MQTT

Connecting to the Mosquitto MQTT Broker

Vers. 1.0 – Jul 2021

1. Introduction

All of Infinite’s devices that support the MQTT protocol, are capable to connect to any local or remote MQTT Broker. Mosquitto is an open source message broker that implements the MQTT protocol. It is lightweight and suitable for use on all devices ranging from low power single board computers to full servers.

This document is a brief how-to guide for all device communications between Infinite’s devices and the Mosquitto MQTT Broker.

2. Installing the Mosquitto MQTT Broker to your server

Installation is a straightforward procedure. Simply download the package (x64 or x32) from here and follow the Eclipse Mosquitto Setup.

Once installed, open a Command Prompt or PowerShell window in the installed directory and type “mosquitto -c mosquitto.conf -v”. This command starts the Mosquitto Broker with the settings that are configured in the mosquitto.conf file, in verbose mode. The .conf file is where the TLS, listening port, IPv and many other options are set.

2.1 Configuring TLS on Mosquitto

For a more secure connection we offer TLS support which is the standard in MQTT.

On the Mosquitto side we need to create the Broker certificates and keys. We do that with the commercial-grade TLS toolkit openssl. The easiest way to do that is to simply install git on your computer and locate the openssl.exe file in this directory: `C:\Program Files\Git\usr\bin\openssl.exe.`

Open a Command Prompt or PowerShell window in the above directory and type the following commands to create the server certificates and keys:

genrsa -des3 -out ca.key 2048 - creates a key pair for the CA

req -new -x509 -days 1826 -key ca.key -out ca.crt - creates a certificate for the CA

gensra -out server.key 2048 - creates a server key pair

x509 -req -in server.csr -CA ca.crt -CAkey ca.key -CAcreateserial -out server.crt -days 36 - creates the server certificate

(These commands are for testing purposes and should be adjusted for different requirements.)

You can now edit the .conf file with the directories of the above files along with your preferred listener port (8883 for TLS). TLS version should be 1.2.

Follow this detailed tutorial on how to create the server certificates and keys as well as edit the .conf file.

2.2 Creating the Device Certificate and Private Key

Using openssl, we create the device (client) certificate and key using one of the files that we created previously. In an openssl window type the following commands to create the client certificate and private key:

gensra -out client.key 2048 - creates a client private key

req -new -out client.csr -key client.key - creates a certificate request

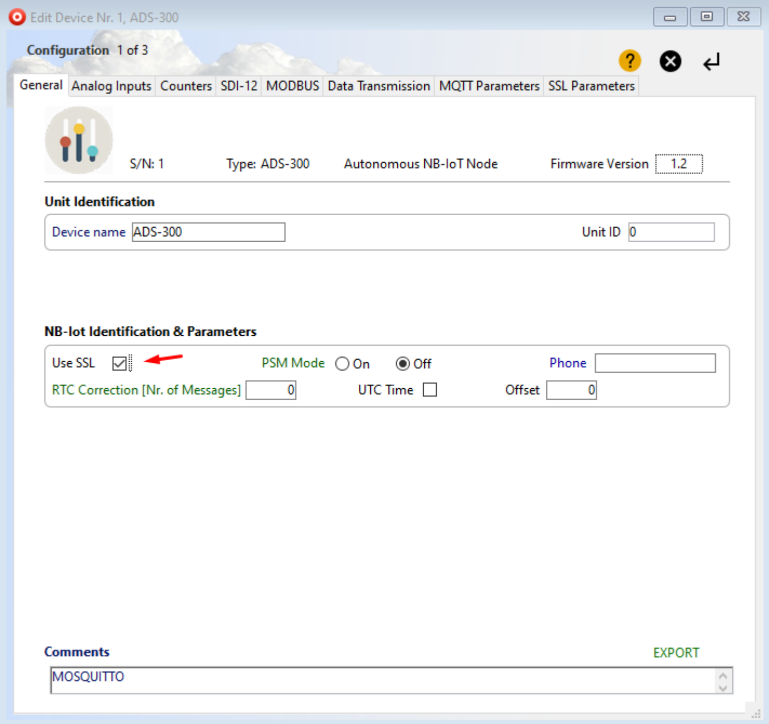
x509 -req -in client.csr -CA ca.crt -Cakey ca.key -CAcreateserial -out client.crt -days 360 - creates client certificate

(These commands are for testing purposes and should be adjusted for different requirements.)

Follow this detailed tutorial on how to create the client certificate and key.

3. Device Configuration with WA Manager

In the Edit Device window in WA Manager, tick the Use SSL box.



S/N: 1 Type: ADS-300 Autonomous NB-IoT Node Firmware Version 1.2

Unit Identification

Device name ADS-300 Unit ID 0

NB-Iot Identification & Parameters

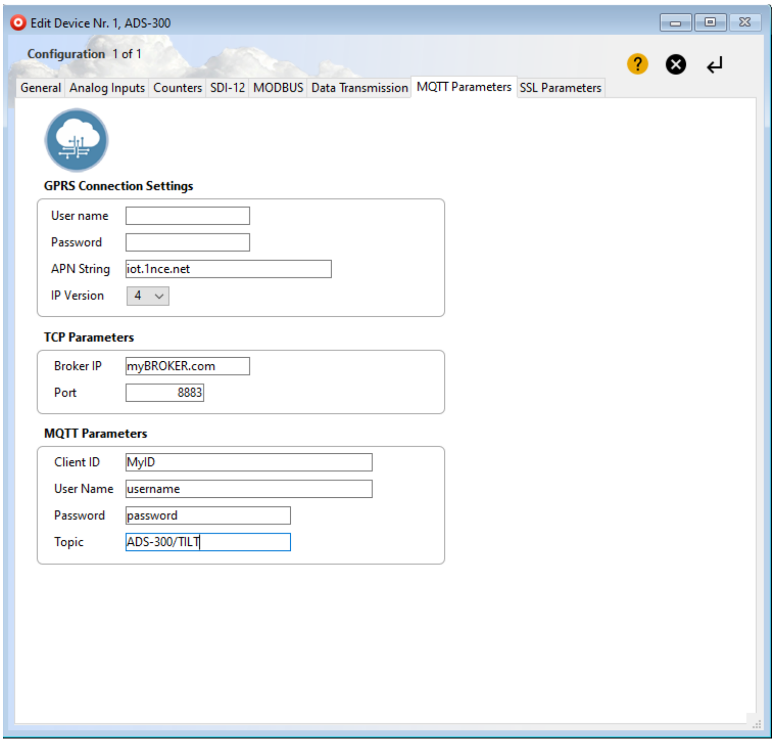
Use SSL ☑ PSM Mode ○ On ● Off Phone

RTC Correction [Nr. of Messages] 0 UTC Time □ Offset 0

Comments

MOSQUITTO

Next, we configure the MQTT parameters.



GPRS Connection Settings

User name
Password
APN String: iot.1nce.net
IP Version: 4

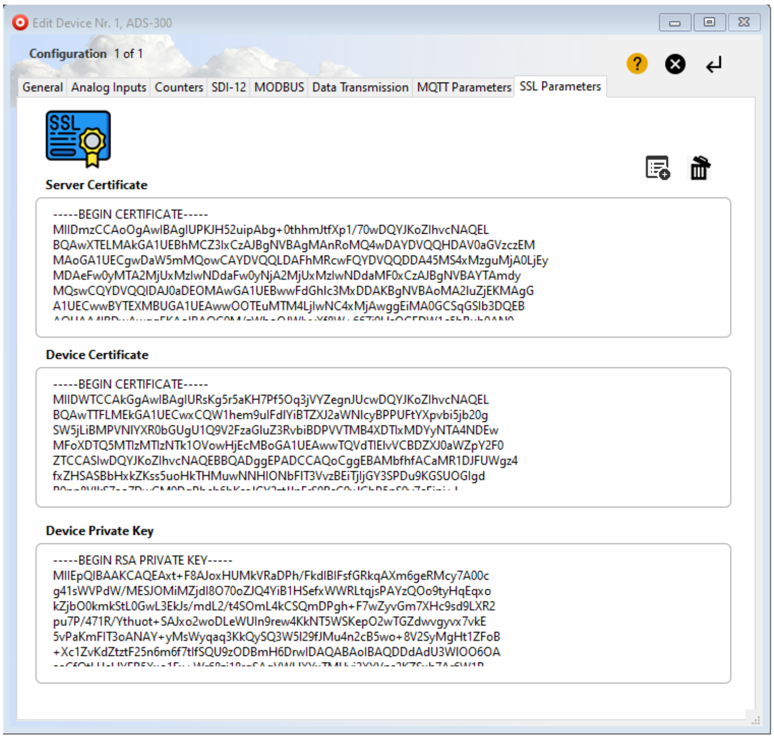
TCP Parameters

Broker IP: myBROKER.com
Port: 8883

MQTT Parameters

Client ID: MyID
User Name: username
Password: password
Topic: ADS-300/TILT

Lastly, in the SSL Parameters tab, we copy and paste the three files needed for the TLS communication: Server Certificate (CA), Device Certificate and Device Private Key.



Server Certificate

```
-----BEGIN CERTIFICATE-----
MIIDmzCCAoOgAwIBAgIUPKJH52uipAbg+0thhmJtfXp1/70wDQYJKoZIhvcNAQEL
BQAwXTELMAkGA1UEBhMCZ3IxCzAJBgNVBAgMAnRoMQ4wDAYDVQQHDAV0aGVzczEM
MAoGA1UECgwDaW5mMQowCAYDVQQLDAFhMRcwFQYDVQQDDA45MS4xMzguMjA0LjEy
MDAeFw0yMTA2MjUxMzIwNDdaFw0yNjA2MjUxMzIwNDdaMF0xCzAJBgNVBAYTAmdy
MQswCQYDVQQIDAJ0aDEOMAwGA1UEBwwFdGhlc3MxDDAKBgNVBAoMA2luZjEKMAgG
A1UECwwBYTEXMBUGA1UEAwwOOTEuMTM4LjIwNC4xMjAwggEiMA0GCSqGSIb3DQEB
```

Device Certificate

```
-----BEGIN CERTIFICATE-----
MIIDWTCCAkGgAwIBAgIURsKg5r5aKH7Pf5Oq3jVYZegnJUcwDQYJKoZIhvcNAQEL
BQAwTTFLMEkGA1UECwxCQW1hem9uIFdlYiBTZXJ2aWNlcyBPPUFtYXpvbi5jb20g
SW5jLiBMPVNlYXR0bGUgU1Q9V2FzaGluZ3RvbiBDPVVTMB4XDTIxMDYyNTA4NDEw
MFoXDTQ5MTIzMTIzNTk1OVowHjEcMBoGA1UEAwwTQVdTIElvVCBDZXJ0aWZpY2F0
ZTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAMbfhfACaMR1DJFUWgz4
fxZHSASBbHxkZKss5uoHkTHMuwNNHIONbFIT3VvzBEiTjljGY3SPDu9KGSUOGlgd
```

Device Private Key

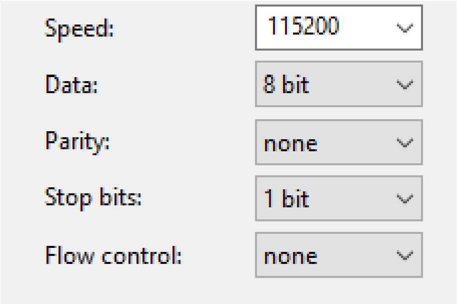
```
-----BEGIN RSA PRIVATE KEY-----
MIIEpQIBAAKCAQEAxt+F8AJoxHUMkVRaDPh/FkdIBIFsfGRkqAXm6geRMcy7A00c
g41sWVPdW/MESJOMiMZjdI8O70oZJQ4YiB1HSefxWWRLtqjsPAYzQOo9tyHqEqxo
kZjbO0kmkStL0GwL3EkJs/mdL2/t4SOmL4kCSQmDPgh+F7wZyvGm7XHc9sd9LXR2
pu7P/471R/Ythuot+SAJxo2woDLeWUIn9rew4KkNT5WSKepO2wTGZdwvgyvx7vkE
5vPaKmFIT3oANAY+yMsWyqaq3KkQySQ3W5I29fJMu4n2cB5wo+8V2SyMgHt1ZFoB
+Xc1ZvKdZtztF25n6m6f7tIfSQU9zODBmH6DrwIDAQABAoIBAQDDdAdU3WIOO6OA
```

The Server Certificate is the ca.crt file we created, the Device Certificate is the client.crt file and the Device Private Key is the client,key file. These files should be first opened with Notepad++ and their contents should be copy and pasted in the above tab. All files must be PEM formatted.

4. Load Certificates via Terminal

Alternatively, the certificates can also be loaded via a terminal program of your choice. This example uses Tera Term.

The serial port settings are shown in the image below.

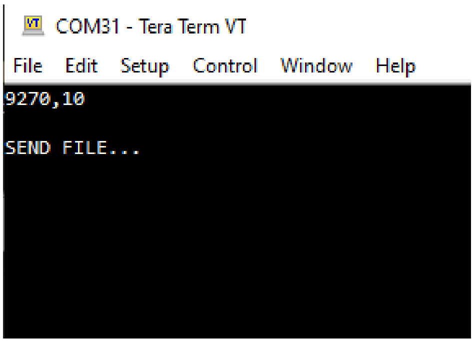


Speed: 115200
Data: 8 bit
Parity: none
Stop bits: 1 bit
Flow control: none

The commands for sending each of the certificates are shown in the table below.

9270	Send SSL Certificates	cmd, n	n: 10: Root Certificate, 20: Device Certificate, 30: Device Private Key

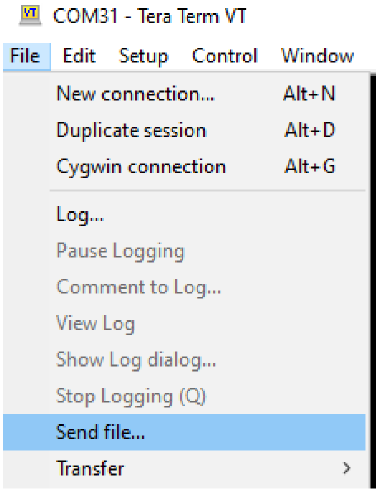
So, for sending the Root Certificate we should enter the command 9270,10 in the terminal.

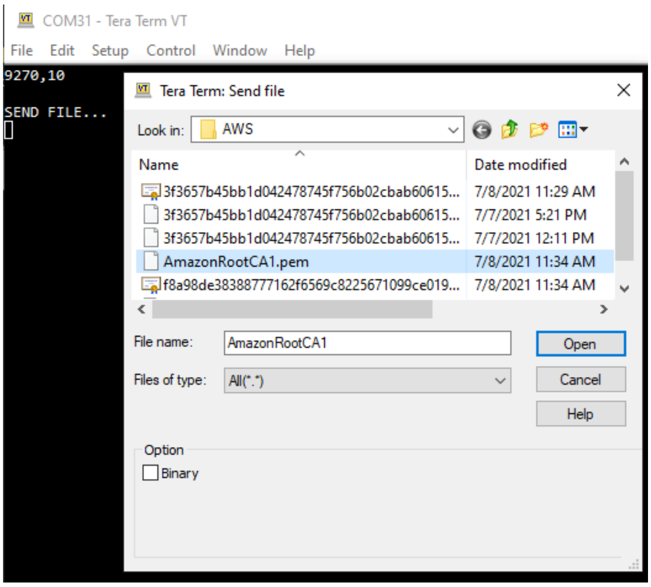


```
9270,10

SEND FILE...
```

Then, send the appropriate file.





```
9270,10

SEND FILE...
```

Look in: AWS

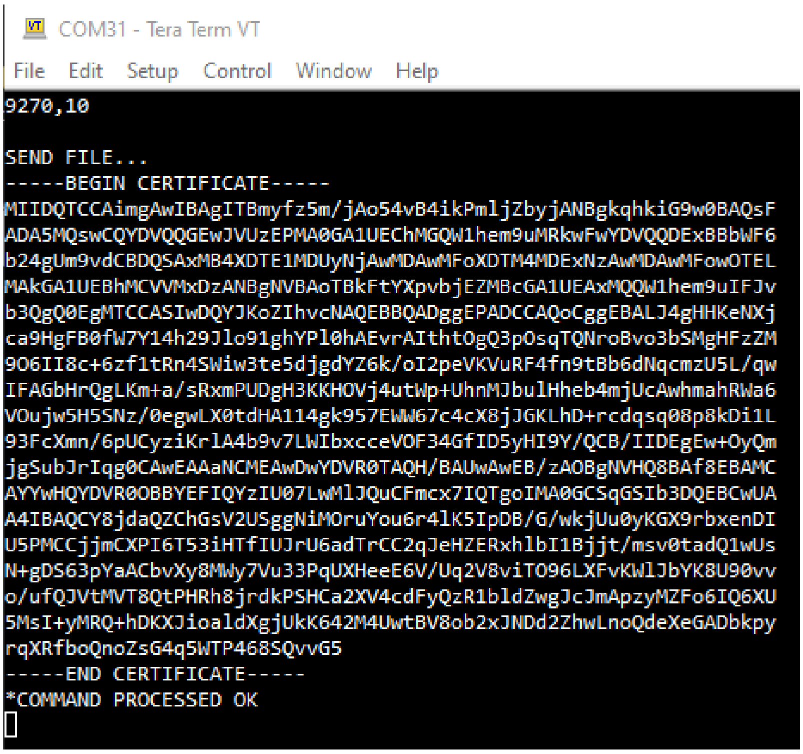
Name	Date modified
3f3657b45bb1d042478745f756b02cbab60615...	7/8/2021 11:29 AM
3f3657b45bb1d042478745f756b02cbab60615...	7/7/2021 5:21 PM
3f3657b45bb1d042478745f756b02cbab60615...	7/7/2021 12:11 PM
AmazonRootCA1.pem	7/8/2021 11:34 AM
f8a98de38388777162f6569c8225671099ce019...	7/8/2021 11:34 AM

File name: AmazonRootCA1

Files of type: All(*.*)

□ Binary

And enter the special character *. This is achieved by pressing Ctrl+8.



```
9270,10

SEND FILE...
-----BEGIN CERTIFICATE-----
MIIDQTCCAimgAwIBAgITBmyfz5m/jAo54vB4ikPmljZbyjANBgkqhkiG9w0BAQsF
ADA5MQswCQYDVQQGEwJVUzEPMA0GA1UEChMGQW1hem9uMRkwFwYDVQQDExBBbWF6
b24gUm9vdCBDQSAxMB4XDTE1MDUyNjAwMDAwMFoXDTM4MDExNzAwMDAwMFowOTEL
MAkGA1UEBhMCVVMxDzANBgNVBAoTBkFtYXpvbjEZMBcGA1UEAxMQQW1hem9uIFJv
b3QgQ0EgMTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBALJ4gHHKeNXj
ca9HgFB0fW7Y14h29Jlo91ghYPl0hAEvrAIthtOgQ3pOsqTQNroBvo3bSMgHFzZM
9O6II8c+6zf1tRn4SWiw3te5djgdYZ6k/oI2peVKVuRF4fn9tBb6dNqcmzU5L/qw
IFAGbHrQgLKm+a/sRxmPUDgH3KKHOVj4utWp+UhnMJbulHheb4mjUcAwhmahRWa6
VOujw5H5SNz/0egwLX0tdHA114gk957EWW67c4cX8jJGKLhD+rcdqsq08p8kDi1L
93FcXmn/6pUCyziKrlA4b9v7LWIbxcceVOF34GfID5yHI9Y/QCB/IIDEgEw+OyQm
jgSubJrIqg0CAwEAAaNCMEAwDwYDVR0TAQH/BAUwAwEB/zAOBgNVHQ8BAf8EBAMC
AYYwHQYDVR0OBBYEFIQYzIU07LwMlJQuCFmcx7IQTgoIMA0GCSqGSIb3DQEBCwUA
A4IBAQCY8jdaQZChGsV2USggNiMOruYou6r4lK5IpDB/G/wkjUu0yKGX9rbxenDI
U5PMCCjjmCXPI6T53iHTfIUJrU6adTrCC2qJeHZERxhlbI1Bjjt/msv0tadQ1wUs
N+gDS63pYaACbvXy8MWy7Vu33PqUXHeeE6V/Uq2V8viTO96LXFvKWlJbYK8U90vv
o/ufQJVtMVT8QtPHRh8jrdkPSHCa2XV4cdFyQzR1bldZwgJcJmApzyMZFo6IQ6XU
5MsI+yMRQ+hDKXJioaldXgjUkK642M4UwtBV8ob2xJNDd2ZhwLnoQdeXeGADbkpy
rqXRfboQnoZsG4q5WTP468SQvvG5
-----END CERTIFICATE-----
*COMMAND PROCESSED OK
```

The device will answer with the message COMMAND PROCESSED OK if the configuration was successful.

Do the same for the other two certificates with their respective commands.

Your device can now connect to the Mosquitto Broker and send your encrypted data safely.

Disclaimer:

Mosquitto is an open source (EPL/EDL licensed) message broker that implements the MQTT protocol. All products and software mentioned in this document for educational and demonstration purposes.

Revision: 1.1

Infinite Informatics, Ltd
1, Valaoritou Street
GR-54626 Thessaloniki, Greece
Phone: +30-2310-553545
E: info@indinf.gr
W: www.infinite.com.gr