure IoT Hub. A pre-built version of the Device Explorer for Windows can be downloaded from: https://github.com/Azure/azure-iot-sdk-csharp/releases/tag/2019-9-11.

In this URL, scroll down for the SetupDeviceExplorer.msi

1. In the Device Explorer tool, go to the **Configuration** tab and add the Connection String for your Hub. The Connection String can be found in the IoT Hub.



2. Go to Settings / Shared access policies and select the Policy "iothubowner".

Micro	psoft Azure $P$ Search resources, ser	es, and docs >_ 🗤 🛱 🗘 🏟 ? 😳	
»	Dashboard > manufacture hub-1	hared access policies iothubowner	×
+	💦 📷 hub-1	Shared access policies	
<b>A</b>		Add	na More
	D Search (Ctn+/)	Access policy pam	
≣	🕅 Overview	In Thus uses permissions to grant access to each In Thus	c
- * -	Activity log	Ior hub based on functionality.	
	💒 Access control (IAM)	Registry read	0
<b>(*)</b>	🛷 Tags		0
٢	🗲 Events	POLICY PERM	ct 🛛
<i>«</i>	Settings	iothubowner regist	† <b>0</b>
2	Shared access policies	service servic	
2	O Pricing and scale	device devic	
<b>Q</b>	Operations monitoring	Shared access keys	
<b>\</b>	📑 + IP Filter		
	🔎 Certificates	registrykeadwirite regist Secondary key 🕥	
<b>~~&gt;</b>	e Built-in endpoints	Connection string—pri	mary key 🚯
	E Properties	Connection string—see	condary key
0	Locks	HostName=	-hub-1.az



### B.1.1 Device-to-cloud

1. In the Device Explorer tool, go to the Data tab and select the Device ID of the device to monitor (for example "device1")

Configuration	Managem	ent Data	Messages To	Device	Call Method on Device	
Monitoring						
Event Hub	p:	iihu	ıb-1			
Device ID	device1					Ŧ
Start Time: 02/27/2019 14:51:19						
Consume	r Group: \$	Default			Enable	
Mon	itor		Cancel		Clear	Show system properties
Event Hub D	ata					
Receiving e 2019-02-27 1	vents 4:51:22> De	evice: [devi	ce1], Data:[Hello	from OE	DIN-W2]	

#### 2. Select Monitor.

### B.1.2 Cloud-to-device

- 1. In the Device Explorer tool, go to the "Message to Device" tab and select the Device ID of the device to send message to (for example "device1").
- 2. Type a message in the Message text box and select **Send**.

Device Explorer To	win		. <b> x</b>				
Configuration N	Configuration Management Data Messages To Device Call Method on Device						
Send Messag	Send Message to Device:						
loT Hub:	loT Hub:						
Device ID:	device1		•				
Message:	Hello from Azure						
	Add Time Stamp Monitor Feedbar	ck Endpoint					
Properties	System Properties						
Кеу	Key Value						
•	•						
Ser	Send Clear						
Output	Output						
Sent to Device	Sent to Device ID: [device1], Message:"Hello from Azure", message Id: 46bbb9da-e439-432d-a55b-4bf02c5304c9						

3. Monitor the Cloud-to-Device messages.



## **Related documents**

- [1] u-connectXpress AT commands manual, UBX-14044127
- [2] u-connectXpress MQTT application note, UBX-19005066
- [3] u-connectXpress user guide, UBX-16024251
- [4] u-connectXpress Wi-Fi security application note, UBX-20012830
- [5] Tutorial: Provision multiple X.509 devices using enrollment groups
- [6] Quickstart: Set up the IoT Hub Device Provisioning Service with the Azure portal

## **Revision history**

Revision	Date	Name	Comments
R01	24-Mar-2023	Idas	Initial release.

For product change notifications and regular updates of u-blox documentation, register on our website, www.u-blox.com.

## Contact

For further support and contact information, visit us at www.u-blox.com/support.