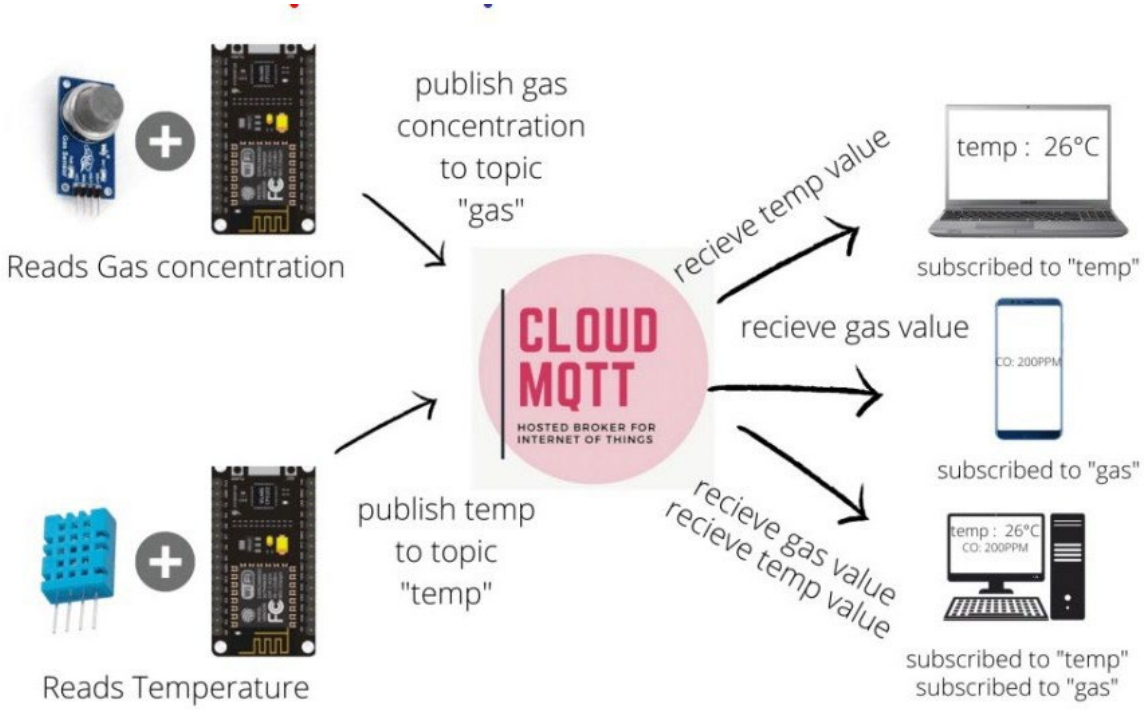
THỰC HÀNH CLOUD

1. Giới thiệu về giao thức MQTT

MQTT (Message Queuing Telemetry Transport) là giao thức truyền thông điệp (message) theo mô hình publish/subscribe (cung cấp / thuê bao), được sử dụng cho các thiết bị IoT với băng thông thấp, độ tin cậy cao và khả năng được sử dụng trong mạng lưới không ổn định. Nó dựa trên một Broker (tạm dịch là "Máy chủ môi giới") "nhẹ" (khá ít xử lý) và được thiết kế có tính mở (tức là không đặc trưng cho ứng dụng cụ thể nào), đơn giản và dễ cài đặt.



MQTT là lựa chọn lý tưởng trong các môi trường như:

- Những nơi mà giá mạng viễn thông đắt đỏ hoặc băng thông thấp hay thiếu tin cậy.
- Khi chạy trên thiết bị nhúng bị giới hạn về tài nguyên tốc độ và bộ nhớ.
- Bởi vì giao thức này sử dụng băng thông thấp trong môi trường có độ trễ cao nên nó là một giao thức lý tưởng cho các ứng dụng M2M (Machine to Machine).
- MQTT cũng là giao thức được sử dụng trong Facebook Messenger

2. Đặc điểm

- Dạng truyền thông điệp theo mô hình Pub/Sub cung cấp việc truyền tin phân tán một chiều, tách biệt với phần ứng dụng.

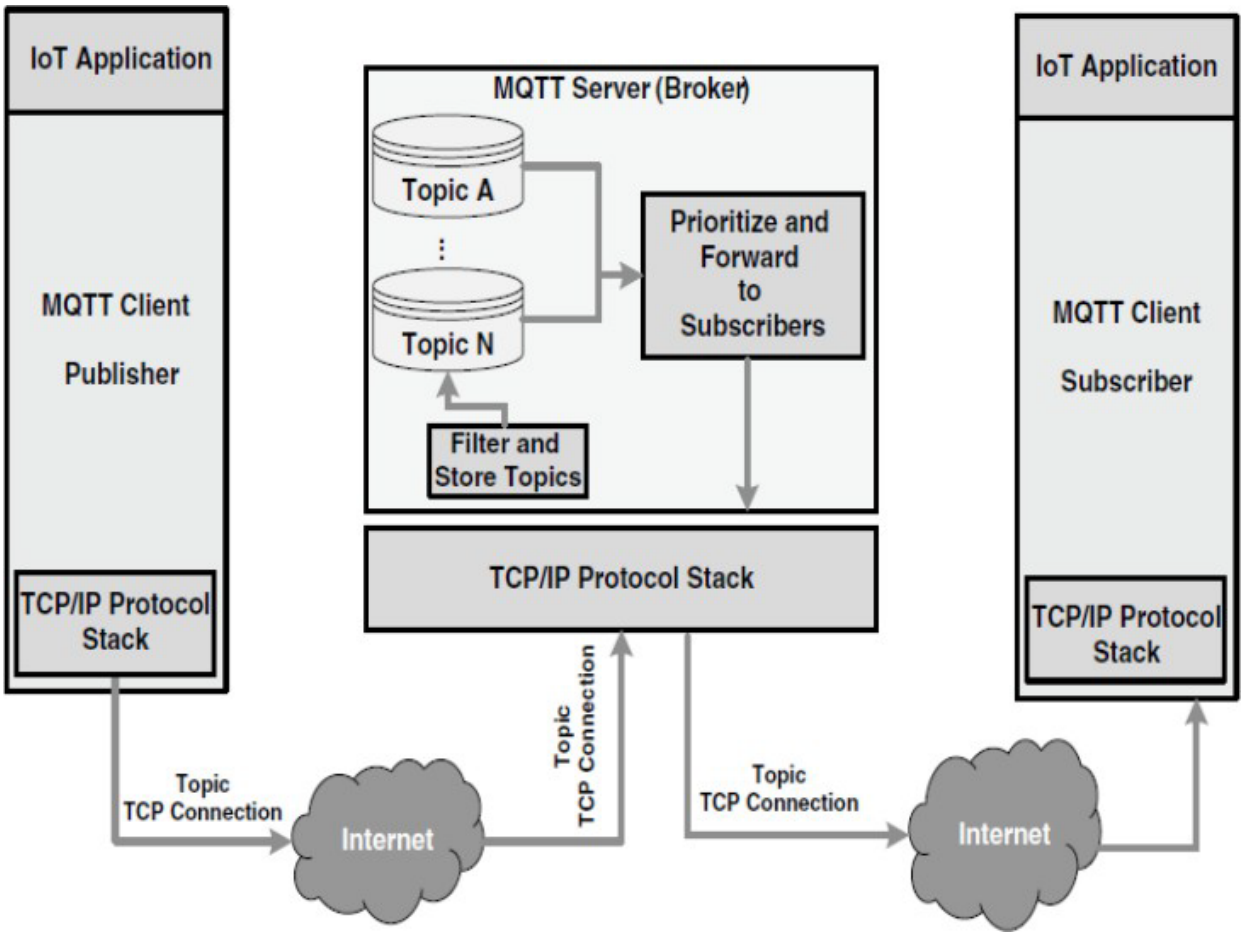
- Việc truyền thông điệp là ngay lập tức, không quan tâm đến nội dung được truyền.
- Được thiết kế như một giao thức nhắn tin đơn giản và gọn nhẹ .
- Sử dụng hệ thống xuất bản/đăng ký để trao đổi thông tin giữa máy khách và máy chủ .
- Không yêu cầu cả máy khách và máy chủ thiết lập kết nối cùng một lúc.
- Cung cấp khả năng truyền dữ liệu nhanh hơn .
- Là một giao thức nhắn tin thời gian thực.
- Cho phép client đăng ký lựa chọn các chủ đề để họ có thể nhận được thông tin mà họ đang tìm kiếm
- Là giao thức M2M tiêu thụ băng thông thấp .
- Giúp các thiết bị IoT trao đổi thông tin với nhau .
- Sử dụng giao thức lớp giao vận là TCP .
- Bảo mật nhờ SSL
- Sử dụng TCP/IP là giao thức nền.
- Tồn tại ba mức độ tin cậy cho việc truyền dữ liệu (QoS: Quality of service)

• QoS 0: Broker/client sẽ gửi dữ liệu đúng một lần, quá trình gửi được xác nhận bởi chỉ giao thức TCP/IP.

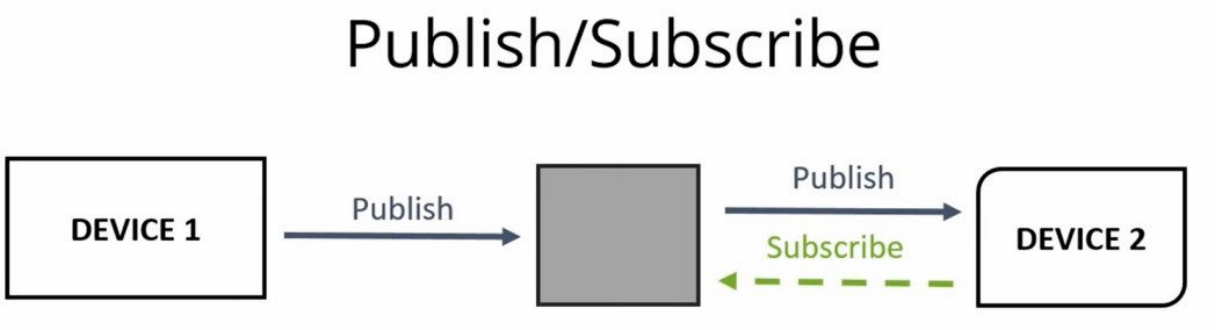
• QoS 1: Broker/client sẽ gửi dữ liệu với ít nhất một lần xác nhận từ đầu kia, nghĩa là có thể có nhiều hơn 1 lần xác nhận đã nhận được dữ liệu.

• QoS 2: Broker/client đảm bảo khi gửi dữ liệu thì phía nhận chỉ nhận được đúng một lần, quá trình này phải trải qua 4 bước bắt tay.

3. Kiến trúc của MQTT

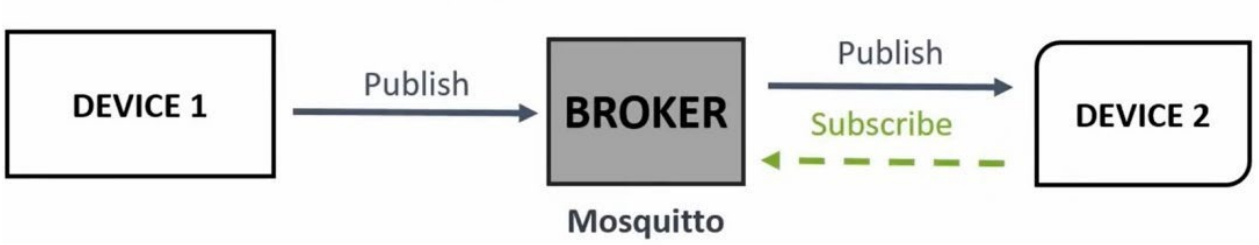


- Client
  - Publisher - Nơi gửi thông điệp
  - Subscriber - Nơi nhận thông điệp



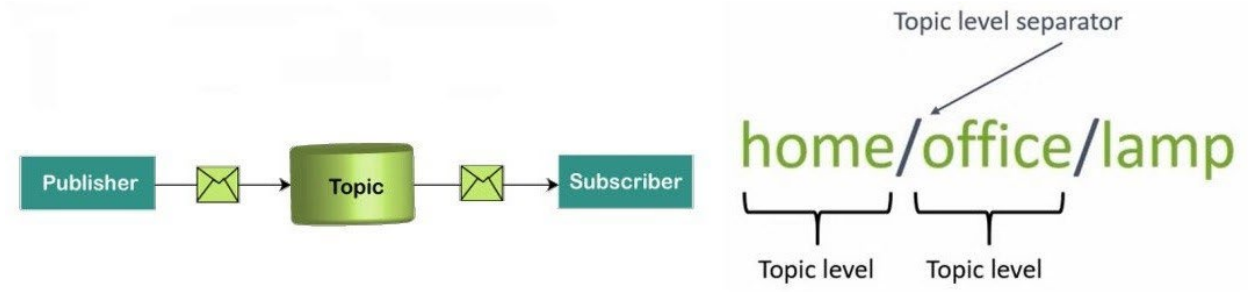
Client thì được chia thành hai nhóm là Publisher và Subscriber. Client chỉ làm ít nhất một trong 2 việc là publish các thông điệp (message) lên một/nhiều topic cụ thể hoặc subscribe một/nhiều topic nào đó để nhận message từ topic này.

- Broker - Máy chủ môi giới



Trong đó Broker được coi như trung tâm, nó là điểm giao của tất cả các kết nối đến từ Client (Publisher/Subscriber). Nhiệm vụ chính của Broker là nhận thông điệp (message) từ Publisher, xếp vào hàng đợi rồi chuyển đến một địa điểm cụ thể. Nhiệm vụ phụ của Broker là nó có thể đảm nhận thêm một vài tính năng liên quan tới quá trình truyền thông như: bảo mật message, lưu trữ message, logs, ...

- Topic



Có thể coi Topic là một hàng đợi các thông điệp, và có sẵn khuôn mẫu dành cho Subscriber hoặc Publisher. Một cách logic thì các topic cho phép Client trao đổi thông tin với những ngữ nghĩa đã được định nghĩa sẵn.

4. Thực hành cài đặt phần mềm MQTT

4.1. Cài đặt phần mềm MQTT Mosquitto broker trên máy chủ của IoT LAB (đã cài sẵn trên máy chủ , các bạn sinh viên đọc để tham khảo)

Có rất nhiều loại MQTT broker khác nhau, được viết bằng nhiều loại ngôn ngữ lập trình khác nhau. Trong bài viết này, mình sẽ giới thiệu đến các bạn một MQTT broker khá phổ biến là Mosquitto, cũng như cách cài đặt của nó trên Window.

Mosquitto là một MQTT Broker mã nguồn mở cho phép thiết bị truyền nhận dữ liệu theo giao thức MQTT version 5.0, 3.1.1 và 3.1 – Một giao thức nhanh, nhẹ theo mô hình publish/subscribe được sử dụng rất nhiều trong lĩnh vực Internet of Things. Mosquitto cung cấp một thư viện viết bằng ngôn ngữ C để triển khai các MQTT Client và có thể dễ dàng sử dụng bằng dòng lệnh: “mosquitto_pub” và “mosquitto_sub”. Ngoài ra, Mosquitto cũng là một phần của Eclipse Foundation, là dự án iot.eclipse.org và được tài trợ bởi cedalo.com

- Ưu điểm:

• Ưu điểm nổi bật của Mosquitto là tốc độ truyền nhận và xử lí dữ liệu nhanh, độ ổn định cao, được sử dụng rộng rãi và phù hợp với những ứng dụng embedded.

• Mosquitto rất nhẹ và phù hợp để sử dụng trên tất cả các thiết bị.

• Ngoài ra, Mosquitto cũng được hỗ trợ các giao thức TLS/SSL (các giao thức nhằm xác thực server và client, mã hóa các message để bảo mật dữ liệu).

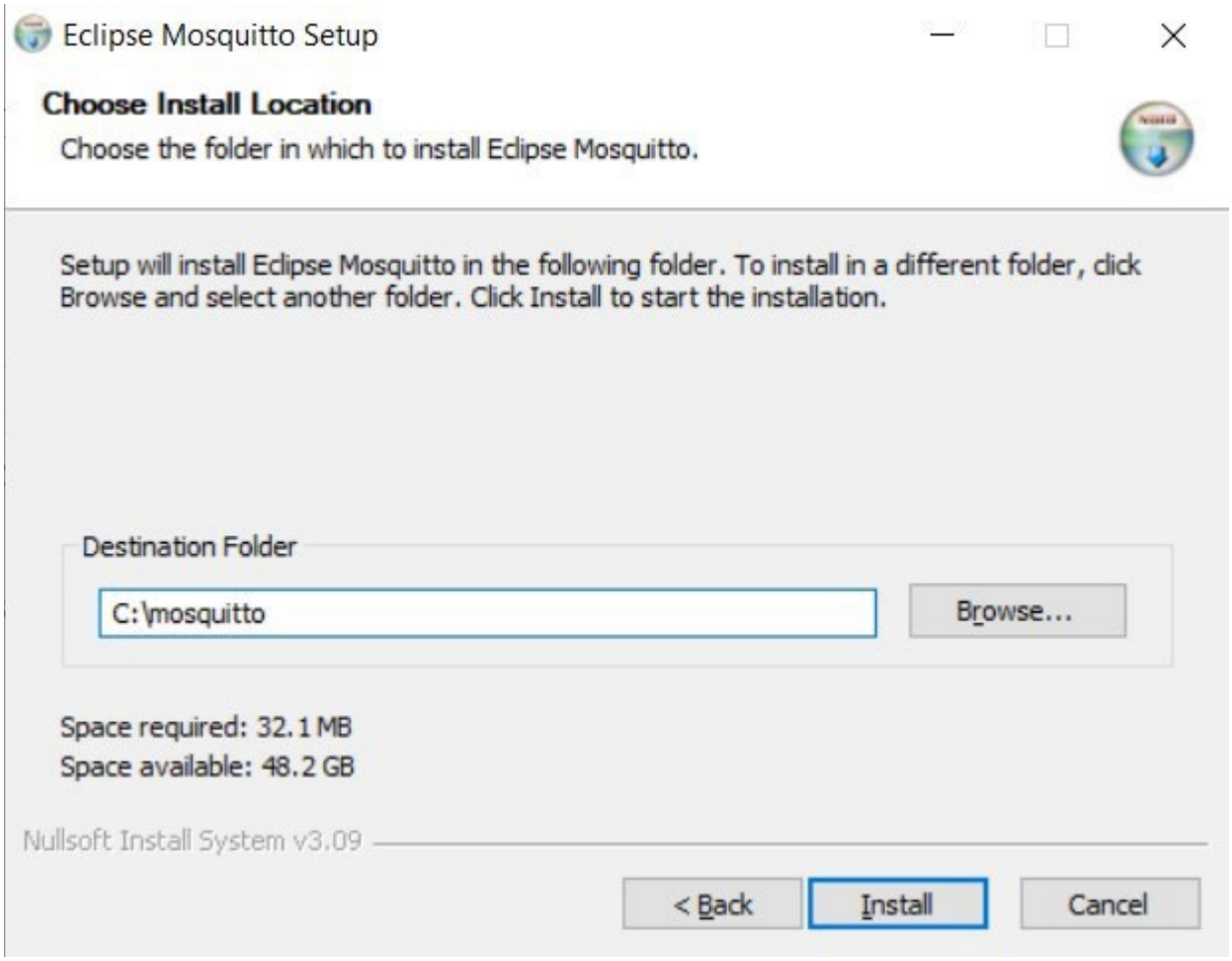
- Nhược điểm:

• Một số nhược điểm của mosquitto là khó thiết kế khi làm những ứng dụng lớn và ít phương thức xác thực thiết bị nên khả năng bảo mật vẫn chưa tối ưu.

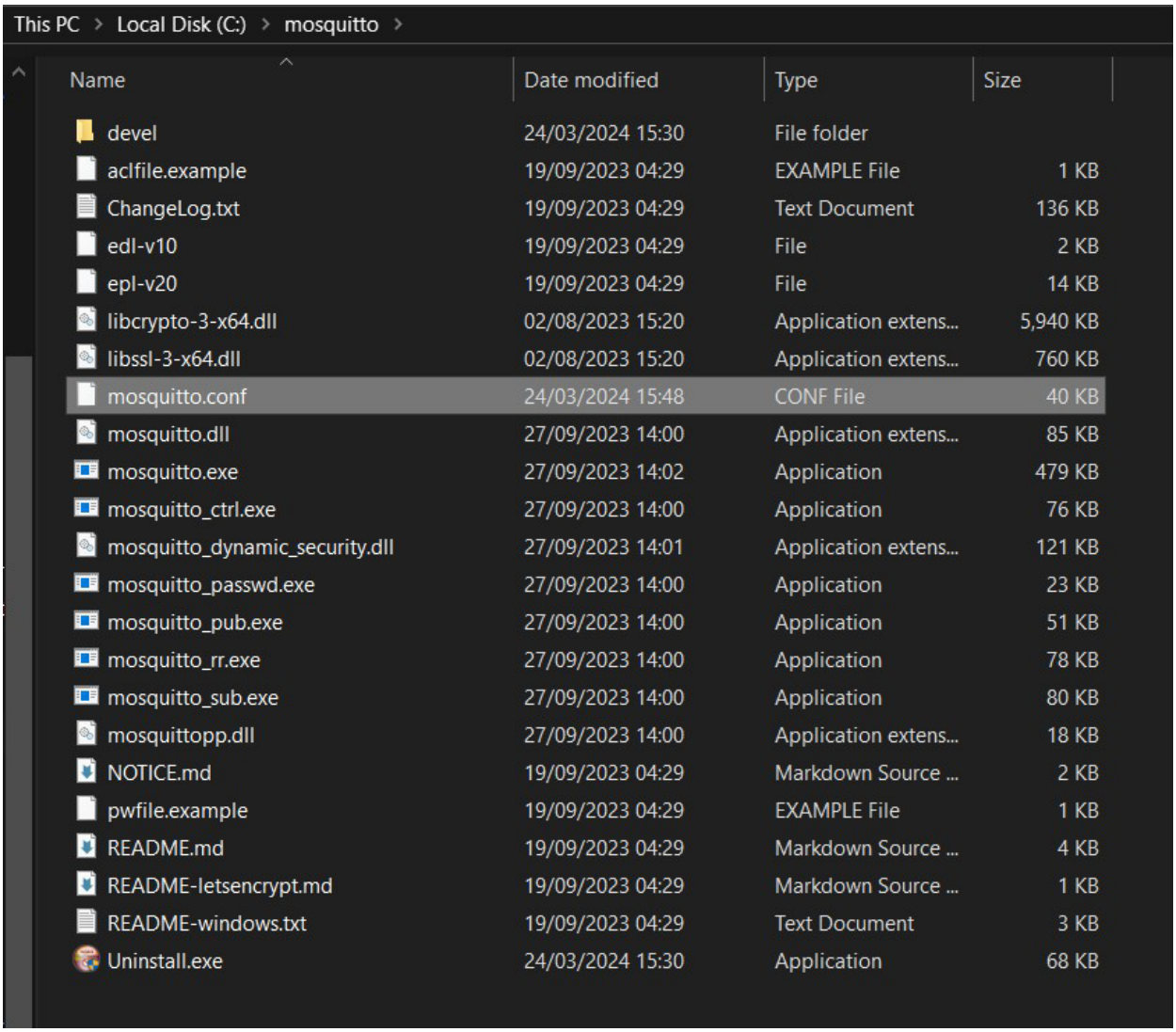
4.2. Cài đặt Mosquitto B roker

Trước tiên, chúng ta cần download mosquitto về máy, các bạn lựa chọn phiên bản phù hợp với máy tính của mình window/lunux/ubuntu. Sau đây là hướng dẫn cài Mosquitto broker trên Window:

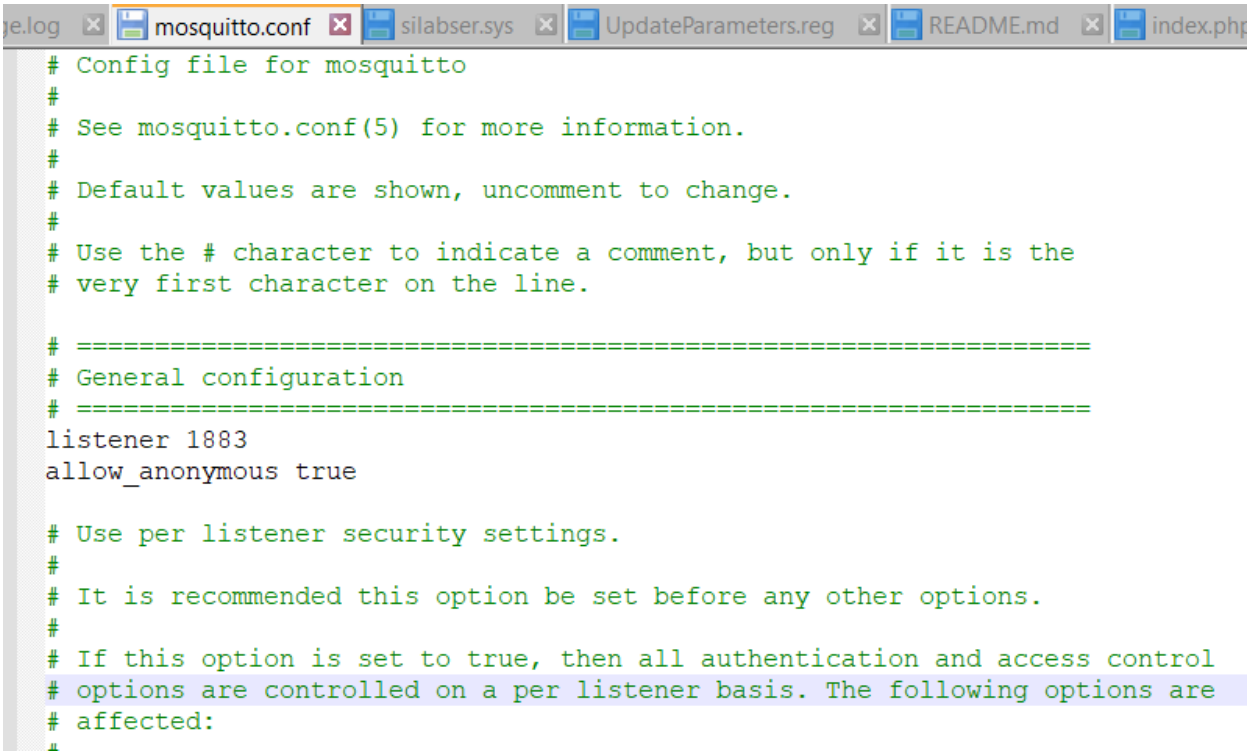
Sau khi download mosquitto về máy, thực hiện install như bình thường (nên để destination folder như dưới cho dễ thao tác sau này).



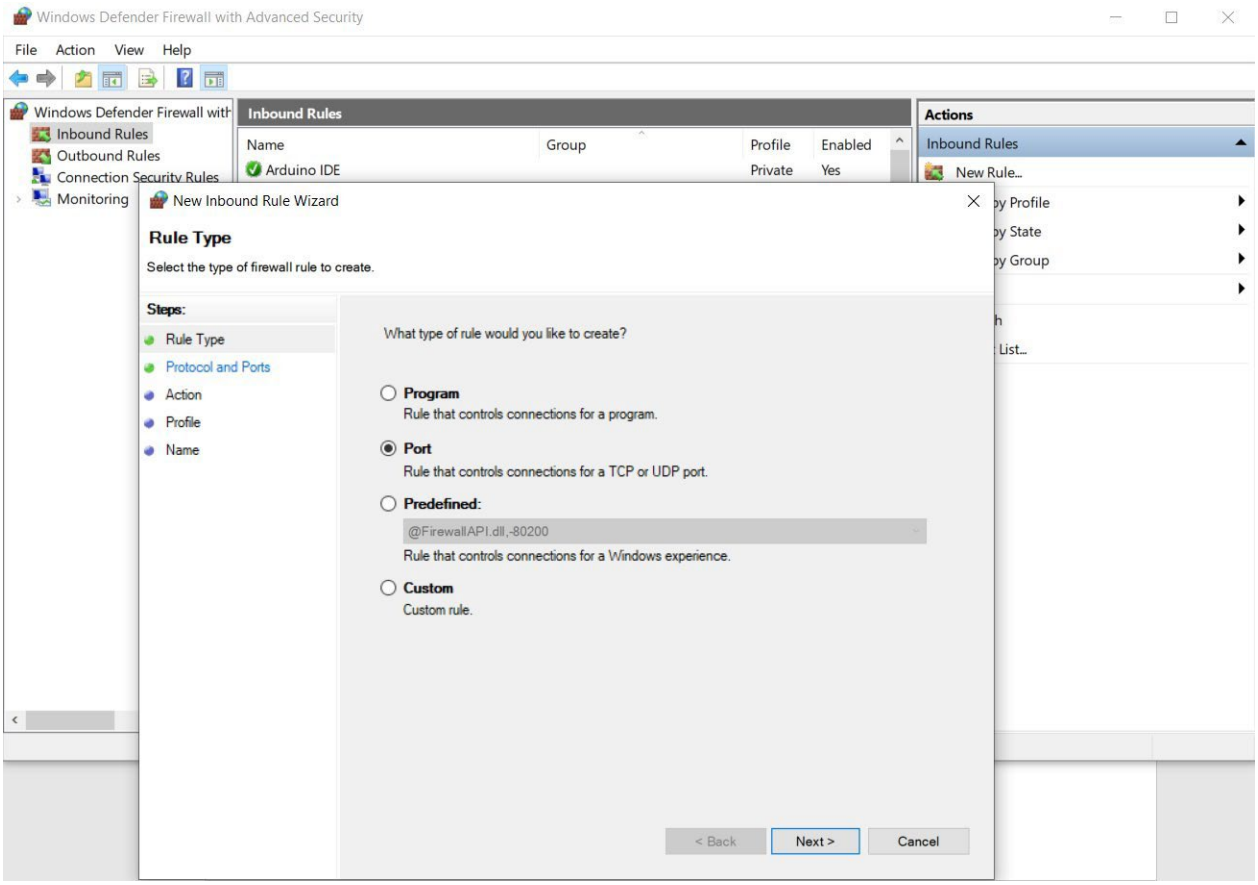
Tiếp theo vào thư mục mosquitto và chúng ta vừa cài đặt và mở file **mosquitto. conf**



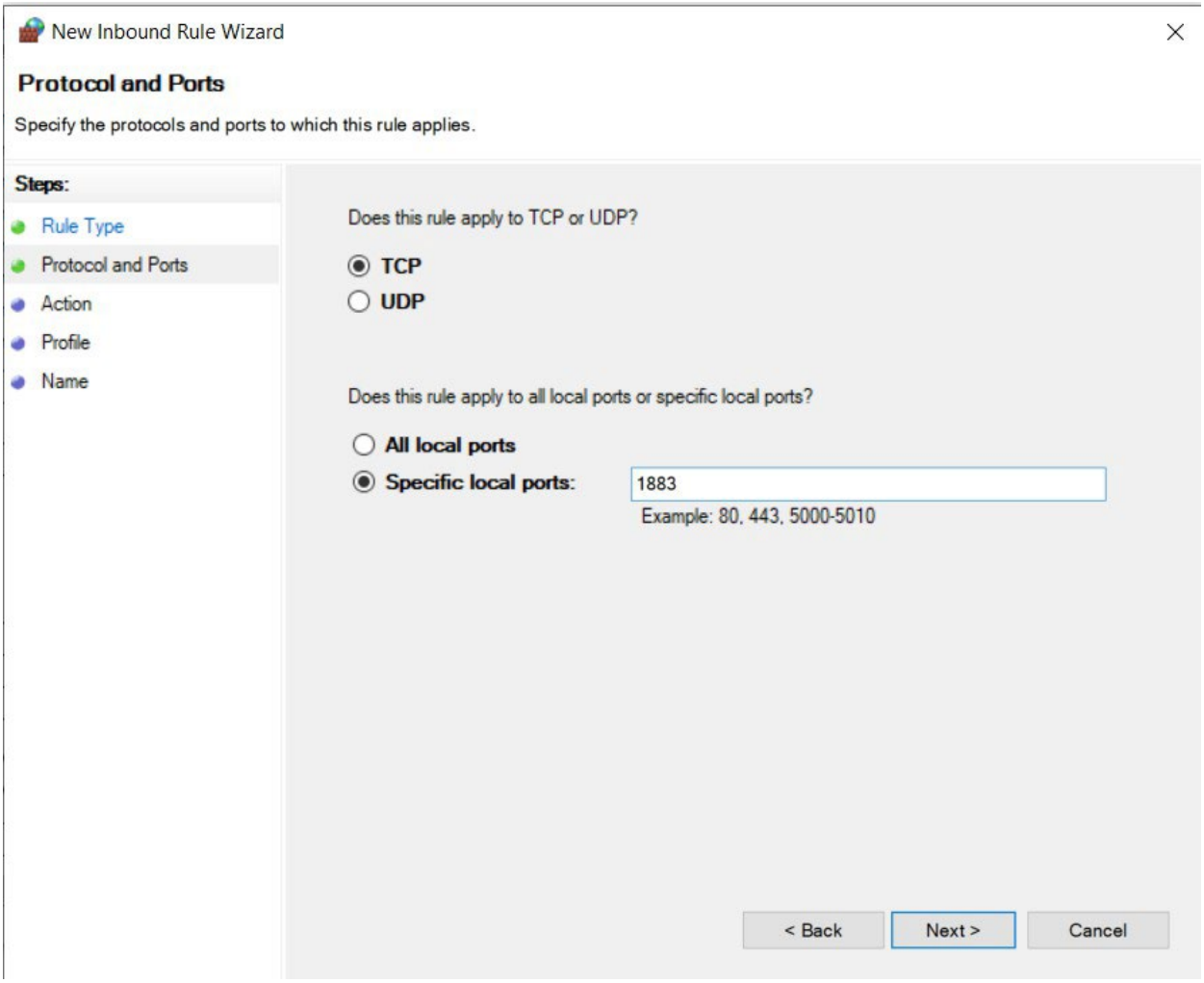
Thực hiện cấu hình như dưới hình, nhớ là lưu lại nhé.

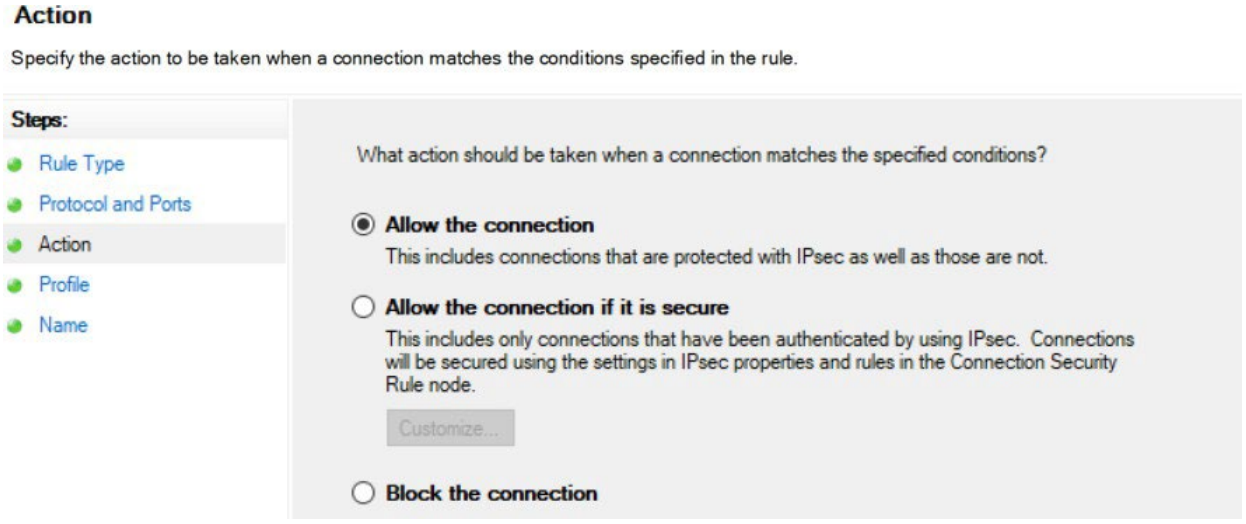


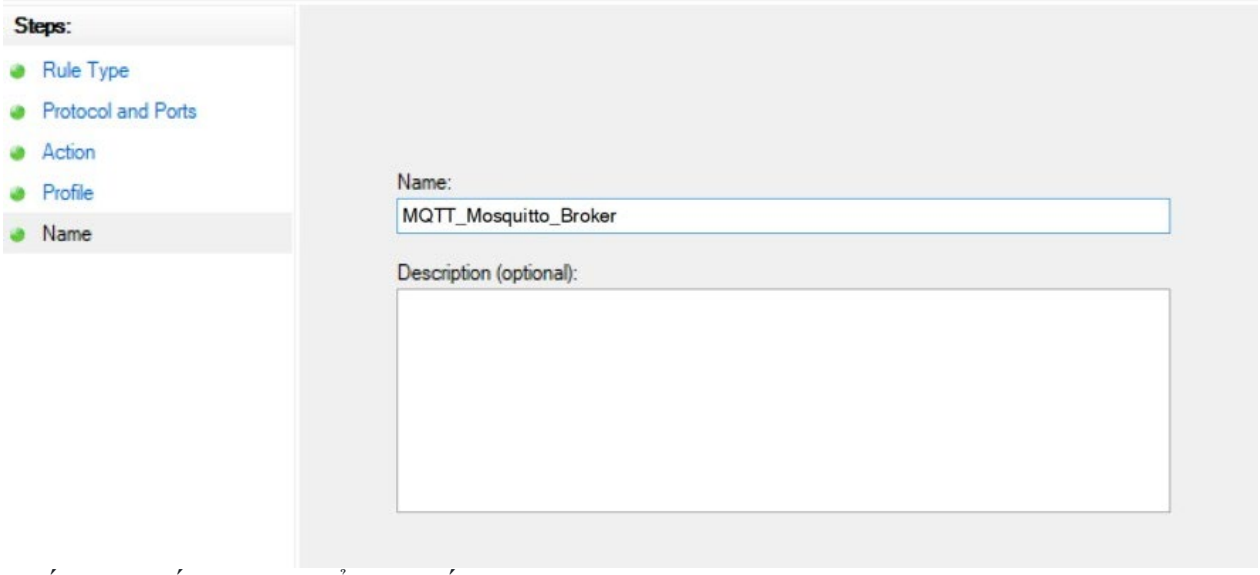
Tiếp đến, thực hiện cấu hình port cho Mosquitto broker: mở Windows Defender Firewall with Advanced Security, bấm vào Inbound Rules, chọn New Rule..., chọn Port và sau đó bấm Next.



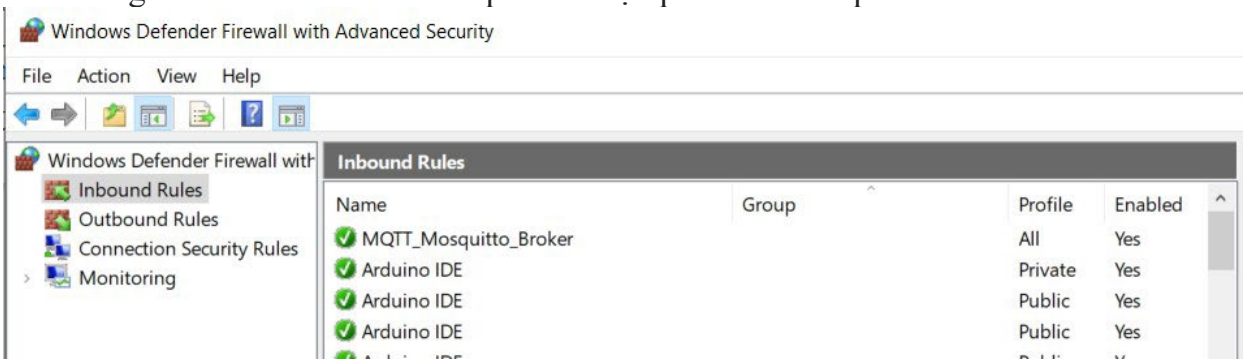
Tiếp theo chọn TCP và Specific local ports: 1883


New Inbound Rule Wizard
Protocol and Ports
Specify the protocols and ports to which this rule applies.
Steps:
Rule Type
Protocol and Ports
Action
Profile
Name
Does this rule apply to TCP or UDP?
TCP
UDP
Does this rule apply to all local ports or specific local ports?
All local ports
Specific local ports:
1883
Example: 80, 443, 5000-5010
< Back
Next >
Cancel


Action
Specify the action to be taken when a connection matches the conditions specified in the rule.
Steps:
Rule Type
Protocol and Ports
Action
Profile
Name
What action should be taken when a connection matches the specified conditions?
Allow the connection
This includes connections that are protected with IPsec as well as those are not.
Allow the connection if it is secure
This includes only connections that have been authenticated by using IPsec. Connections will be secured using the settings in IPsec properties and rules in the Connection Security Rule node.
Customize...
Block the connection

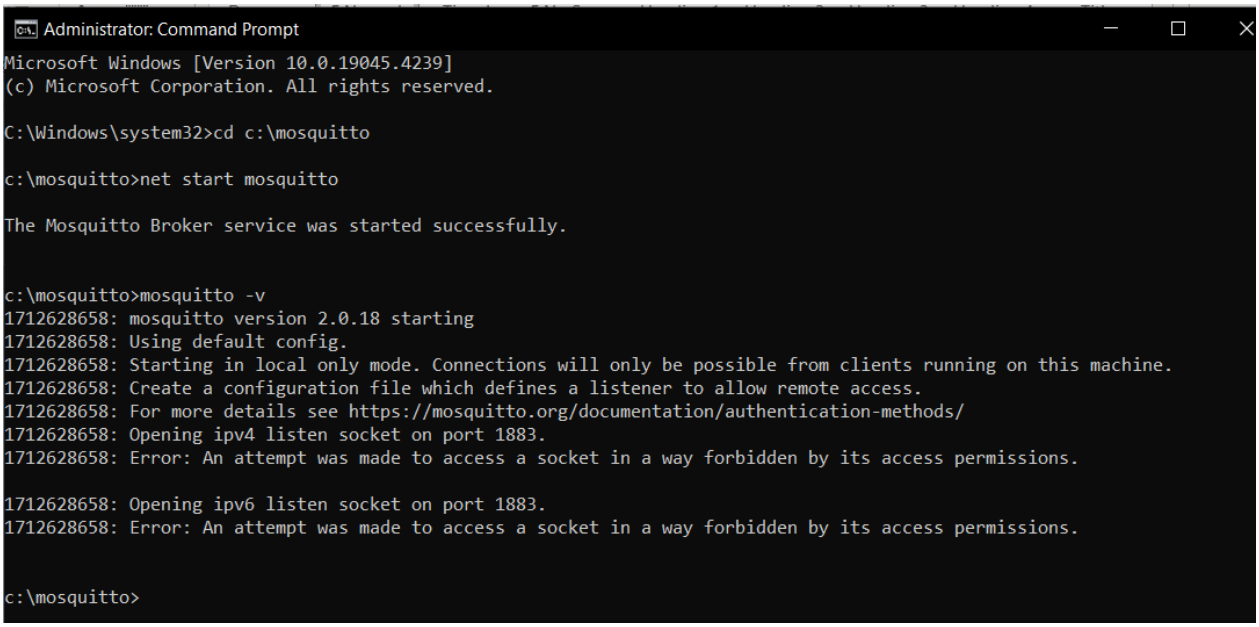


Cuối cùng bấm Finish để hoàn tất quá trình tạo port cho Mosquitto broker:



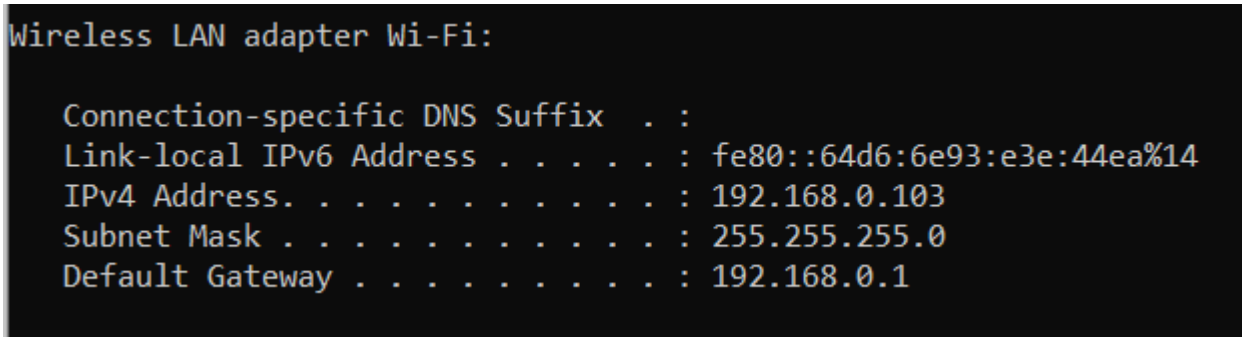
4.3. Thực hiện khởi chạy Mosquitto B roker

Mở Command Prompt với quyền Admin, truy cập vào folder mosquitto và ta đã cài đặt lúc đầu, thực hiện lệnh net start mosquiito để khởi chạy Mosquitto broker:

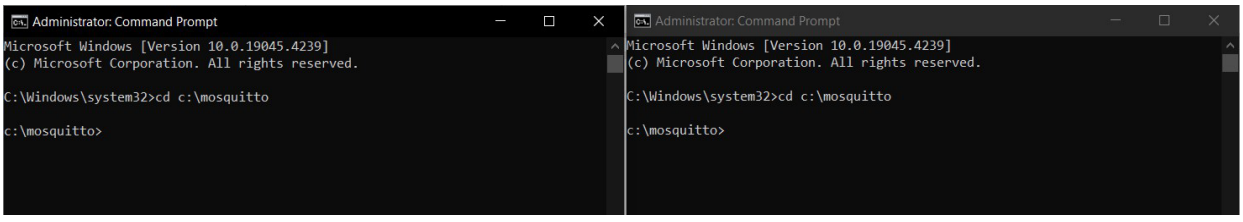


4.4. Thực hiện truyền và nhận message giữa Publisher và Subscriber

Trước tiên, kiểm tra địa chỉ IP Network máy, đây chính là địa chỉ IP Broker mà ta thiết lập trước đó, để các thiết bị trong cùng 1 mạng có thể Pub và Sub nội dung qua Broker này: 192.168.0.103

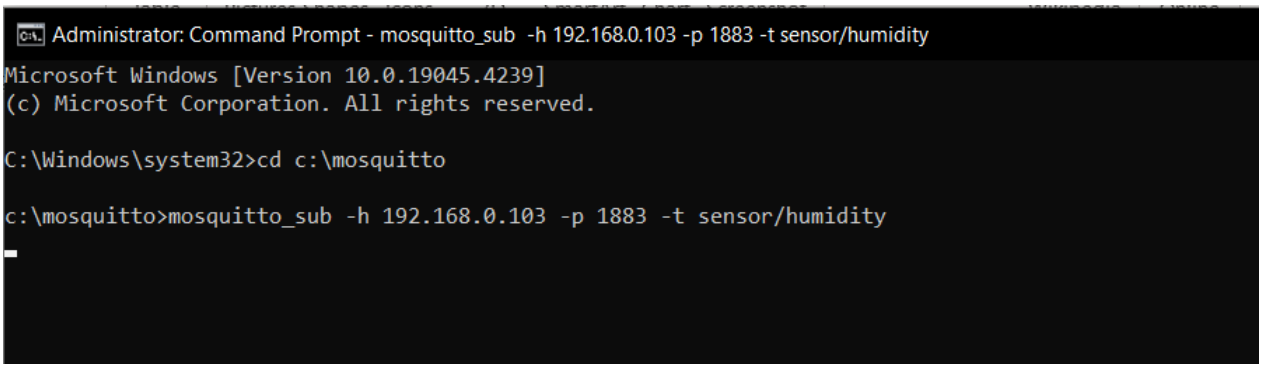


Tiếp theo, mở 2 tab Command Prompt chạy với quyền Admin và truy cập vào folder mosquitto:

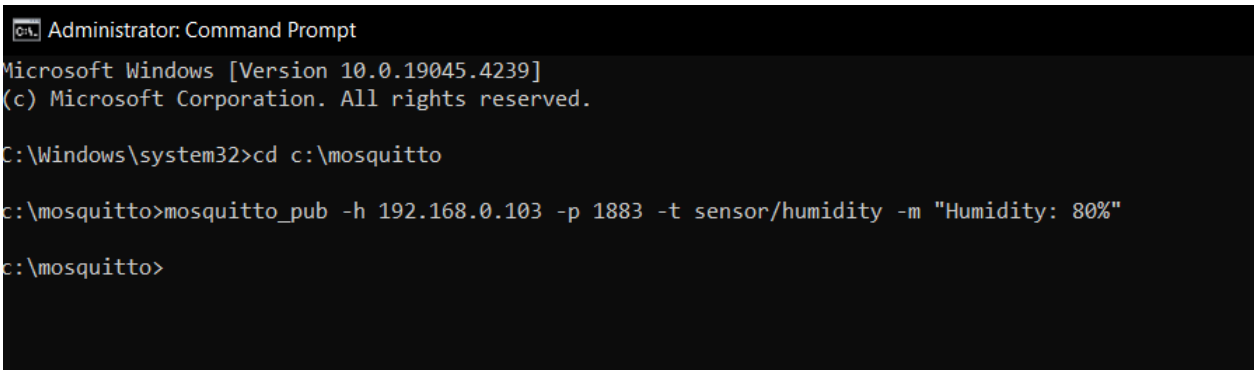


Chọn 1 Command Prompt làm Publisher và cái còn lại làm Subscriber.

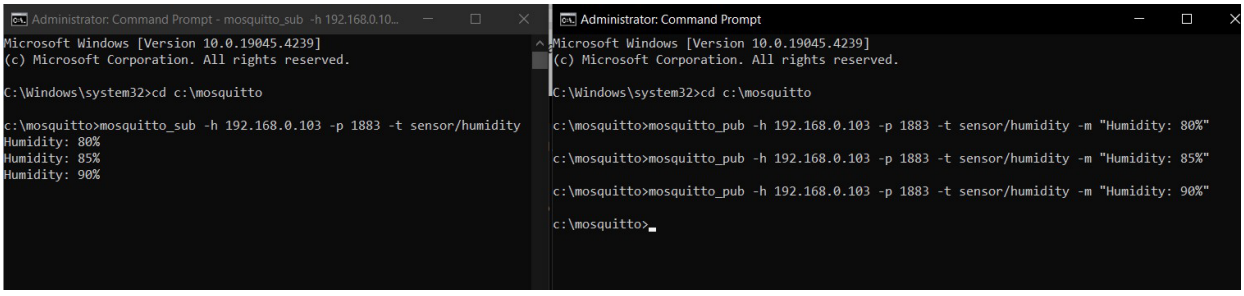
Với Command Prompt làm Subscriber ta thực hiện đăng ký các chủ đề cụ thể để nhận thông báo từ broker khi Publisher xuất bản các chủ đề đó:



Với Command Prompt còn lại làm Publisher ta thực hiện xuất bản các thông điệp về một chủ đề nào đó. Trong ví dụ này ta thự hiện xuất bản thông điệp về chủ đề mà Subscriber đã đăng ký ở bên trên:



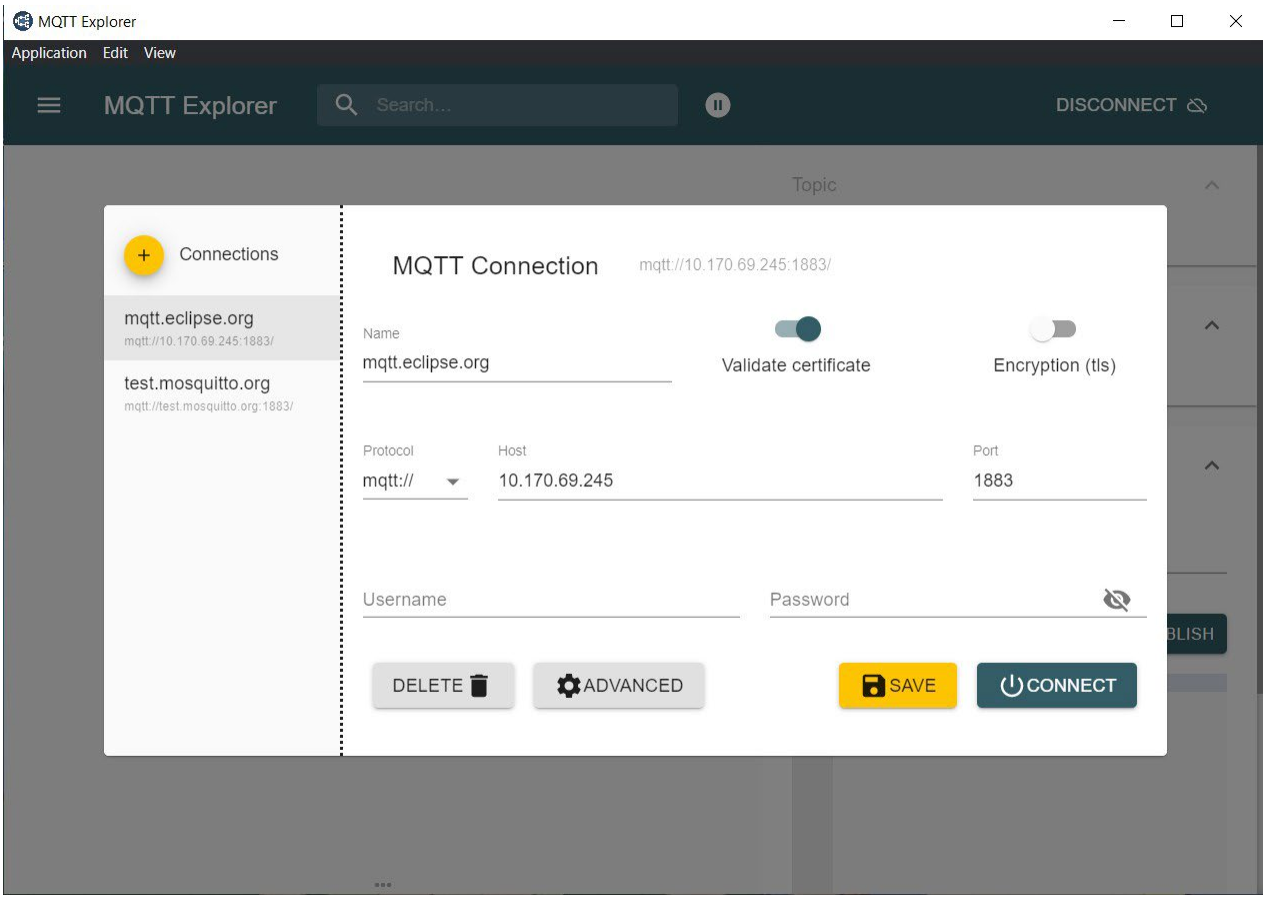
Như thế là đã hoàn thành quá trình truyền và nhận bản tin giữa Publisher và Subscriber:



4.5. Cài đặt phần mềm MQTT Explorer trên máy khách của sinh viên (các bạn sinh viên thực hành)

Để trực quan hóa dữ liệu, thay vì sử dụng Command Prompt để hiển thị bản tin được Publish gửi lên MQTT Broker như trên, chúng ta download phần mềm MQTT Explorer , chọn hệ điều hành và phiên bản phù hợp với máy tính của bạn.

Mở ứng dụng, chọn host của MQTT Broker là 10.170.69.245 và port là 1883, bấm Save và Connect để sử dụng.



5. Thực hành gửi bản tin từ KIT(PUBLISHER) lên MÁY CHỦ(BROKER) sử dụng giao thức MQTT

Sinh viên thực hành gửi bản tin từ KIT (bản tin là dữ liệu DHT / MQ-2 thu thập từ KIT) lên Broker Server

Chuẩn bị phần cứng

- Cảm biến DHT11/DHT22, cảm biến khí gas MQ-2
- Board Arduino Uno
- Chip WiFi D1 mini Lite
- Dây cáp nạp code cho arduino và chip WiFi
- Bus

Sơ đồ đấu nối

- Sơ đồ đấu nối DHT / MQ-2 : sinh viên xem lại tài liệu buổi 2

Arduino	MQ2
Vcc	5V
GND	GND
A1	Data

Arduino	DHT
Vcc	5V
GND	GND
D8	Data

- Sơ đồ đấu nối Arduino- Chip WiFi D1 mini Lite

Arduino	D1 Mini
Vcc	5V
GND	GND
D10	D4
D11	D3

Code tham khảo

Code nạp cho Arduino Uno

Có nhiệm vụ là thu thập dữ liệu từ cảm biến DHT22 và cảm biến khí gas MQ-2 và gửi dữ liệu cho Chip WiFi D1 mini Lite thông qua giao thức kết nối UART:

```
#include <DHT.h>
#include <ArduinoJson.h>
#include <SoftwareSerial.h>

#define DHT_Data 8
DHT dht (DHT_Data, DHT22);  // DHT11

#define MQ_Data A1

#define RX_PIN 10  // Chân RX của cổng nối tiếp mềm
#define TX_PIN 11  // Chân TX của cổng nối tiếp mềm

SoftwareSerial mySerial (RX_PIN, TX_PIN);  // Tạo một đối tượng cổng nối tiếp mềm

void setup () {
  Serial . begin ( 9600 );
  mySerial . begin ( 9600 );
  while (!Serial) delay ( 1 );

  dht . begin ();
}

unsigned long timeUpdata = millis ();
```

```
DynamicJsonDocument doc ( 1024 );

void loop () {
  if ( millis () - timeUpdata > 2000 ) {
    float hum = dht . readHumidity ();
    float temp = dht . readTemperature ();
    float gas = analogRead (MQ_Data);

    doc [ "temperature" ] = temp;
    doc [ "humidity" ] = hum;
    doc [ "gas" ] = gas;

    // gửi data thông qua cổng giao tiếp mềm cho Chip WiFi
    serializeJson (doc, mySerial);
    mySerial . println ();

    // In ra màn hình Serial
    serializeJson (doc, Serial);
    Serial . println ();

    timeUpdata = millis ();
  }
}
```

Code nạp cho Chip WiFi D1 mini Lite

Có nhiệm vụ nhận data từ Arduino Uno thông qua chuẩn giao thức UART để gửi data lên MQTT Broker (lúc này đóng vai trò là 1 Publisher) : **Sinh viên nhớ sửa topic khi publish và khi subscribe data từ Broker Server.**

Nếu sử dụng ESP32 thì sửa cổng nối Tx-Rx

Chú ý cài Lib cho board ESP8266/32

```
#ifdef ESP8266
#include <ESP8266WiFi.h> /* WiFi library for ESP8266 */
#else
#include <WiFi.h> /* WiFi library for ESP32 */
#endif
#include <Wire.h>
#include <SoftwareSerial.h>
#include <ArduinoJson.h>
#include <PubSubClient.h>
#include <WiFiClientSecure.h>

// Khai báo các chân RX và TX cho espsoftwareserial
```

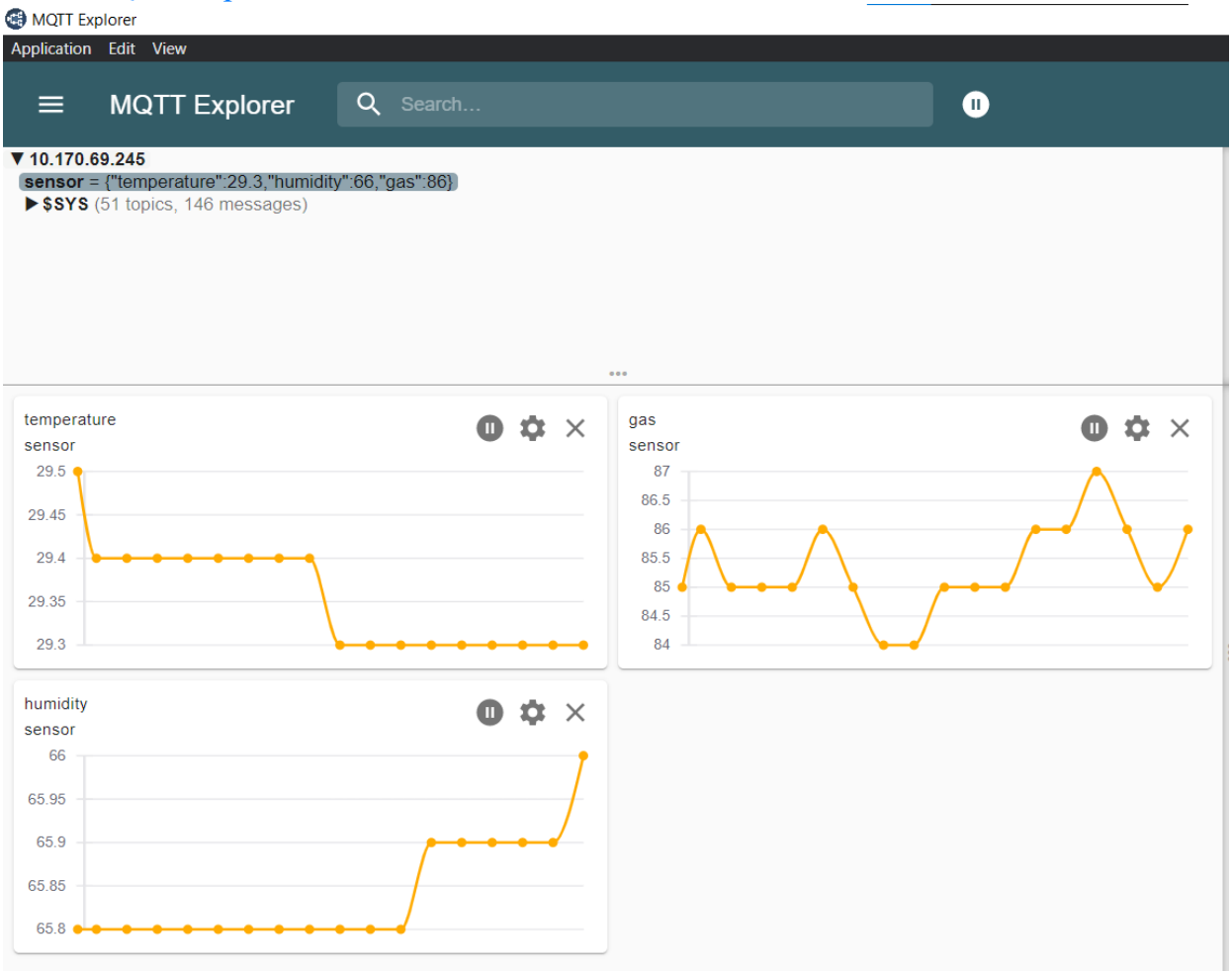
```
// Nếu bạn dùng Chip WiFi D1 mini Lite
#define RX_PIN D3
#define TX_PIN D4

// Nếu bạn sử dụng Esp32
// #define RX_PIN 12
// #define TX_PIN 14

// Tạo một đối tượng espsoftwareserial với tốc độ baud là 9600
SoftwareSerial mySerial (RX_PIN, TX_PIN);

#define wifi_ssid " IoT LAB "
#define wifi_password " kvt1ptit "
#define mqtt_server " 10.170.69.245 " // địa chỉ IP của MQTT Broker

WiFiClient espClient;
PubSubClient client (espClient);

// Kết nối tới WiFi
void setup_wifi () {
delay ( 10 );
  Serial . println ();
  Serial . print ( " Connecting to " );
  Serial . println (wifi_ssid);

  WiFi . begin (wifi_ssid, wifi_password);

  while ( WiFi . status () != WL_CONNECTED) {
    delay ( 500 );
    Serial . print ( " . " );
  }

  Serial . println ( "" );
  Serial . println ( " WiFi connected " );
  Serial . println ( " IP address: " );
  Serial . println ( WiFi . localIP ());
}

// Kết nối tới MQTT Broker
void reconnect () {
  while (! client . connected ()) {
    Serial . print ( " Attempting MQTT connection … " );
```

```
    if ( client . connect ( " IoTLABClient " )) {
      Serial . println ( " connected " );
      client . subscribe ( " Nhóm của bạn " ); // tự đặt topic Sub theo nhóm của bạn. Ví
dụ “ sensor1 ”
    } else {
      Serial . print ( " failed, rc= " );
      Serial . print ( client . state ());
      Serial . println ( " try again in 5 seconds " );
      delay ( 5000 );
    }
  }
}

//-----Method for Publishing MQTT Messages---------
void publishMessage ( const char* topic, String payload, boolean retained) {
  if ( client . publish (topic, payload . c_str (), true ))
    Serial . println ( " Message published [ " + String (topic) + " ]: " + payload);
}

void setup () {
  // Khởi tạo cổng nối tiếp phần cứng với tốc độ baud là 115200
  Serial . begin ( 115200 );

  // Khởi tạo cổng nối tiếp phần mềm với tốc độ baud là 9600
  mySerial . begin ( 9600 );

  setup_wifi ();
  client . setServer (mqtt_server, 1883 );
}

DynamicJsonDocument doc ( 1024 );

void loop () {
  if (! client . connected ()) {
    reconnect ();
  }
  client . loop ();
  // nhận data từ Arduino thông qua Tx-Rx và sau đó publish data lên MQTT Broker
  if ( mySerial . available ()) {
    String data = mySerial . readStringUntil ( ' \n ' );
    DeserializationError error = deserializeJson (doc, data);
    char mqtt_message [ 1024 ];
```

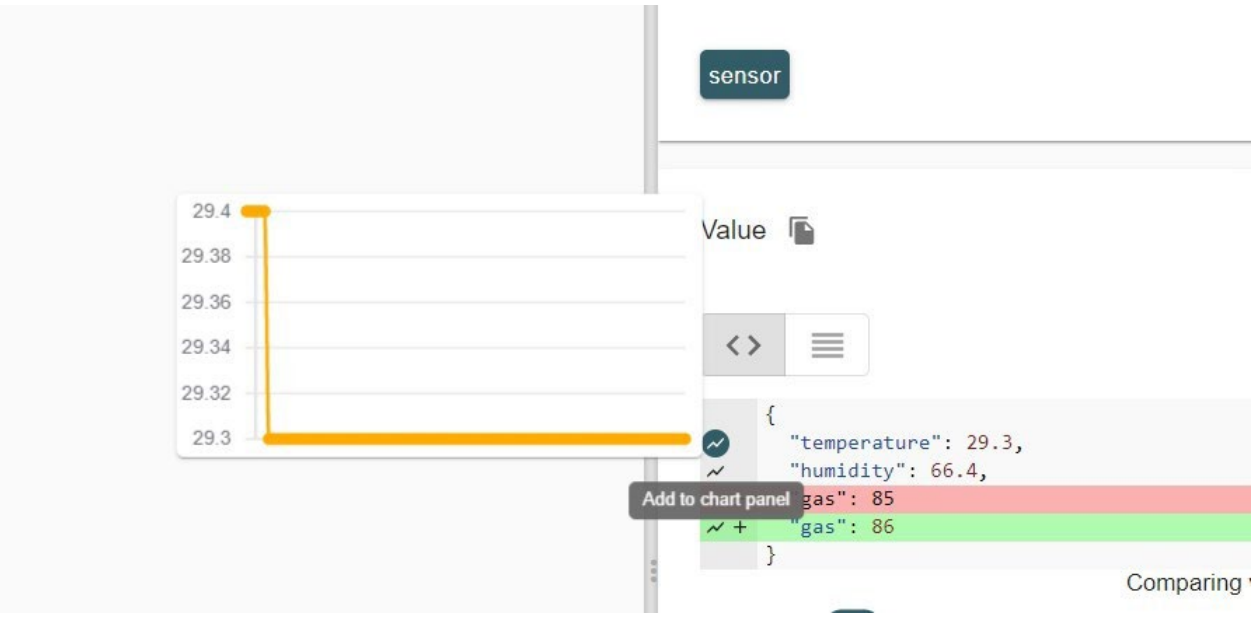
```
    serializeJson (doc, mqtt_message);
    publishMessage ( " Nhóm của bạn " , mqtt_message, true ); // tự đặt topic Pub theo nhóm của bạn. Ví dụ “ sensor1"
  }
}
```

Kết quả trên MQTT Explorer

Để quan sát được bản tin đã gửi đến được MQTT Broker hay chưa ta sử dụng phần mềm MQTT Explorer :



Cách vẽ biểu đồ:



6. Thực hành kéo bản tin từ MÁY CHỦ(BROKER) xuống KIT(SUBSCRIBER) sử dụng giao thức MQTT

Sinh viên thực hành kéo bản tin từ Broker Server (bản tin là dữ liệu DHT / MQ-2 mà sinh viên đã gửi lên từ nội dung 2)

Chuẩn bị phần cứng

- Cảm biến DHT11/DHT22, cảm biến khí gas MQ-2
- Board Arduino Uno
- Chip WiFi D1 mini Lite
- Dây cáp nạp code cho arduino và chip WiFi
- LCD
- Bus

Sơ đồ đấu nối

- **Sơ đồ đấu nối LCD-Arduino Uno: sinh viên xem lại buổi 1**
-

Arduino	LCD
Vcc	5V
GND	GND
SDA	SDA
SCL	SCL

- **Sơ đồ đấu nối Arduino- Chip WiFi D1 mini Lite**

Arduino	D1 Mini
Vcc	5V

GND	GND
D10	D4
D11	D3

Code tham khảo

Code nạp cho Arduino Uno

Có nhiệm vụ là thu thập dữ liệu từ cảm biến DHT22 và cảm biến khí gas MQ-2 và gửi dữ liệu cho Chip WiFi D1 mini Lite thông qua giao thức kết nối UART. Sau đó nhận data từ Chip WiFi D1 mini Lite lấy từ MQTT Broker để hiển thị lên LCD:

```
#include <DHT.h>
#include <LiquidCrystal_I2C.h>
#include <ArduinoJson.h>
#include <SoftwareSerial.h>

#define DHT_Data 8
DHT dht (DHT_Data, DHT22);  // DHT11

#define MQ_Data A1

#define RX_PIN 10  // Chân RX của cổng nối tiếp mềm
#define TX_PIN 11  // Chân TX của cổng nối tiếp mềm

SoftwareSerial mySerial (RX_PIN, TX_PIN);  // Tạo một đối tượng cổng nối tiếp mềm

LiquidCrystal_I2C lcd (0x 27 , 16 , 2 );  // or set address on 0x20

void setup () {
  Serial . begin ( 9600 );
  mySerial . begin ( 9600 );
  while (!Serial) delay ( 1 );

  dht . begin ();

  lcd . init ();
  lcd . backlight ();
}

unsigned long timeUpdata = millis ();

DynamicJsonDocument doc ( 1024 );
DynamicJsonDocument docSub ( 1024 );
```

```
void loop () {
  if ( millis () - timeUpdata > 2000 ) {
    float hum = dht . readHumidity ();
    float temp = dht . readTemperature ();
    float gas = analogRead (MQ_Data);

    doc [ "temperature" ] = temp;
    doc [ "humidity" ] = hum;
    doc [ "gas" ] = gas;

    // gửi data thông qua cổng giao tiếp mềm cho Chip WiFi
    serializeJson (doc, mySerial);
    mySerial . println ();

    // In ra màn hình Serial
    serializeJson (doc, Serial);
    Serial . println ();

    timeUpdata = millis ();
  }

  // nhận data được gửi về từ Chip WiFi và hiển thị ra màn hình LCD
  if ( mySerial . available ()) {
    String data = mySerial . readStringUntil (' \n ');
    DeserializationError error = deserializeJson (docSub, data);

    float tempSub = docSub [ "temperature" ];
    float humSub = docSub [ "humidity" ];
    float gasSub = docSub [ "gas" ];

    lcd . clear ();
    lcd . setCursor ( 0 , 0 );
    lcd . print ( "T: " );
    lcd . print (tempSub);
    lcd . print ( " C" );
    lcd . setCursor ( 9 , 0 );
    lcd . print ( "H: " );
    lcd . print (humSub);
    lcd . print ( " %" );
    lcd . setCursor ( 0 , 1 );
    lcd . print ( "G: " );
    lcd . print (gasSub);
  }
```

```
}
```

Code nạp cho Chip WiFi D1 mini Lite

Có nhiệm vụ nhận data từ Arduino Uno thông qua chuẩn giao thức UART để gửi data lên MQTT Broker (lúc này đóng vai trò là Publisher). Sau đó lấy data về từ MQTT Broker (lúc này đóng vai trò là Subscriber) để truyền data về cho Arduino Uno: **nhớ sửa topic khi publish và khi subscribe data từ Broker Server.**

Nếu sử dụng ESP32 thì sửa cổng nối Tx-Rx

Chú ý cài Lib cho board ESP8266/32

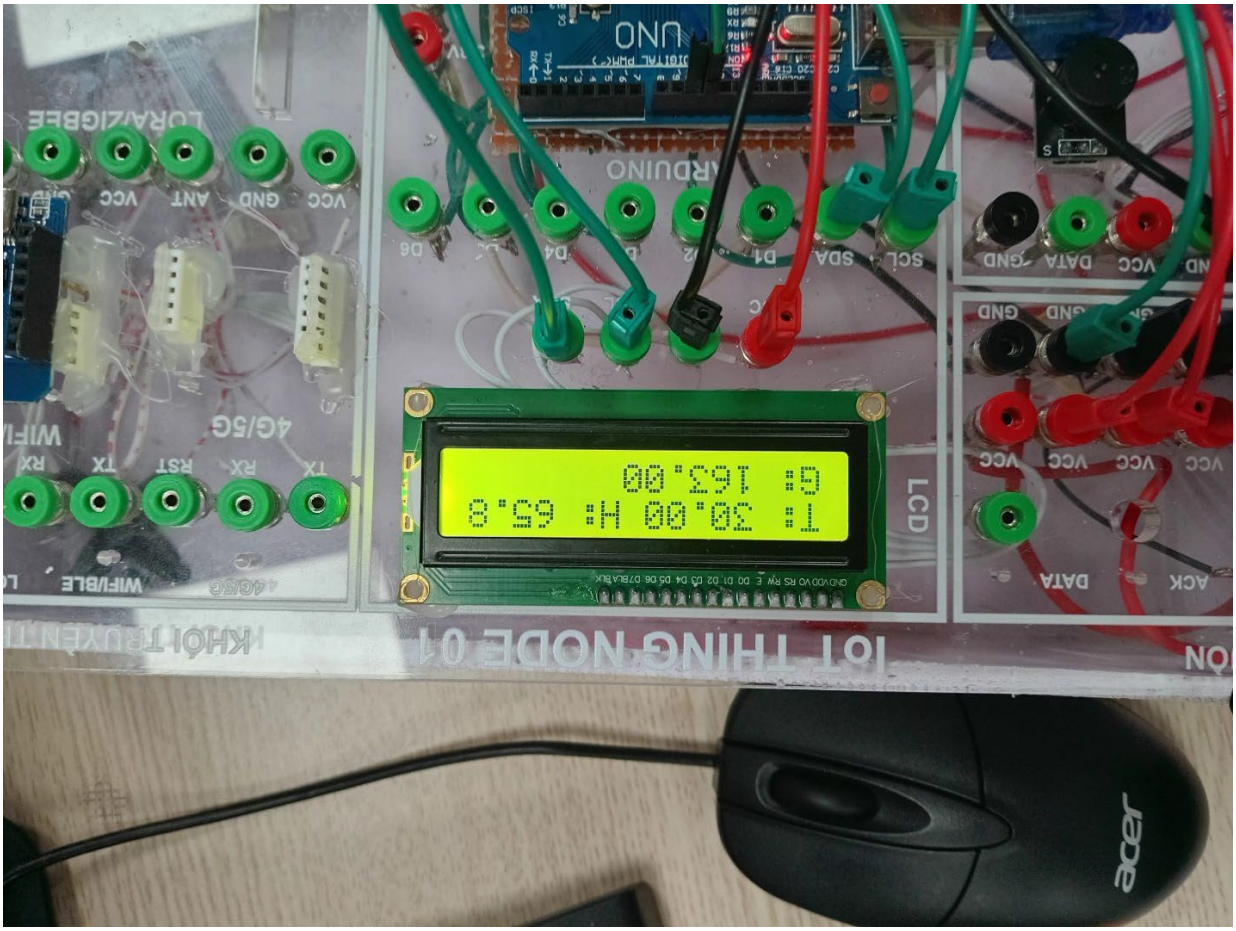
```
#ifdef ESP8266
#include <ESP8266WiFi.h> /* WiFi library for ESP8266 */
#else
#include <WiFi.h> /* WiFi library for ESP32 */
#endif
#include <Wire.h>
#include <SoftwareSerial.h>
#include <ArduinoJson.h>
#include <PubSubClient.h>
#include <WiFiClientSecure.h>

// Khai báo các chân RX và TX cho espsoftwareserial
// Nếu bạn dùng Chip WiFi D1 mini Lite
#define RX_PIN D3
#define TX_PIN D4

// Nếu bạn sử dụng Esp32
// #define RX_PIN 12
// #define TX_PIN 14

// Tạo một đối tượng espsoftwareserial với tốc độ baud là 9600
SoftwareSerial mySerial (RX_PIN, TX_PIN);
DynamicJsonDocument doc ( 1024 );

#define wifi_ssid "IoT LAB"
#define wifi_password "kvt1ptit"
#define mqtt_server "10.170.69.245" // địa chỉ IP của MQTT Broker

WiFiClient espClient;
PubSubClient client (espClient);

// Kết nối tới WiFi
void setup_wifi () {
delay ( 10 );
```

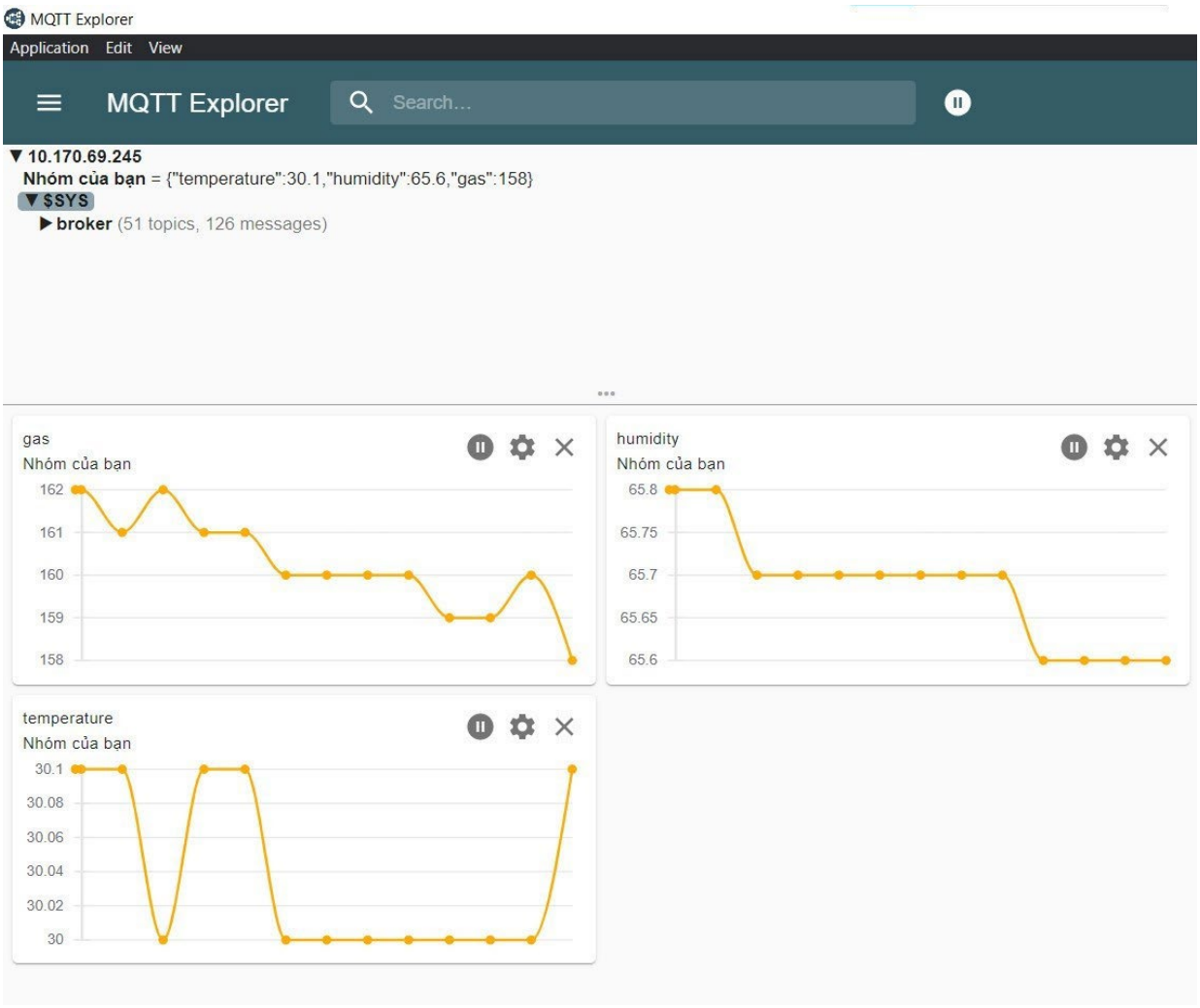
```
  Serial . println ();
  Serial . print ( "Connecting to " );
  Serial . println (wifi_ssid);

  WiFi . begin (wifi_ssid, wifi_password);

  while ( WiFi . status () != WL_CONNECTED) {
    delay ( 500 );
    Serial . print ( "." );
  }

  Serial . println ( "" );
  Serial . println ( "WiFi connected" );
  Serial . println ( "IP address: " );
  Serial . println ( WiFi . localIP ());
}

// Kết nối tới MQTT Broker
void reconnect () {
  while (! client . connected ()) {
    Serial . print ( "Attempting MQTT connection..." );

    if ( client . connect ( "IoTLABClient" )) {
      Serial . println ( "connected" );
      client . subscribe ( "Nhóm của bạn" ); // tự đặt topic Sub theo nhóm của bạn. Ví dụ
"sensor1"
    } else {
      Serial . print ( "failed, rc=" );
      Serial . print ( client . state ());
      Serial . println ( " try again in 5 seconds" );
      delay ( 5000 );
    }
  }
}

//----------Call back Method for Receiving MQTT massage and Switch LED---------
void callback ( char* topic, byte * payload, unsigned int length) {
  String incommingMessage = "" ;
  for ( int i = 0 ; i < length; i++) incommingMessage += ( char ) payload [i];
  Serial . println ( "Massage arived [" + String (topic) + "]" + incommingMessage);

  DynamicJsonDocument docSub ( 1024 );
  DeserializationError error = deserializeJson (docSub, incommingMessage);
```

```
  if (error) {
    Serial . print ( "deserializeJson() failed: " );
    Serial . println ( error . c_str ());
    return ;
  }

  serializeJson (docSub, mySerial);
  mySerial . println ();
}

//-----Method for Publishing MQTT Messages---------
void publishMessage ( const char* topic, String payload, boolean retained) {
  if ( client . publish (topic, payload . c_str (), true ))
    Serial . println ( "Message published [" + String (topic) + "]: " + payload);
}

void setup () {
  // Khởi tạo cổng nối tiếp phần cứng với tốc độ baud là 115200
  Serial . begin ( 115200 );

  // Khởi tạo cổng nối tiếp phần mềm với tốc độ baud là 9600
  mySerial . begin ( 9600 );

  setup_wifi ();
  client . setServer (mqtt_server, 1883 );
  client . setCallback (callback);
}

void loop () {
  if (! client . connected ()) {
    reconnect ();
  }
  client . loop ();
  // nhận data từ Arduino thông qua Tx-Rx và sau đó publish data lên MQTT Broker
  if ( mySerial . available ()) {
    String data = mySerial . readStringUntil (' \n ');
    DeserializationError error = deserializeJson (doc, data);
    char mqtt_message [ 1024 ];

    serializeJson (doc, mqtt_message);
    publishMessage ( "Nhóm của bạn" , mqtt_message, true ); // tự đặt topic Pub theo
nhóm của bạn. Ví dụ "sensor1"
```

```
    serializeJson (doc, Serial);
    Serial . println ();
  }
}
```

Kết quả




MQTT Explorer
Application Edit View
MQTT Explorer
Search...
▼ 10.170.69.245
Nhóm của bạn = {"temperature":30.1,"humidity":65.6,"gas":158}
▼ $SYS
▶ broker (51 topics, 126 messages)
gas
Nhóm của bạn
162
161
160
159
158
humidity
Nhóm của bạn
65.8
65.75
65.7
65.65
65.6
temperature
Nhóm của bạn
30.1
30.08
30.06
30.04
30.02
30