物聯網應用資訊安全與風險管理

MQTT 實習與設計

組員：劉定睿、曾彥輔、羅勻瑄、黃世君

日期：2025-11-19

目錄

背景

MQTT 在物聯網實務專案上, 非常廣泛應用於與工業物聯網設備的通訊
透過 MQTT 的設計, 能夠做到物聯網設備的監看與控制

實習目標

- MQTT broker 的安裝, 包含
- 使用者帳號密碼的設定
- 如何設定 TLS 加密
- 了解如何用 MQTT publish 控制遠端設備
- 了解如何用 MQTT subscribe 接收遠端設備的資訊
- 學習 Topic 如何設計

場景設定

客戶是一個蘭花的養殖場, 一共有六個場域
因為台灣溫度與濕度過高, 不利於蘭花生長, 因此客戶希望打造一個溫室環境, 控制溫度在 20度, 濕度控制在 60%

- 每個場域各有裝 1 支溫度計
- 每個場域各有裝 1 支濕度計
- 每個場域各裝有 2 支風扇, 當場域溫度過高時, 需同時開啟兩支風扇
- 每個場域各裝有 2 台除濕設備, 當場域濕度過高時, 需同時開啟兩台除濕設備

請設計 Topic 內容

1. 透過 subscribe 可監看六個場域個別的即時溫度
2. 透過 subscribe 可監看六個場域個別的即時濕度
3. 當場域溫度過高時, 可透過 publish 同時開啟兩支風扇
4. 當場域溫度過高時, 可透過 publish 同時開啟兩台除濕設備

安裝 Mosquitto Broker

更新軟體包索引：`sudo apt update`
安裝 Mosquitto 及 其客戶端：`sudo apt install -y mosquitto mosquitto-clients`

```
└─$ sudo apt install -y mosquitto mosquitto-clients
mosquitto is already the newest version (2.0.22-3).
mosquitto set to manually installed.
Installing:
  mosquitto-clients

Summary:
  Upgrading: 0, Installing: 1, Removing: 0, Not Upgrading: 287
  Download size: 117 kB
  Space needed: 295 kB / 1,000 GB available
```

啟動並設置 Mosquitto 在系統啟動時自動啟動：

`sudo systemctl start mosquitto`

`sudo systemctl enable mosquitto`

```
└─$ sudo systemctl enable mosquitto
Synchronizing state of mosquitto.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable mosquitto
Created symlink '/etc/systemd/system/multi-user.target.wants/mosquitto.service' → '/usr/lib/systemd/system/mosquitto
service'.
```

設定帳號與密碼

建立 Mosquitto 密碼文件：

`sudo chown root:root /etc/mosquitto/passwd`

`sudo mosquitto_passwd -c /etc/mosquitto/passwd <username>`

```
└─$ sudo mosquitto_passwd -c /etc/mosquitto/passwd jonathan
Password:
Reenter password:
```

修改 Mosquitto 設定檔以啟用帳號密碼認證：

`sudo vim /etc/mosquitto/mosquitto.conf`

在文件中新增以下內容：

```
allow_anonymous false
password_file /etc/mosquitto/passwd
```

```
# Place your local configuration in /etc/mosquitto/conf.d/
#
# A full description of the configuration file is at
# /usr/share/doc/mosquitto/examples/mosquitto.conf

#pid_file /run/mosquitto/mosquitto.pid

persistence true
persistence_location /var/lib/mosquitto/

log_dest file /var/log/mosquitto/mosquitto.log

include_dir /etc/mosquitto/conf.d

allow_anonymous false
password_file /etc/mosquitto/passwd
```

設置 allow_anonymous false 以禁止匿名訪問

賦予權限：

- `sudo chown mosquitto:mosquitto /etc/mosquitto/passwd`
- `sudo chmod 640 /etc/mosquitto/passwd`

重啟 Mosquitto 來套用更改：`sudo systemctl restart mosquitto`

設定加密 (TLS)

創建自簽名憑證或使用受信任的 CA 簽發的憑證：
若無憑證，可使用 openssl 生成：

`sudo openssl req -x509 -nodes -newkey rsa:2048 -keyout mosquitto.key -out mosquitto.crt -days 365`

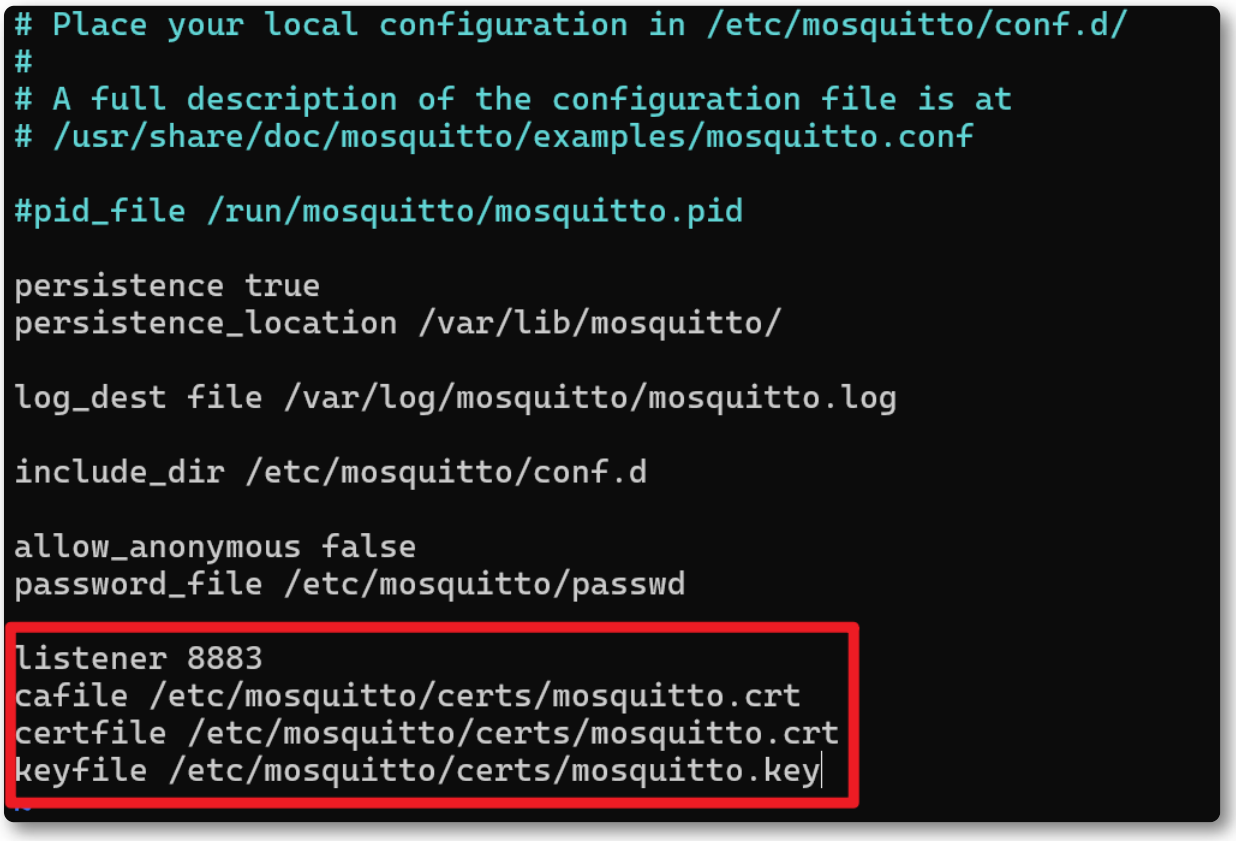
```
┌──$ sudo openssl req -x509 -newkey rsa:2048 -keyout mosquitto.key -out mosquitto.crt -days 365
..+...+......+.+..+++++++++++++++++++++++++++++++++++++++++*.+........+....+........+....+...++++++++++++++++++++++++++
++++++++++++*.......+.......+.........+.....+.........................+......+.+..............+..+..................
+...+....+........+.............+.........+.............+...+......+.........+...+........+............+...+.+.....+..
.+...+.......+..............+..+......+............+.......++++++
.........+...+...+.....+....+...+..+.......+.......+.....+.+..+.........+....++++++++++++++++++++++++++++++++++++++++*.
....+............+......+..+.++++++++++++++++++++++++++++++++++++++++*..+......+.+..+.+.............+..............+...
+..........+..+.+.....+...........+.+..+.......+..+...+...+.......+.....+......+......+.....+..+......+............+....
.....+.....+.........................+.........................+.......+...........+...+..................+..+........
.+.+........+.++++++
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:TW
State or Province Name (full name) [Some-State]:taiwan
Locality Name (eg, city) []:taipei
Organization Name (eg, company) [Internet Widgits Pty Ltd]:ntust
Organizational Unit Name (eg, section) []:ntust
Common Name (e.g. server FQDN or YOUR name) []:ntust
Email Address []:a@ntust.edu.tw
```

將生成的 `mosquitto.key` 和 `mosquitto.crt` 文件放置於安全的目錄，例如

```
sudo mv mosquitto.key mosquitto.crt /etc/mosquitto/certs
```

修改 Mosquitto 設定檔以啟用 TLS：
開啟設定檔：`sudo vim /etc/mosquitto/mosquitto.conf`
增加以下設定來啟用加密通訊：

```
listener 8883
cafile /etc/mosquitto/certs/mosquitto.crt
certfile /etc/mosquitto/certs/mosquitto.crt
keyfile /etc/mosquitto/certs/mosquitto.key
```



listener 8883 指定了加密通訊的端口 (通常為 8883)；可根據需求更改
重啟 Mosquitto 來套用 TLS 配置：

- 檢查私鑰檔權限及擁有者：

```
ls -l /etc/mosquitto/certs/mosquitto.key
```

- 確保檔案擁有者是 mosquitto 用戶，且權限允許該用戶讀取：

```
sudo chown mosquitto:mosquitto /etc/mosquitto/certs/mosquitto.key
sudo chmod 640 /etc/mosquitto/certs/mosquitto.key
```

- 確認父目錄 `/etc/mosquitto/certs` 權限允許 mosquitto 用戶進入：
  `ls -ld /etc/mosquitto/certs`
  `sudo chown -R mosquitto:mosquitto /etc/mosquitto/certs`
  `sudo chmod 750 /etc/mosquitto/certs`

設定完權限後，重新啟動 mosquitto 服務：
`sudo systemctl restart mosquitto.service`

測試 MQTT Broker 設定

使用客戶端測試加密通訊及帳號密碼認證：
`mosquitto_pub -h localhost -p 8883 -t test/topic -m "Hello MQTT" --cafile /etc/mosquitto/certs/mosquitto.crt -u <username> -P <password>`

將 `<username>` 與 `<password>` 替換為你的用戶名稱和密碼
若客戶端能成功發送訊息，則說明設定成功

Topic 實作

Topic 1: 透過 subscribe 可監看六個場域個別的即時溫度

Topic 命名規則：系統/場域/資料類型

```
orchid/field1/temperature
orchid/field2/temperature
...
orchid/field6/temperature
```

- orchid → 系統主題（蘭花溫室監控系統）
- fieldX → 場域編號（六個場域）
- temperature → 感測類型（溫度）
- 資料載荷（payload）使用簡單數值："23.7"（輕量、直覺）

實作角色	任務	使用指令
Publisher（溫度感測器）	上報各場域即時溫度	`mosquitto_pub`
Subscriber（監控端）	監看所有場域的溫度變化	`mosquitto_sub`

1. 啟動訂閱端（Subscribe）
   目的：監看所有場域的溫度資料
   指令：

```
mosquitto_sub -h localhost -p 8883 --cafile /opt/homebrew/etc/mosquitto/certs/mos
```

```
(~/MQTT)                                                                    (sharonlo@sharonM2:s001)
(19:28:19)—> mosquitto_sub -h localhost -p 8883 --cafile /opt/homebrew/etc/mosquitto/certs/mosquitto.crt --tls-version
tlsv1.3 --insecure -t "orchid/+/temperature" -q 1 -v -u orchid_user -P
```

2. 啟動發佈端（Publish）

   目的：模擬某一場域的溫度感測器上傳資料

   這邊先設定 field1 溫度為 25.6 測試是否訂閱端有收到

   指令：

```
mosquitto_pub -h localhost -p 8883 --cafile /opt/homebrew/etc/mosquitto/certs/mos
```

```
(~)                                                                         (sharonlo@sharonM2:s002)
(19:46:33)—> mosquitto_pub -h localhost -p 8883 --cafile /opt/homebrew/etc/mosquitto/certs/mosquitto.crt --tls-version
tlsv1.3 --insecure -t "orchid/field1/temperature" -m "25.6" -u orchid_user -P
```

3. 驗證結果

   在訂閱端終端機畫面會即時顯示 `orchid/field1/temperature 25.6` 表示發佈成功

```
(~/MQTT)                                                                    (sharonlo@sharonM2:s001)
(19:28:19)—> mosquitto_sub -h localhost -p 8883 --cafile /opt/homebrew/etc/mosquitto/certs/mosquitto.crt --tls-version
tlsv1.3 --insecure -t "orchid/+/temperature" -q 1 -v -u orchid_user -P

orchid/field1/temperature 25.6
```

   若驗證成功，則繼續以一樣的指令模擬剩下五個場域發佈溫度

```
(~)                                                                         (sharonlo@sharonM2:s002)
(19:46:33)—> mosquitto_pub -h localhost -p 8883 --cafile /opt/homebrew/etc/mosquitto/certs/mosquitto.crt --tls-version
tlsv1.3 --insecure -t "orchid/field1/temperature" -m "25.6" -u orchid_user -P
(~)                                                                         (sharonlo@sharonM2:s002)
(19:50:08)—> mosquitto_pub -h localhost -p 8883 --cafile /opt/homebrew/etc/mosquitto/certs/mosquitto.crt --tls-version
tlsv1.3 --insecure -t "orchid/field2/temperature" -m "25.5" -u orchid_user -P
(~)                                                                         (sharonlo@sharonM2:s002)
(19:57:34)—> mosquitto_pub -h localhost -p 8883 --cafile /opt/homebrew/etc/mosquitto/certs/mosquitto.crt --tls-version
tlsv1.3 --insecure -t "orchid/field3/temperature" -m "25.4" -u orchid_user -P
(~)                                                                         (sharonlo@sharonM2:s002)
(19:57:43)—> mosquitto_pub -h localhost -p 8883 --cafile /opt/homebrew/etc/mosquitto/certs/mosquitto.crt --tls-version
tlsv1.3 --insecure -t "orchid/field4/temperature" -m "25.7" -u orchid_user -P
(~)                                                                         (sharonlo@sharonM2:s002)
(19:57:55)—> mosquitto_pub -h localhost -p 8883 --cafile /opt/homebrew/etc/mosquitto/certs/mosquitto.crt --tls-version
tlsv1.3 --insecure -t "orchid/field5/temperature" -m "25.8" -u orchid_user -P
(~)                                                                         (sharonlo@sharonM2:s002)
(19:58:06)—> mosquitto_pub -h localhost -p 8883 --cafile /opt/homebrew/etc/mosquitto/certs/mosquitto.crt --tls-version
tlsv1.3 --insecure -t "orchid/field6/temperature" -m "25.9" -u orchid_user -P
```

   訂閱端會依序收到剩下五個場域發佈的溫度

```
(~/MQTT)                                                                    (sharonlo@sharonM2:s001)
(19:28:19)—> mosquitto_sub -h localhost -p 8883 --cafile /opt/homebrew/etc/mosquitto/certs/mosquitto.crt --tls-version
tlsv1.3 --insecure -t "orchid/+/temperature" -q 1 -v -u orchid_user -P

orchid/field1/temperature 25.6
orchid/field2/temperature 25.5
orchid/field3/temperature 25.4
orchid/field4/temperature 25.7
orchid/field5/temperature 25.8
orchid/field6/temperature 25.9
```

四、實驗結論與觀察

- 使用 `orchid/+/temperature` 可同時監看六個場域個別的即時溫度。
- MQTT 傳輸與 TLS 加密連線運作正常。
- 成功利用 MQTT 架構設計多場域資料通道。
- 使用 `+` 萬用字元可同時監看多個場域。
- TLS 8883 埠運作正常，自簽憑證確保傳輸安全。

Topic 2. 透過 subscribe 可監看六個場域個別的即時濕度

1. Topic 設計

為了清楚區分場域，每個場域用一個 Topic，命名規則如下：

```
orchid/area1/humidity
orchid/area2/humidity
orchid/area3/humidity
orchid/area4/humidity
orchid/area5/humidity
orchid/area6/humidity
```

- orchid → 系統名稱
- areaX → 場域編號 1~6
- humidity → 表示濕度資訊

2. 開啟兩個terminal，一個訂閱監看端、一個發布端

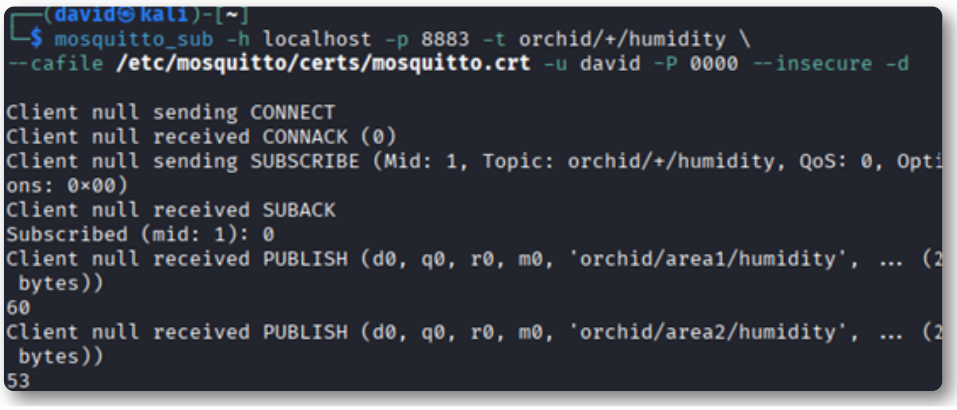
3. 監看場域

```
mosquitto_sub -h localhost -p 8883 -t orchid/+/humidity \
  --cafile /etc/mosquitto/certs/mosquitto.crt -u david -P 0000 --insecure -d
```

orchid/+/humidity → + 代表任意單層場域名稱，會同時收到 area1~area6 的濕度訊息

```
mosquitto_sub -h localhost -p 8883 -t orchid/# \
  --cafile /etc/mosquitto/certs/mosquitto.crt -u david -P 0000 --insecure -d
```

orchid/# → # 代表該系統下所有 topic，包括溫度、濕度、控制訊息



4. 發布端

```
mosquitto_pub -h localhost -p 8883 -t orchid/area1/humidity -m "58" \
--cafile /etc/mosquitto/certs/mosquitto.crt -u david -P 0000 --insecure -d
```

5. 實驗結果

修改格式:

- awk 負責把 Topic + Payload 格式化成 Topic : 值%
- -d為debug mode 會顯示連線資訊，不使用結果更可視化

發布端

```
mosquitto_pub -h localhost -p 8883 -t orchid/area1/humidity -m "58" --cafile /et
mosquitto_pub -h localhost -p 8883 -t orchid/area2/humidity -m "62" --cafile /et
mosquitto_pub -h localhost -p 8883 -t orchid/area3/humidity -m "60" --cafile /et
```

監控端

```
┌──(david㉿kali)-[~]
└─$ mosquitto_sub -h localhost -p 8883 -t "orchid/+/humidity" \
--cafile /etc/mosquitto/certs/mosquitto.crt -u david -P 0000 --insecure -v \
| awk '{print $1 " : " $2 "%"}'

orchid/area1/humidity : 58%
orchid/area2/humidity : 62%
orchid/area3/humidity : 60%
```

6. 實驗結論

- MQTT Topic 設計與資料監控可行性
  成功建立六個場域的濕度監控 Topic
- TLS 加密連線設定
  成功啟用 TLS (port 8883)，確保資料傳輸安全。
  需要注意私鑰是否有密碼保護，以及 `mosquitto_sub` 與 `mosquitto_pub` 是否使用 –insecure 或正確的 CA 憑證來驗證。
- 資料發布與格式化
  使用 `mosquitto_pub -t <topic> -m <value>` 成功發布濕度值。
  在訂閱端使用 -v 參數將 Topic 與 Payload 顯示在同一行，搭配 awk 進行格式化，成功得到直覺可讀的顯示格式

Topic 3. 當場域溫度過高時, 可透過 publish 同時開啟兩支風扇

準備工作

1. 開啟**三個**獨立的 Terminal。
2. 確保 Mosquitto broker 正在 `localhost` 上運行。

在終端機 1 啟動風扇 1 (訂閱者)

- 模擬第一支風扇，使其開始監聽指令。

```
mosquitto_sub -h localhost -p 8883 -t "field/fans/command" --cafile /etc/mosquit
```

- **說明：**
  - `mosquitto_sub`：訂閱者指令。
  - `-h localhost`：連接到本地的 broker。
  - `-t "field/fans/command"`：訂閱名為 "field/fans/command" 的主題。
  - `-v`：顯示詳細資訊（這樣我們才能看到接收到的訊息及其主題）。
- **狀態：** 此終端機現在會停住，等待訊息。

```
jona@jonafk:~$ mosquitto_sub -h localhost -p 8883 -t "field/fans/command" --cafile /etc/mosqui
tto/certs/mosquitto.crt -u jonathan -P 0000 --insecure -d
Client (null) sending CONNECT
Client (null) received CONNACK (0)
Client (null) sending SUBSCRIBE (Mid: 1, Topic: field/fans/command, QoS: 0, Options: 0x00)
Client (null) received SUBACK
Subscribed (mid: 1): 0
```

在終端機 2 啟動風扇 2 (訂閱者)

- 目的：模擬第二支風扇，使其監聽的指令。

```
mosquitto_sub -h localhost -p 8883 -t "field/fans/command" --cafile /etc/mosquit
```

- **說明：**
  - 此指令與步驟一完全相同。這表示兩支風扇都在監聽同一個廣播頻道。
- **狀態：** 此終端機現在也會停住，等待訊息。

```
jona@jonafk:~$ mosquitto_sub -h localhost -p 8883 -t "field/fans/command" --cafile /etc/mosqui
tto/certs/mosquitto.crt -u jonathan -P 0000 --insecure -d
Client (null) sending CONNECT
Client (null) received CONNACK (0)
Client (null) sending SUBSCRIBE (Mid: 1, Topic: field/fans/command, QoS: 0, Options: 0x00)
Client (null) received SUBACK
Subscribed (mid: 1): 0
```

在終端機 3 發布指令 (發布者)

- **目的：** 模擬溫度感測器或控制中心，在溫度過高時發布「開啟」指令。
- **指令：** 複製並貼上此指令到**第三個**終端機視窗，然後按下 Enter。

```
mosquitto_pub -h localhost -p 8883 -t "field/fans/command" -m "ON" --cafile /etc/
 --tls-version tlsv1.3 -u jonathan -P 0000 --insecure
```

- **說明：**
  - `mosquitto_pub`：發布者指令。
  - `-h localhost`：連接到本地的 broker。
  - `-t "field/fans/command"`：將訊息發布到這個主題。
  - `-m "ON"`：發送的訊息內容（Payload）是 "ON"。
- **狀態：** 此指令執行後會立即結束

```
jona@jonafk:~$ mosquitto_pub -h localhost -p 8883 -t "field/fans/command" -m "ON" --cafile /etc/mosquitto/certs/mosquitto.crt
 --tls-version tlsv1.3 -u jonathan -P 0000 --insecure
```

驗證結果

終端機 1 ：

```
field/fans/command ON
```

```
jona@jonafk:~$ mosquitto_sub -h localhost -p 8883 -t "field/fans/command" --cafile /etc/mosqui
tto/certs/mosquitto.crt -u jonathan -P 0000 --insecure -d
Client (null) sending CONNECT
Client (null) received CONNACK (0)
Client (null) sending SUBSCRIBE (Mid: 1, Topic: field/fans/command, QoS: 0, Options: 0x00)
Client (null) received SUBACK
Subscribed (mid: 1): 0
Client (null) received PUBLISH (d0, q0, r0, m0, 'field/fans/command', ... (2 bytes))
ON
```

終端機 2 ：

```
field/fans/command ON
```

```
jona@jonafk:~$ mosquitto_sub -h localhost -p 8883 -t "field/fans/command" --cafile /etc/mosqui
tto/certs/mosquitto.crt -u jonathan -P 0000 --insecure -d
Client (null) sending CONNECT
Client (null) received CONNACK (0)
Client (null) sending SUBSCRIBE (Mid: 1, Topic: field/fans/command, QoS: 0, Options: 0x00)
Client (null) received SUBACK
Subscribed (mid: 1): 0
Client (null) received PUBLISH (d0, q0, r0, m0, 'field/fans/command', ... (2 bytes))
ON
```

Topic 4. 當場域溫度過高時, 可透過 publish 同時開啟兩台除濕設備

目的

在單一 Linux 虛擬機上，建置一個具備 TLS 加密 和 用戶身份驗證 的 Mosquitto Broker，並設計一個 Python 自動化控制器，實現「當任一場域溫度超過 $30^{\circ}C$ 時，自動發布指令開啟兩台除濕設備」的物聯網控制迴路。

I. 建立 Python 自動化控制器環境

為了避免系統環境衝突 (externally-managed-environment 錯誤)，我們建立一個隔離的虛擬環境。

1. 建立並進入虛擬環境

- 安裝 venv 工具： `sudo apt install python3-venv`
- 建立虛擬環境： `python3 -m venv mqtt_venv`
- 進入虛擬環境： `source mqtt_venv/bin/activate`

```
username@myserversname:~$ sudo apt install python3-venv
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
python3-venv is already the newest version (3.12.3-0ubuntu2.1).
0 upgraded, 0 newly installed, 0 to remove and 28 not upgraded.
```

2. 安裝 Paho-MQTT 函式庫安裝函式庫： `pip install paho-mqtt`

```
(mqtt_venv) username@myserversname:~$ pip install paho-mqtt
Collecting paho-mqtt
  Downloading paho_mqtt-2.1.0-py3-none-any.whl.metadata (23 kB)
Downloading paho_mqtt-2.1.0-py3-none-any.whl (67 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 67.2/67.2 kB 1.5 MB/s eta 0:00:00
Installing collected packages: paho-mqtt
Successfully installed paho-mqtt-2.1.0
```

3. 編寫自動化控制器程式 (auto_controller.py)撰寫一個 Python 程式，扮演邏輯大腦的角色：

- 訂閱： 監聽所有場域的溫度 (orchid/+/temperature)。
- 邏輯： 判斷接收到的溫度是否超過 $30.0^{\circ}\mathrm{C}$。
- 發布： 若溫度過高，同時向兩台除濕設備的主題發布 "ON" 指令 (control/dehumidifier1/power 和 control/dehumidifier2/power)。

- 發布： 若溫度恢復正常且設備為開啟狀態，發布 "OFF" 指令。

```python
import paho.mqtt.client as mqtt

# --- 配置參數 (請修改為您的實際值) ---
MQTT_BROKER_HOST = "localhost"
MQTT_PORT = 8883
CA_CERTS_PATH = "/etc/mosquitto/certs/mosquitto.crt"
USERNAME = "username"       # 替換為您的 MQTT 用戶名
PASSWORD = '[redacted]'  # 替換為您的 MQTT 密碼
TEMP_THRESHOLD = 30.0     # 溫度過高閾值 (例如 30.0 攝氏度)

# --- 要控制的除濕機主題 ---
DEHUMIDIFIER_TOPIC_1 = "control/dehumidifier1/power"
DEHUMIDIFIER_TOPIC_2 = "control/dehumidifier2/power"
SUBSCRIPTION_TOPIC = "orchid/+/temperature"

# --- 全局變數：追蹤除濕機狀態，避免重複發送指令 ---
DEHUMIDIFIERS_ON = False

def on_connect(client, userdata, flags, rc):
    """連線成功時的回調函數"""
    if rc == 0:
        print("連線成功！開始監測溫度...")
        # 訂閱所有場域的溫度主題
        client.subscribe(SUBSCRIPTION_TOPIC, qos=1)
    else:
        print(f"連線失敗，返回代碼: {rc}")

def on_message(client, userdata, msg):
    """收到訂閱消息時的回調函數"""
```

```
    global DEHUMIDIFIERS_ON

    try:
        # 取得主題和負載 (payload)
        topic = msg.topic
        temperature_str = msg.payload.decode("utf-8")
        temperature = float(temperature_str)

        print(f"[{topic}] 收到溫度: {temperature}°C")

        # --- 判斷邏輯 ---
        if temperature > TEMP_THRESHOLD and not DEHUMIDIFIERS_ON:

            print(f"🚨 溫度 {temperature}°C 超過閾值 {TEMP_THRESHOLD}°C！發出開啟指令...")

            # 發布開啟指令給兩台設備
            client.publish(DEHUMIDIFIER_TOPIC_1, "ON", qos=1)
            client.publish(DEHUMIDIFIER_TOPIC_2, "ON", qos=1)

            DEHUMIDIFIERS_ON = True  # 標記狀態已開啟

        elif temperature <= TEMP_THRESHOLD and DEHUMIDIFIERS_ON:

            print(f"✅ 溫度 {temperature}°C 恢復正常，發出關閉指令...")

            # 發布關閉指令給兩台設備
            client.publish(DEHUMIDIFIER_TOPIC_1, "OFF", qos=1)
            client.publish(DEHUMIDIFIER_TOPIC_2, "OFF", qos=1)
```

```
            DEHUMIDIFIERS_ON = False # 標記狀態已關閉

    except ValueError:
        print(f"錯誤：無法解析消息為數字: {msg.payload.decode()}")
    except Exception as e:
        print(f"處理消息時發生意外錯誤: {e}")


# --- 主程式 ---
client = mqtt.Client(client_id="Automation_Controller_v1")

# 設定 TLS/SSL 加密
client.tls_set(ca_certs=CA_CERTS_PATH)

# 設定用戶名和密碼
client.username_pw_set(USERNAME, PASSWORD)

# 設定回調函數
client.on_connect = on_connect
client.on_message = on_message

# 開始連線
client.connect(MQTT_BROKER_HOST, MQTT_PORT, 60)

# 保持迴圈運行，持續監聽和處理網絡數據
client.loop_forever()
```

II. 實驗測試與結果驗證 (使用 4 個終端機)

我們使用 4 個並行的終端機來模擬整個自動化迴路，以證明邏輯正確。

1. 啟動所有監控端 (3 個終端機持續運行)

- 終端機 1 (控制器 / 邏輯大腦)： 執行 Python 程式。

  `python3 auto_controller.py`

```
(mqtt_venv) username@myserversname:~$ python3 auto_controller.py
/home/username/auto_controller.py:68: DeprecationWarning: Callback API version 1 is deprecated, update to latest version
  client = mqtt.Client(client_id="Automation_Controller_v1")
連線成功！開始監測溫度...
```

- 終端機 2 (除濕機 1 / 接收端)： 監聽對除濕機 1 的控制指令。

  `mosquitto_sub -h localhost -p 8883 --cafile /etc/mosquitto/certs/mosquitto.crt -t "control/#" -q 1 -v -u <username> -P <password>`

- 終端機 3 (溫度監看)： 監看所有發布的原始溫度數據 (非控制迴路必須，但方便觀察)。 `mosquitto_sub -h localhost -p 8883 --cafile /etc/mosquitto/certs/mosquitto.crt -t "orchid/+/temperature" -q 1 -v -u <username> -P <password>`

2. 執行測試數據發布 (使用第 4 個終端機)

- 情境 A：溫度過高 (觸發開啟)指令：

  `mosquitto_pub -h localhost -p 8883 --cafile ... -t "orchid/roomA/temperature" -m "32.5" -u <username> -P <password>`

```
username@myserversname:~$ mosquitto_pub -h localhost -p 8883 --cafile /etc/mosquitto/certs/mosquitto.crt -t "orchid/room
A/temperature" -m "25.0" -q 1 -u username -P [redacted]
username@myserversname:~$ mosquitto_pub -h localhost -p 8883 --cafile /etc/mosquitto/certs/mosquitto.crt -t "orchid/room
B/temperature" -m "32.5" -q 1 -u username -P [redacted]
```

  - 預期結果：

    - 控制器 (終端機 1) 打印：超過閾值！發出開啟指令...

```
連線成功！開始監測溫度...
[orchid/roomA/temperature] 收到溫度: 25.0°C
[orchid/roomB/temperature] 收到溫度: 32.5°C
🚨 溫度 32.5°C 超過閾值 30.0°C！發出開啟指令...
```

    - 除濕機 1/2 (終端機 2/3) 均收到：ON

```
username@myserversname:~$ mosquitto_sub -h localhost -p 8883 --cafile /etc/mosquitto/certs/mosquitto.crt -t "control/#"
-q 1 -v -u username -P [redacted]
control/dehumidifier1/power ON
control/dehumidifier2/power ON
```

- 情境 B：溫度恢復正常 (觸發關閉)

  - 指令：

  - `mosquitto_pub -h localhost -p 8883 --cafile ... -t "orchid/roomB/temperature" -m "28.0" -u <username> -P <password>`

  - 預期結果：

- 控制器 (終端機 1) 打印：恢復正常，發出關閉指令...

- 除濕機 1/2 (終端機 2/3) 均收到：OFF

- 終端機 3 (所有溫度監看過程)：