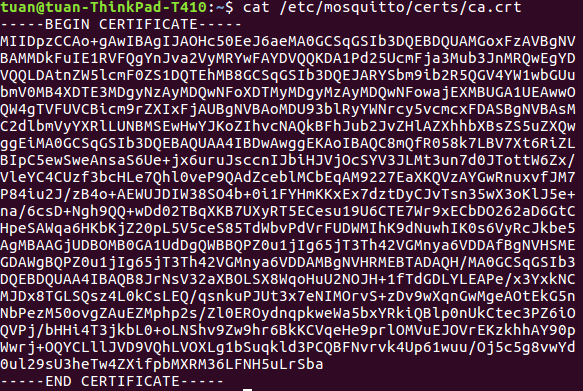
# **[Demo 30: How to use Arduino ESP32 MQTTS with MQTTS Mosquitto broker (TLS/SSL)](http://www.iotsharing.com/2017/08/how-to-use-esp32-mqtts-with-mqtts-mosquitto-broker-tls-ssl.html)**

**1. Introduction**  
- In this tutorial, I will show you how to use ESP32 MQTTS with MQTTS Mosquitto broker (TLS/SSL).  
- In order to make this tutorial, please refer topics:  
[How to set up secure transportation for MQTT Mosquitto broker with SSL/TLS](http://www.iotsharing.com/2017/08/how-to-set-up-secure-transportation-for-mqtt-mosquitto-broker-tls-ssl.html)  
[Demo 29: How to use HTTPS in Arduino ESP32](http://www.iotsharing.com/2017/08/how-to-use-https-in-arduino-esp32.html)  
[Demo 14: How to use MQTT and Arduino ESP32 to build a simple Smart home system](http://www.iotsharing.com/2017/05/how-to-use-mqtt-to-build-smart-home-arduino-esp32.html)  
- The requirement for this demo: ESP32 with a LED on it will turn On/Off when subscribing a topic "**smarthome/room1/led**" with values (0: off, 1: on).  
**2. Steps**  
- After installing [How to set up secure transportation for MQTT Mosquitto broker with SSL/TLS](http://www.iotsharing.com/2017/08/how-to-set-up-secure-transportation-for-mqtt-mosquitto-broker-tls-ssl.html), you have to extract the content of**ca.crt** for SSL/TLS handshake phase by running the command: **cat /etc/mosquitto/certs/ca.crt**

[](https://1.bp.blogspot.com/-hBjrU0659Ug/WaI9bnggyeI/AAAAAAAAEPM/BT8tCix2F88jBZOPN4g396LlqrZQ_RsVgCLcBGAs/s1600/esp32-mqtts-2.png)

**Figure: Extract content of ca.crt**

- Get the IP address of machine that run MQTT mosquitto by running command: **ifconfig** or if you use MDNS for that machine you can follow this tutorial to get the IP directly from ESP32: [How to get the IP address of a node by its mdns host name in Arduino ESP32](http://www.iotsharing.com/2017/06/how-to-get-ip-address-from-mdns-host-name-in-arduino-esp32.html).

- We will use **WiFiClientSecure**class for SSL/TLS handshake phase and**PubSubClient**library for ESP32 MQTT communication.Please see the topics that I mentioned above.

|  |
| --- |
| #include <WiFi.h>  #include <WiFiClientSecure.h>  #include <PubSubClient.h>  #include <ESPmDNS.h>  /\* change it with your ssid-password \*/  const char\* ssid = "dd-wrt";  const char\* password = "0000000000";  /\* this is the MDNS name of PC where you installed MQTT Server \*/  const char\* serverHostname = "iotsharing";  const char\* ca\_cert = \  "-----BEGIN CERTIFICATE-----\n" \  "MIIDpzCCAo+gAwIBAgIJAOHc50EeJ6aeMA0GCSqGSIb3DQEBDQUAMGoxFzAVBgNV\n" \  "BAMMDkFuIE1RVFQgYnJva2VyMRYwFAYDVQQKDA1Pd25UcmFja3Mub3JnMRQwEgYD\n" \  "VQQLDAtnZW5lcmF0ZS1DQTEhMB8GCSqGSIb3DQEJARYSbm9ib2R5QGV4YW1wbGUu\n" \  "bmV0MB4XDTE3MDgyNzAyMDQwNFoXDTMyMDgyMzAyMDQwNFowajEXMBUGA1UEAwwO\n" \  "QW4gTVFUVCBicm9rZXIxFjAUBgNVBAoMDU93blRyYWNrcy5vcmcxFDASBgNVBAsM\n" \  "C2dlbmVyYXRlLUNBMSEwHwYJKoZIhvcNAQkBFhJub2JvZHlAZXhhbXBsZS5uZXQw\n" \  "ggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQC8mQfR058k7LBV7Xt6RiZL\n" \  "BIpC5ewSweAnsaS6Ue+jx6uruJsccnIJbiHJVjOcSYV3JLMt3un7d0JTottW6Zx/\n" \  "VleYC4CUzf3bcHLe7Qhl0veP9QAdZceblMCbEqAM9227EaXKQVzAYGwRnuxvfJM7\n" \  "P84iu2J/zB4o+AEWUJDIW38SO4b+0i1FYHmKKxEx7dztDyCJvTsn35wX3oKlJ5e+\n" \  "na/6csD+Ngh9QQ+wDd02TBqXKB7UXyRT5ECesu19U6CTE7Wr9xECbDO262aD6GtC\n" \  "HpeSAWqa6HKbKjZ20pL5V5ceS85TdWbvPdVrFUDWMIhK9dNuwhIK0s6VyRcJkbe5\n" \  "AgMBAAGjUDBOMB0GA1UdDgQWBBQPZ0u1jIg65jT3Th42VGMnya6VDDAfBgNVHSME\n" \  "GDAWgBQPZ0u1jIg65jT3Th42VGMnya6VDDAMBgNVHRMEBTADAQH/MA0GCSqGSIb3\n" \  "DQEBDQUAA4IBAQB8JrNsV32aXBOLSX8WqoHuU2NOJH+1fTdGDLYLEAPe/x3YxkNC\n" \  "MJDx8TGLSQsz4L0kCsLEQ/qsnkuPJUt3x7eNIMOrvS+zDv9wXqnGwMgeAOtEkG5n\n" \  "NbPezM50ovgZAuEZMphp2s/Zl0EROydnqpkweWa5bxYRkiQBlp0nUkCtec3PZ6iO\n" \  "QVPj/bHHi4T3jkbL0+oLNShv9Zw9hr6BkKCVqeHe9prlOMVuEJOVrEKzkhhAY90p\n" \  "Wwrj+OQYCLllJVD9VQhLVOXLg1bSuqkld3PCQBFNvrvk4Up61wuu/Oj5c5g8vwYd\n" \  "0ul29sU3heTw4ZXifpbMXRM36LFNH5uLrSba\n" \  "-----END CERTIFICATE-----\n";  /\* create an instance of WiFiClientSecure \*/  WiFiClientSecure espClient;  PubSubClient client(espClient);  /\*LED GPIO pin\*/  const char led = 4;  /\* topics \*/  #define COUNTER\_TOPIC "smarthome/room1/counter"  #define LED\_TOPIC "smarthome/room1/led" /\* 1=on, 0=off \*/  long lastMsg = 0;  char msg[20];  int counter = 0;  void receivedCallback(char\* topic, byte\* payload, unsigned int length) {  Serial.print("Message received: ");  Serial.println(topic);  Serial.print("payload: ");  for (int i = 0; i < length; i++) {  Serial.print((char)payload[i]);  }  Serial.println();  /\* we got '1' -> on \*/  if ((char)payload[0] == '1') {  digitalWrite(led, HIGH);  } else {  /\* we got '0' -> on \*/  digitalWrite(led, LOW);  }  }  void mqttconnect() {  /\* Loop until reconnected \*/  while (!client.connected()) {  Serial.print("MQTT connecting ...");  /\* client ID \*/  String clientId = "ESP32Client";  /\* connect now \*/  if (client.connect(clientId.c\_str())) {  Serial.println("connected");  /\* subscribe topic \*/  client.subscribe(LED\_TOPIC);  } else {  Serial.print("failed, status code =");  Serial.print(client.state());  Serial.println("try again in 5 seconds");  /\* Wait 5 seconds before retrying \*/  delay(5000);  }  }  }  void setup() {  Serial.begin(115200);  // We start by connecting to a WiFi network  Serial.println();  Serial.print("Connecting to ");  Serial.println(ssid);  WiFi.begin(ssid, password);  while (WiFi.status() != WL\_CONNECTED) {  delay(500);  Serial.print(".");  }  /\* set led as output to control led on-off \*/  pinMode(led, OUTPUT);  Serial.println("");  Serial.println("WiFi connected");  Serial.println("IP address: ");  Serial.println(WiFi.localIP());  /\*setup MDNS for ESP32 \*/  if (!MDNS.begin("esp32")) {  Serial.println("Error setting up MDNS responder!");  while(1) {  delay(1000);  }  }  /\* get the IP address of server by MDNS name \*/  Serial.println("mDNS responder started");  IPAddress serverIp = MDNS.queryHost(serverHostname);  Serial.print("IP address of server: ");  Serial.println(serverIp.toString());  /\* set SSL/TLS certificate \*/  espClient.setCACert(ca\_cert);  /\* configure the MQTT server with IPaddress and port \*/  client.setServer(serverIp, 8883);  /\* this receivedCallback function will be invoked  when client received subscribed topic \*/  client.setCallback(receivedCallback);    }  void loop() {  /\* if client was disconnected then try to reconnect again \*/  if (!client.connected()) {  mqttconnect();  }  /\* this function will listen for incomming  subscribed topic-process-invoke receivedCallback \*/  client.loop();  /\* we increase counter every 3 secs  we count until 3 secs reached to avoid blocking program if using delay()\*/  long now = millis();  if (now - lastMsg > 3000) {  lastMsg = now;  if (counter < 100) {  counter++;  snprintf (msg, 20, "%d", counter);  /\* publish the message \*/  client.publish(COUNTER\_TOPIC, msg);  }else {  counter = 0;  }  }  } |