webdyn

Flexitron group

TITAN

Application Note 32

Reading Modbus Devices and Sending Readings to an MQTTS Broker (with SSL/TLS security)

Reading Modbus Devices and Sending Readings to an MQTTS Broker (with SSL/TLS security)

1. Scenario Details

TITAN-based devices have all the typical functionalities of 4G/3G/2G routers, as well as a series of added features that make them one of the most feature-packed routers on the market.

One of the added features is the ability to interrogate Modbus RTU and TCP devices autonomously, subsequently sending the data to a WEB, FTP or MQTT server.

As always, this capability will be illustrated using a simple example.

2. Description of the Example

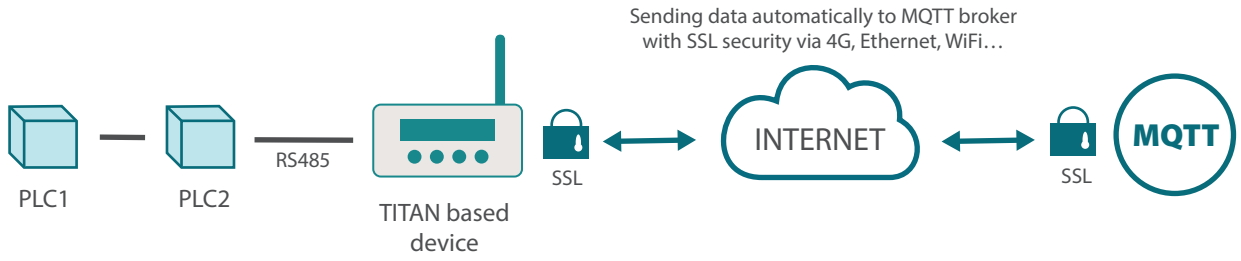
In this example, a TITAN-based device will be configured to collect, store and send the Modbus registers of two PLCs via MQTT. The reads will be done every 10 minutes.

The following Modbus registers must be read from PLC1:

1;10;11;12;55;56;69;70;72;73;74;75;76;77;78;79;80;100;101;102;103;104;105;106;107;108;109;120;121;122;123;124;130;131;132;133;152;153;154;160;161;162;163;164;165;166;170

The following registers must be read from PLC2:

10;11;12;13;14



This means we need to obtain a map of a number of registers, which are not always consecutive, from PLC1. PLC2 is easier as we only need 5 consecutive registers

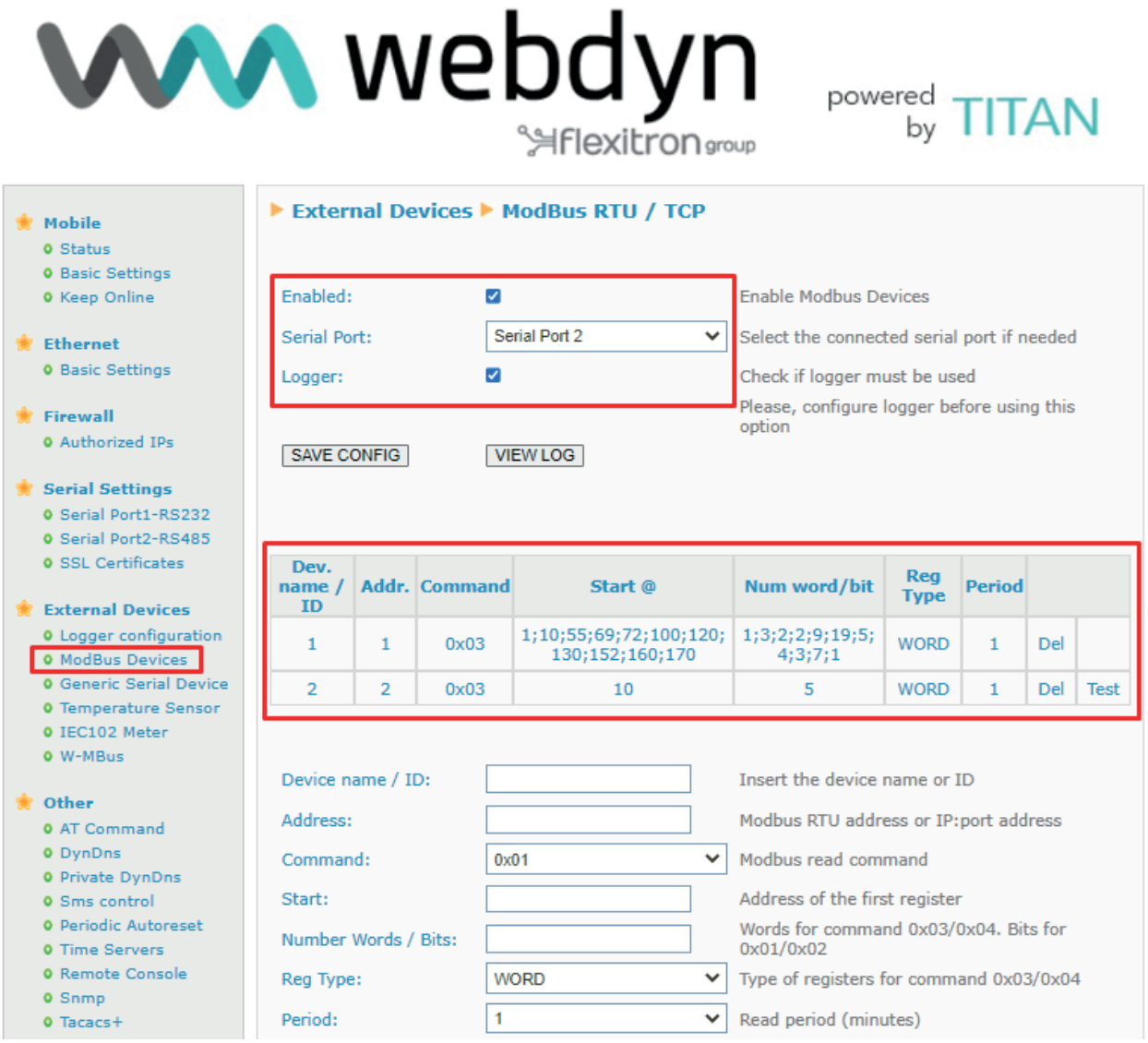
The PLCs are RS485 devices, so we will use Modbus RTU, but the scenario is also perfectly valid for Modbus TCP devices (with Ethernet) or a mix of both (Modbus TCP and Modbus RTU).

3. Configuring the Serial Port of the Titan-Based Device to which the Modbus Devices will be Connected

Let's imagine that the PLCs, which have RS485 ports, have the following serial port configuration: 9600,8,N,1. The first task is to configure the Serial Port2-RS485 section of the TITAN-based device. We will configure it as shown in the following figure:

4. Configuring the Titan to Read Modbus Devices

Click on the link: “External Devices > Modbus Devices” and configure the screen as shown below:



We want to read registers 10,11,12,13,14 from PLC 2. All we need to do is enter register 10 in the “Start” field and 5 in the “Number Words” field (as we want to read 5 registers, from 10 to 14). PLC1 is more complex, since we have a non-consecutive register map. The different blocks of registers we need to read will therefore be separated by “;” (semicolons). This means if we want to read registers:

1;10;11;12;55;56;69;70;72;73;74;75;76;77;78;79;80;100;101;102;103;104;105;106;107;108;109;120;121;122;123;124;130;131;132;133;152;153;154;160;161;162;163;164;165;166;170

We must enter the following in the Start field (the initial register of each block):

1;10;55;69;72;100;120;130;152;160;170

And, in the "Number Words" field (the number of registers to be read from each block):

1;3;2;2;9;10;5;4;3;7;1

5. Configuring the Logger (communication with the MQTT server)

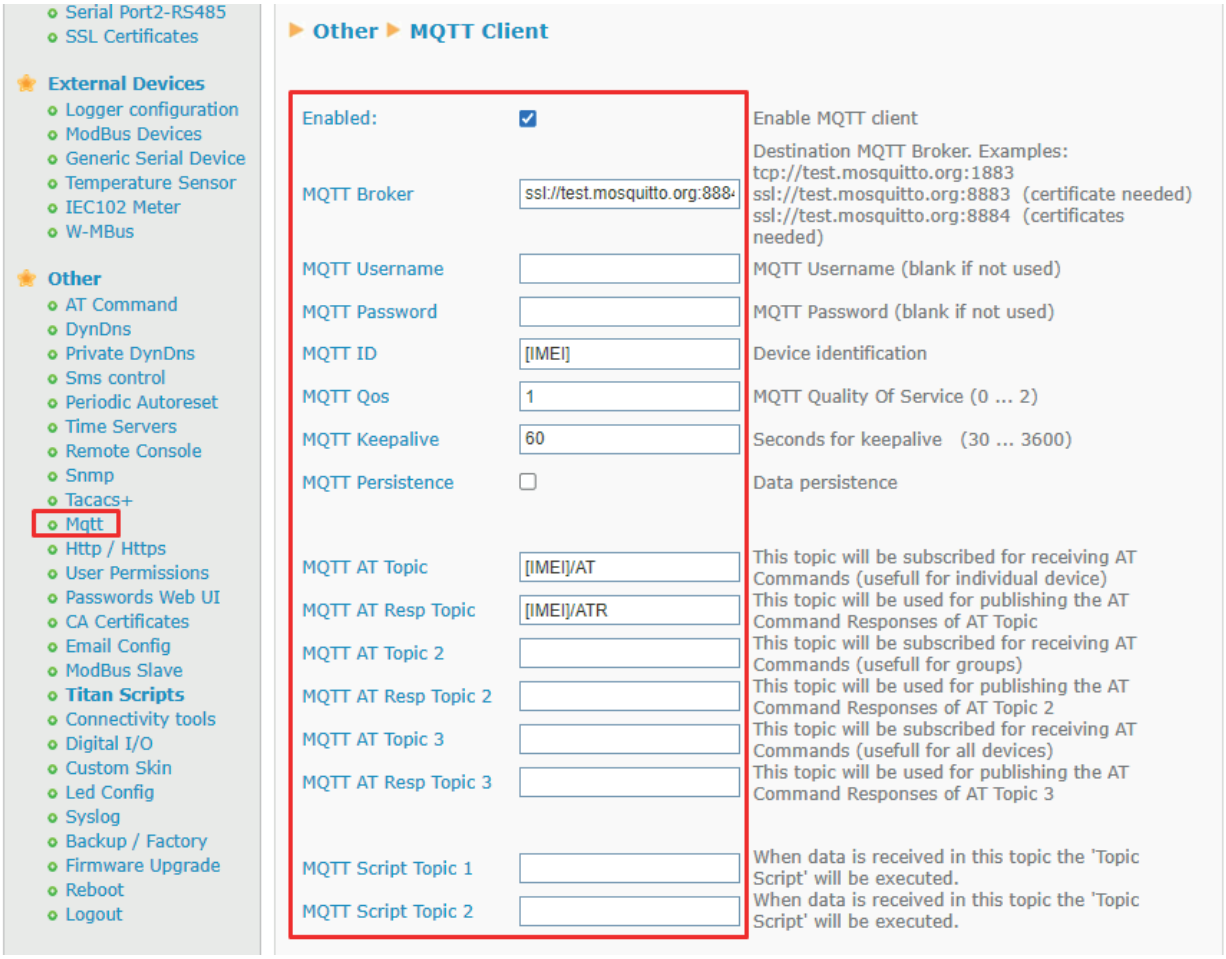
The next step is to configure the Logger. This is the data storage and transmission system used by the TITAN-based device itself. In this example we are going to configure the device to send the data to an MQTT broker.

As an MQTT broker we will use the Mosquitto test broker located at test.mosquito.org. The data from each reading will be sent in JSON format (timestamp, team ID, etc.). We will use SSL/TLS to secure it, which will require a Server certificate and a Client certificate.

As can be seen in the following figure, we will access it through the “External Devices > Logger Configuration” menu and configure the section as follows:

In this screen we have activated MQTT send mode. All data collected by the TITAN-based device will be published in the [IMEI]/logger topic. That the device will replace the [IMEI] tag with your real IMEI. This means that, if the device's IMEI is 357299070082380 (the IMEI can be found in the Mobile > Status menu), the topic used by the TITAN-based device to publish the collected Modbus data will be: 357299070082380/logger.

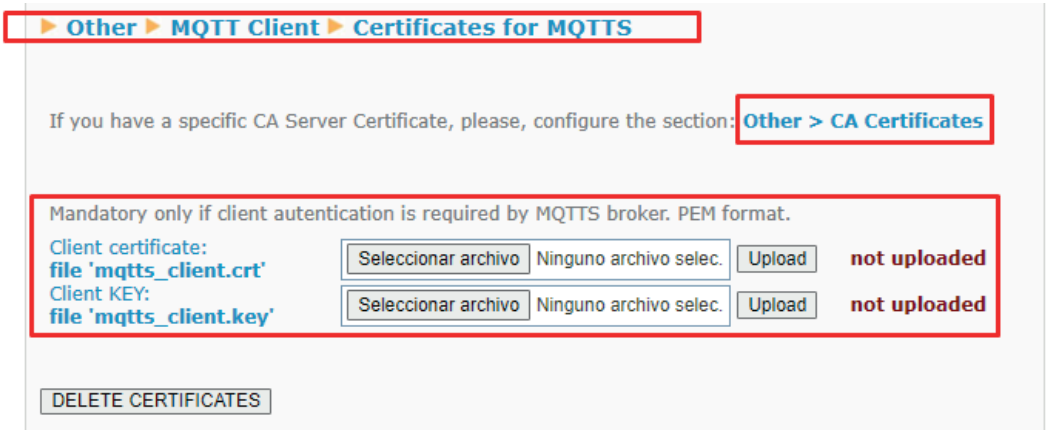
Conversely, if we activate logger mode, we must configure the MQTT client in the “Other > Mqtt” section.



We will check the “Enabled” box. Enter the following in the “MQTT Broker” field: ssl://test.mosquitto.org:8884, indicating the URL of the MQTTS broker we want to connect to. Enter the device’s unique in MQTT ID (if we connect 2 devices with the same ID they will be disconnected, as the broker cannot handle 2 devices with the same ID. If we want to be able to send AT commands to the TITAN-based device (e.g. to configure it or view its status from a mobile phone that is also connected to the MQTT broker), we will add the fields MQTT AT Topic and MQTT AT Response Topic.

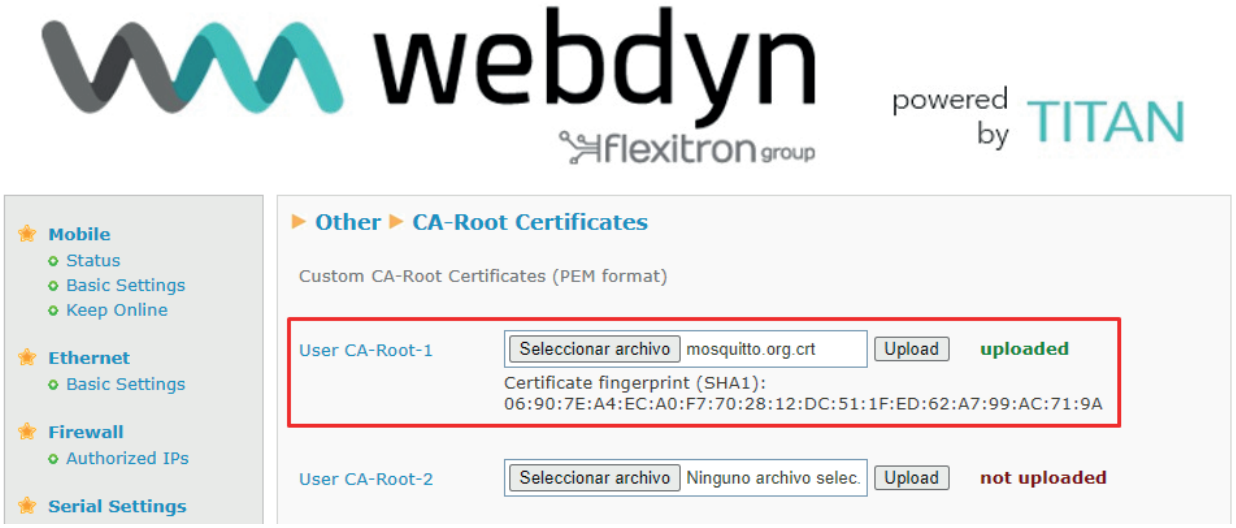
All AT commands sent to the broker under the MQTT AT Topic will be executed by the TITAN-based device. Once executed, the result of the AT command will be published by the TITAN-based device in the MQTT AT Response Topic.

Next we will configure the certificates section.



To do this you should follow the instructions provided at: http://test.mosquitto.org/

The first thing to do is download the CA certificate from mosquitto.org.crt (PEM format) which can be found on the previous page. This certificate must be included in the "Other > CA certificates" section, as the TITAN-based device will use it to authenticate the MQTTS broker when establishing a connection.



If you do not want to upload the certificate, another option would be to check the "Allow all certificates" box, however this is not recommended as it is less secure. It is only valid for tests.

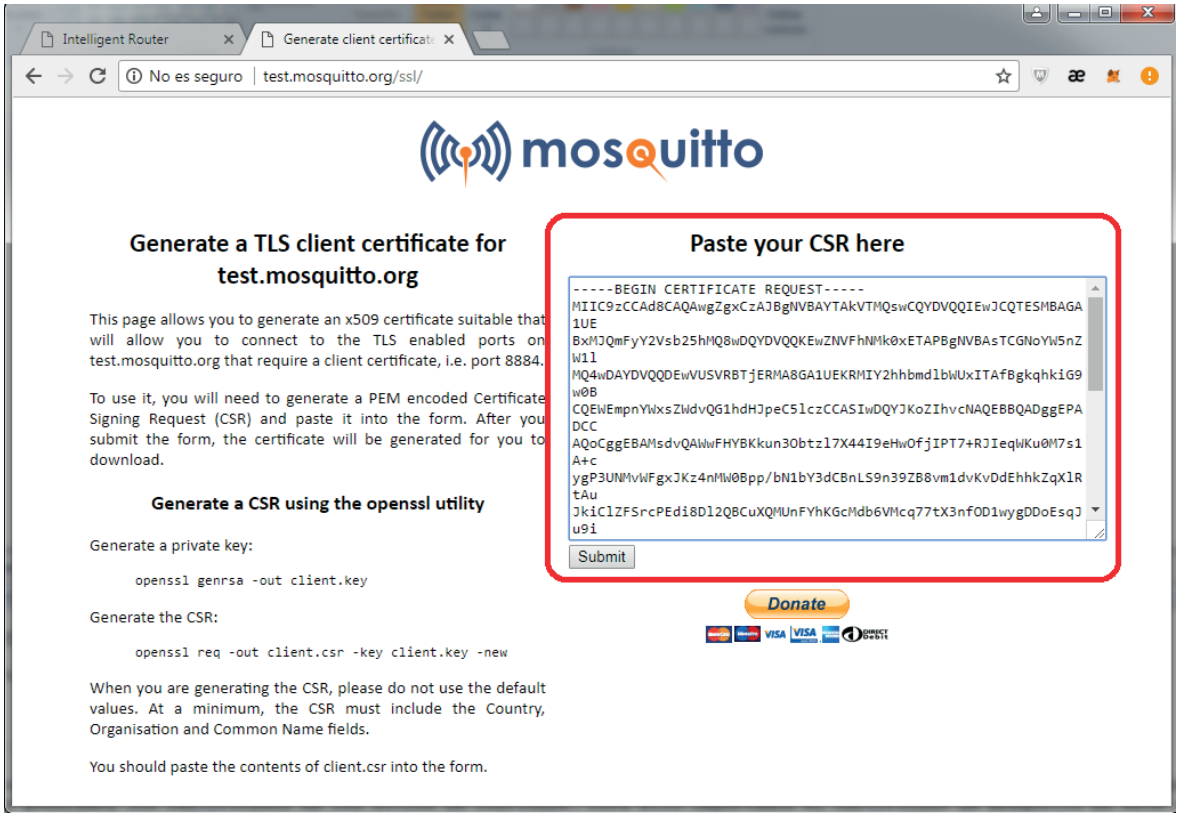
The we need to execute the statements found at this link: http://test.mosquitto.org/ssl/ for which we must have openssl installed on our PC. We will execute:

openssl genrsa -out client.key

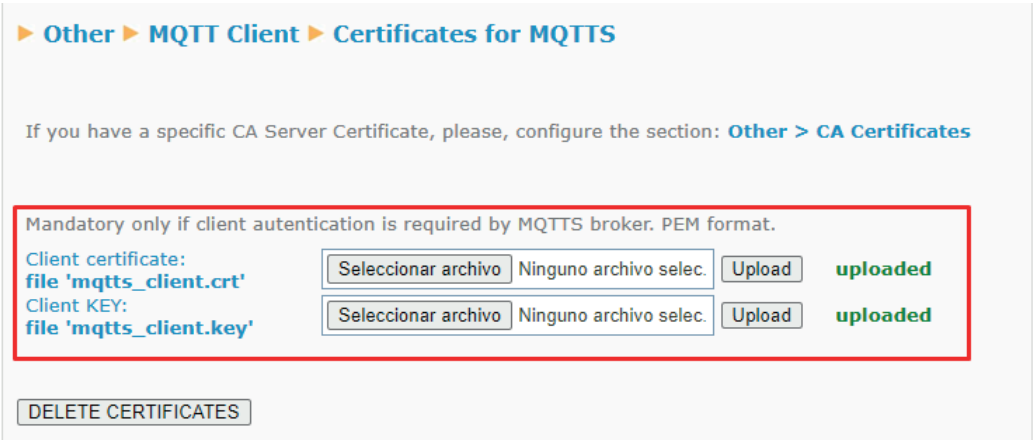
openssl req -out client.csr -key client.key –new

Which will give us the client.key and client.csr files.

Once we have both files, we copy the contents of the client.csr file to the text box and click on “Submit”. This will enable us to obtain the file with the required client.crt certificate.



Having generated the client.key and client.crt files, we can insert them at the bottom of the Other > Mqtt page in the TITAN-based device.



6. Other Considerations

- After configuring the TITAN-based device we will need to reset it so that the new configuration takes effect and it starts reading and sending.
- Each time the TITAN-based device sends a measurement to the MQTT broker, it does so using a JSON object of the following type:

Example JSON for PLC 2

{“IMEI”:”357299070082380”,”TYPE”:”MODB”,”TS”:”2022-07-15T10:14:02Z”,”ID”:”2”,”A”:”2”,”ST”:”10”,”N”:”5”,”V”:[10,11,12,0,0],”P”:”ID-1234”}

Example JSON for PLC 1

{“IMEI”:”357299070082380”,”TYPE”:”MODB”,”TS”:”2022-07-15T10:09:01Z”,”ID”:”1”,”A”:”1”,”STX”:[1,10,55,69,72,100,120,130,152,160,170],”NX”:[1,3,2,2,9,10,5,4,3,7,1], “PX”:[0,1,4,6,8,17,27,32,36,39,46], “V”:[1,10,11,12,55,56,69,70,72,73,74,75,76,77,78,79,80,100,101,102,103,104,105,106,107,108,109,120,121,122,123,124,130,131,132,133,152,153,154,160,161,162,163,164,165,166,170], “P”:”ID-1234”}

Where:

IMEI: is the unique identifier for the modem

TYPE: indicates the type of data (MODB = Modbus reading)

TS: is the Timestamp (the time the reading was read)

ID: name or identifier of the Modbus device

A: Modbus device address

ST: the address of the first Modbus register read

STX: array that indicates the address of the first Modbus registers when reading groups

N: indicates the number of words read

NX: an array that indicates the number of words read when reading groups of registers

PX: an array indicating the position of the initial register of each block within V

V: An array containing the data read

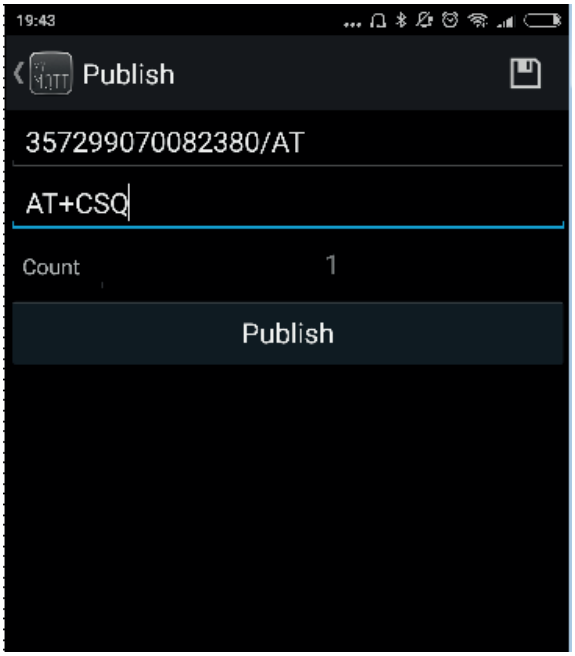
P: the ID field configured in the Logger

It should be noted that there are significant differences between the data sent by PLC1 and that sent by PLC2. PLC1 has groups, meaning that the ST and N fields are replaced by STX and NX in the JSON, these are the arrays in which the initial registers and the number of registers in each block are stored. Similarly, the PX register indicates the initial position of the group within the array V (PX is not really necessary as it can be calculated, but it is included to facilitate decoding of the operation on the server).

7. Communicating with TITAN-based Devices Using AT Commands Sent via MQTT

As mentioned above, you can send AT Commands to the TITAN-based device via MQTT. We can therefore change the configuration of the device, check states such as coverage, reset it, switch a relay, etc.

For example, if we want to check the coverage level, we can use the Android myMQTT app, sending the AT+CSQ command to the [IMEI]/AT topic, which in this example is 357299070082380/AT.



When the “Publish” button is clicked, the command will be received by the TITAN-based device and executed. The answer will be published by the TITAN-based device in the 357299070082380/ATR topic, as was configured beforehand.