

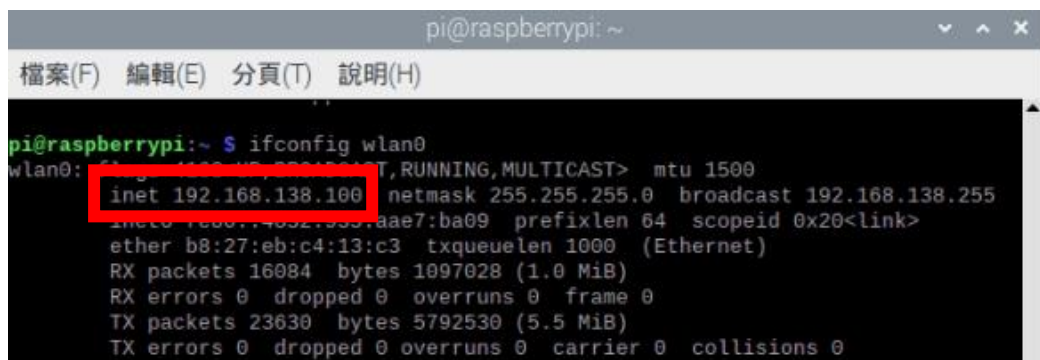
工業物聯網 1124hw

利彥儒 N96104103 廖沁旋 N96104080

(一) MQTT

MQTT 是一種 machine-to-machine (M2M) 的輕量級通訊協定，可以讓各種設備互相溝通，而其所需要的運算與傳輸頻寬很低，非常適合用於物聯網中的各種應用。

因為先前的實驗已經安裝好樹梅派了，因此我們只需要在 cmd 中輸入 `ifconfig wlan0` 此串指令，即能確認自己的 pi。而我們的是 192.168.132.100，如下圖：



```
pi@raspberrypi:~  
檔案(F) 編輯(E) 分頁(T) 說明(H)  
pi@raspberrypi:~ $ ifconfig wlan0  
wlan0: flags=4096<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
    inet 192.168.132.100 netmask 255.255.255.0 broadcast 192.168.132.255  
    inet6 fe80::4832:933:dae7:ba09 prefixlen 64 scopeid 0x20<link>  
    ether b8:27:eb:c4:13:c3 txqueuelen 1000 (Ethernet)  
    RX packets 16084 bytes 1097028 (1.0 MiB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 23630 bytes 5792530 (5.5 MiB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

由於 MQTT 的訊息全都透過稱為代理人 (broker) 的伺服器交流，所以我們在樹梅派中安裝了最常使用的 MQTT Broker: Mosquitto。

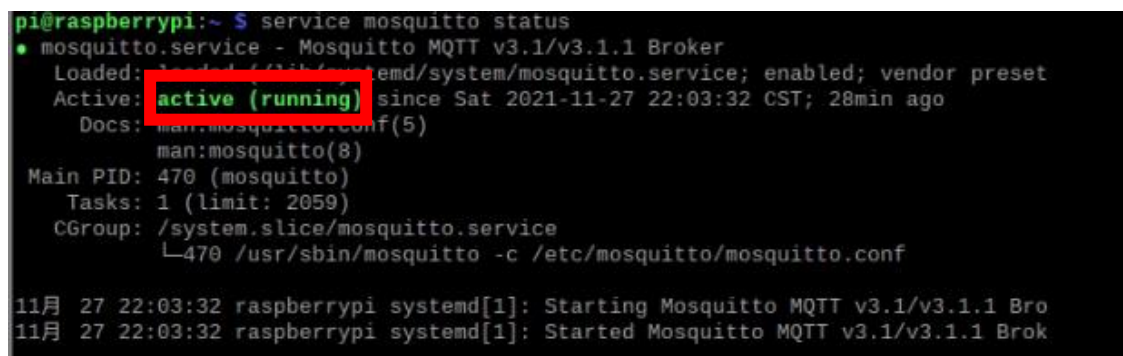
安裝的程式碼如下：

```
apt-get install mosquitto mosquitto-clients
```

正常來說，安裝完成後 mosquitto 服務會自動啟動，我們可以使用 `service` 指令檢查一下 mosquitto 服務的狀態：

```
service mosquitto status
```

而我們得到的結果如下：系統顯示 Mosquitto 正在 running，所以有安裝成功

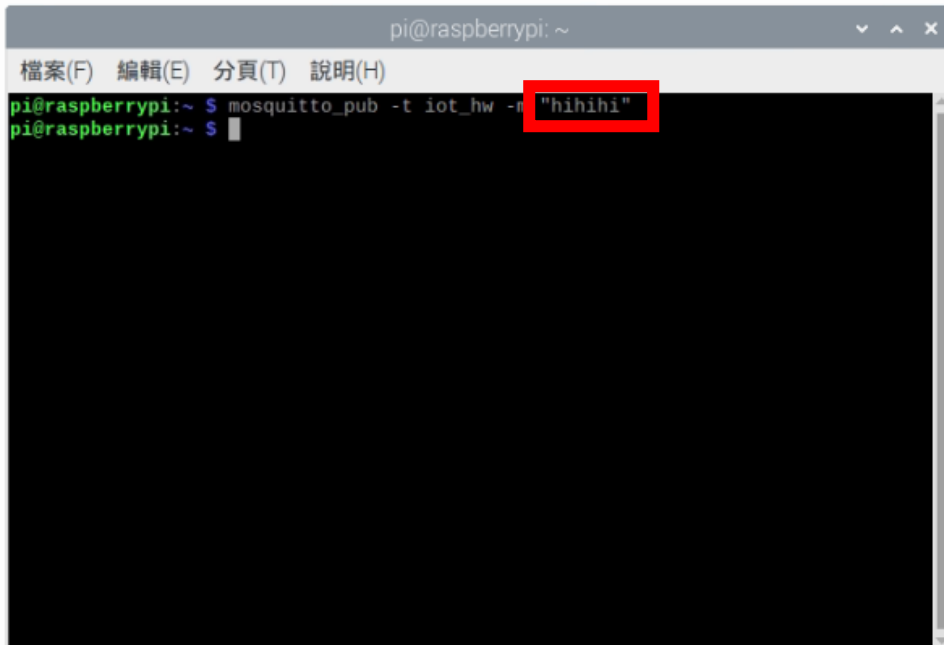


```
pi@raspberrypi:~ $ service mosquitto status  
● mosquitto.service - Mosquitto MQTT v3.1/v3.1.1 Broker  
   Loaded: loaded (/lib/systemd/system/mosquitto.service; enabled; vendor preset  
   Active: active (running) since Sat 2021-11-27 22:03:32 CST; 28min ago  
     Docs: man:mosquitto.conf(5)  
           man:mosquitto(8)  
   Main PID: 470 (mosquitto)  
     Tasks: 1 (limit: 2059)  
    CGroup: /system.slice/mosquitto.service  
            └─470 /usr/sbin/mosquitto -c /etc/mosquitto/mosquitto.conf  
  
11月 27 22:03:32 raspberrypi systemd[1]: Starting Mosquitto MQTT v3.1/v3.1.1 Bro  
11月 27 22:03:32 raspberrypi systemd[1]: Started Mosquitto MQTT v3.1/v3.1.1 Brok
```

接下來就可以試著用 cmd 一方當發布端、一方當接收端了。(要記得在不同視窗

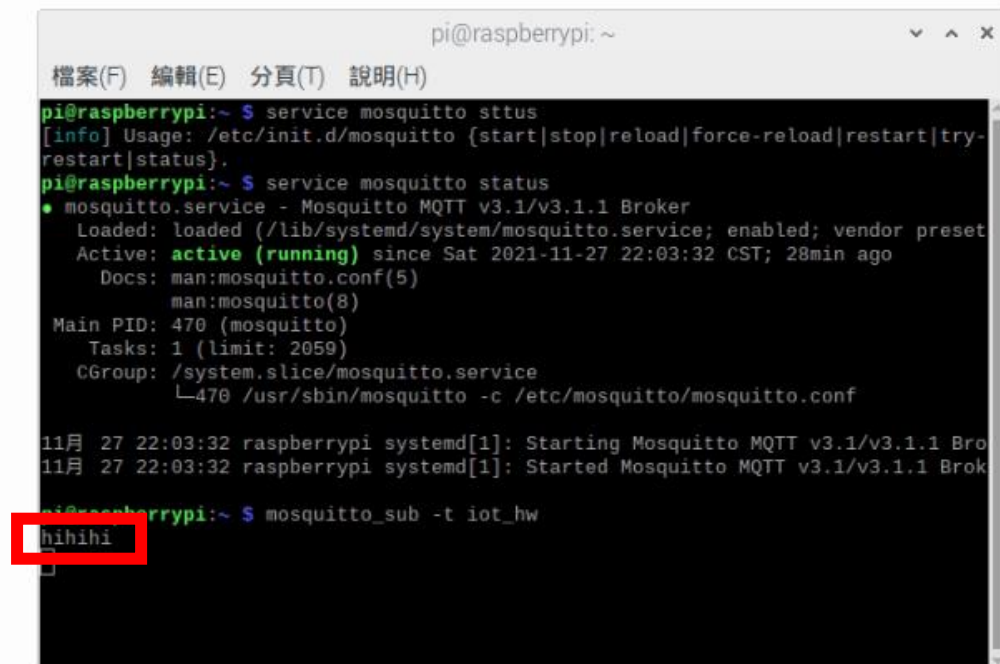
中的 cmd 來分別進行，並且要設定同一個名稱。)

作為發送端的視窗：發送了” hihihi”



```
pi@raspberrypi: ~  
檔案(F) 編輯(E) 分頁(T) 說明(H)  
pi@raspberrypi:~ $ mosquitto_pub -t iot_hw -m "hihihi"  
pi@raspberrypi:~ $
```

此時，接收端就會自動跳出 hihihi 了。



```
pi@raspberrypi: ~  
檔案(F) 編輯(E) 分頁(T) 說明(H)  
pi@raspberrypi:~ $ service mosquitto sttus  
[info] Usage: /etc/init.d/mosquitto {start|stop|reload|force-reload|restart|try-  
restart|status}.  
pi@raspberrypi:~ $ service mosquitto status  
● mosquitto.service - Mosquitto MQTT v3.1/v3.1.1 Broker  
   Loaded: loaded (/lib/systemd/system/mosquitto.service; enabled; vendor preset  
   Active: active (running) since Sat 2021-11-27 22:03:32 CST; 28min ago  
     Docs: man:mosquitto.conf(5)  
           man:mosquitto(8)  
   Main PID: 470 (mosquitto)  
     Tasks: 1 (limit: 2059)  
    CGroup: /system.slice/mosquitto.service  
            └─470 /usr/sbin/mosquitto -c /etc/mosquitto/mosquitto.conf  
  
11月 27 22:03:32 raspberrypi systemd[1]: Starting Mosquitto MQTT v3.1/v3.1.1 Bro  
11月 27 22:03:32 raspberrypi systemd[1]: Started Mosquitto MQTT v3.1/v3.1.1 Brok  
pi@raspberrypi:~ $ mosquitto_sub -t iot_hw  
hihihi
```

(二) Node-RED_SMTTP

Node-RED 是 IBM 以 Node.js 為基礎，開發出來的視覺化 IOT 開發工具，因為純粹透過流程圖的方式工作，所以不需要會 Node.js 也可以透過 Node-RED 完成許多後端才能做的事情。

SMTP 是在網際網路的不同郵件伺服器之間，進行電子郵件的交換與傳輸的通訊協定，SMTP 屬於即時送信與收信的通訊協定，傳送端與接收端的主機必須開機並連線，傳送端送出信件後，接收端立即收到信件。

首先，要先下載 Node-red，也是在 cmd 中進行、因為我沒有截到圖所以附上我們輸入的 code:

```
bash <(curl -sL https://raw.githubusercontent.com/node-red/linux-installers/master/deb/update-nodejs-and-nodered)
```

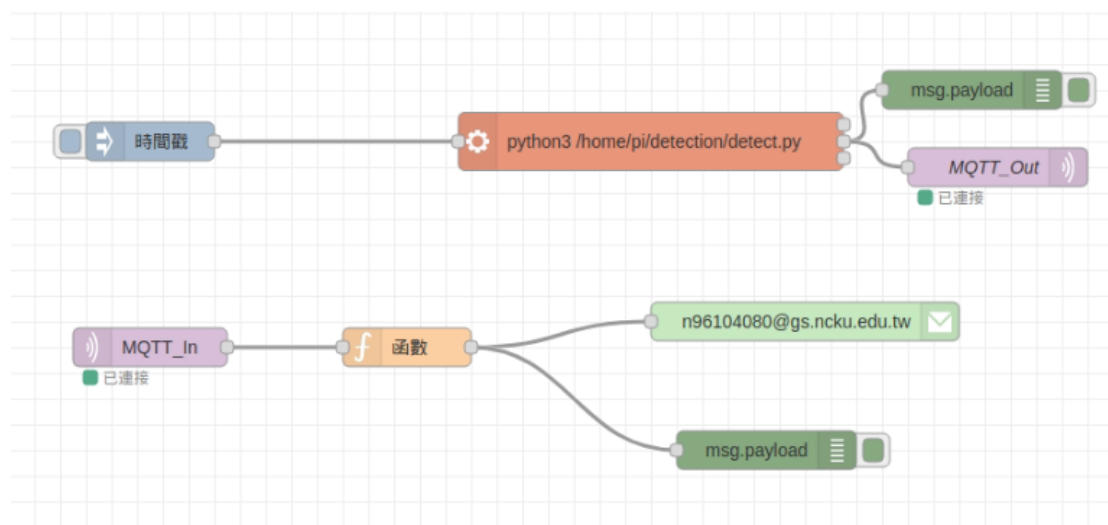
可以得到已經 running 的訊息(如下圖)，即可在瀏覽器中登入。

```
pi@raspberrypi:~ $ node-red
27 Nov 22:08:01 - [info]

Welcome to Node-RED
=====

27 Nov 22:08:01 - [info] Node-RED version: v1.3.4
27 Nov 22:08:01 - [info] Node.js version: v10.24.0
27 Nov 22:08:01 - [info] Linux 5.10.63-v7+ arm LE
27 Nov 22:08:10 - [info] Loading palette nodes
27 Nov 22:08:27 - [info] Settings file : /home/pi/.node-red/settings.js
27 Nov 22:08:27 - [info] Context store : 'default' [module=memory]
27 Nov 22:08:27 - [info] User directory : /home/pi/.node-red
27 Nov 22:08:27 - [warn] Projects disabled : editorTheme.projects.enabled=false
27 Nov 22:08:27 - [info] Flows file : /home/pi/.node-red/flows_raspberrypi.json
27 Nov 22:08:27 - [info] Starting flows
27 Nov 22:08:27 - [error] [mqtt in:75173a96.351324] missing broker configuration
27 Nov 22:08:27 - [error] [mqtt out:889db0d4.48a29] missing broker configuration
27 Nov 22:08:27 - [info] Server started
27 Nov 22:08:27 - [info] Server now running at http://127.0.0.1:1880/
```

在 Node-RED 中繪製我們需要用到的流程圖(要先安裝好 EMAIL Module)



編輯 MQTT broker 的資訊，輸入我們自己的 pi 的 ip 位址

刪除 取消 更新

屬性

名稱 MQTT

連接 安全 消息

服務端 192.168.138.100 埠 1883

☐ 使用 TLS

Protocol MQTT V3.1.1

使用者端ID 留白則自動隨機生成

Keepalive計時(秒) 60

Session ☒ 使用新的會話

設立運行

編輯 exec 節點

刪除 取消 完成

屬性

命令 python3 /home/pi/detection/detect.py

+ 追加 ☒ msg. payload

額外的輸入參數

輸出 當命令進行時 - spawn模式

超時 可選填 秒

名稱 名稱

設定 email 節點(我是使用我們其中一人的學校 email 作為節點)

編輯 email 節點

刪除 取消 完成

屬性

To n96104080@gs.ncku.edu.tw

Server smtp.gmail.com

Port 465 ☒ Use secure connection.

Userid n96104080

Password

TLS option ☒ Check server certificate is valid

名稱 名稱

再到 google 帳戶中開啟存取權(跟助教給的不太一樣所以找了一下)，這樣我們的 Node-RED 按下 DEPLOY 鍵後就不會出現錯誤訊息了！



然後就可以用 Python 去執行我們要的功能(我們更改了信件內容以及標題)

```
1 import smtplib
2 from email.mime.text import MIMEText
3
4 gmail_user= "n96104080@gs.ncku.edu.tw"
5 gmail_password = 
6
7 msg= MIMEText('''工業物聯網的成績怎麼都還沒出來呀！
8 希望每次作業都拿滿分或者高分！
9 這樣就太好啦!''')
10 msg["subject"] = "IOT_HW_GOGOGO"
11 msg["From"] = gmail_user
12 msg["To"] = "n96104080@gs.ncku.edu.tw"
13
14 server =smtplib.SMTP_SSL("smtp.gmail.com",465)
15 server.ehlo()
16 server.login(gmail_user, gmail_password)
17 server.send_message(msg)
18 server.quit()
19
20 print("Email sent!")
21
22
```

Shell

Email sent!

```
>>> %Run smtp.py
```

Email sent!

```
>>>
```

然後就能從 EMAIL 中看到我們的信啦！



試試寄給另外一個信箱帳號：

```
buzzer.py x smtp.py x
1 import smtplib
2 from email.mime.text import MIMEText
3
4 gmail_user= "n96104080@gs.ncku.edu.tw"
5 gmail_password = 
6
7 msg= MIMEText('工業物聯網的成績怎麼都還沒出來呀！
8 希望每次作業都拿滿分或者高分！
9 這樣就太好啦！')
10 msg["subject"] = "IOT_HW_GOGOGO"
11 msg["From"] = gmail_user
12 msg["To"] = "chin.h.liao@gmail.com"
13
14 server = smtplib.SMTP_SSL("smtp.gmail.com",465)
15 server.ehlo()
16 server.login(gmail_user, gmail_password)
17 server.send_message(msg)
18 server.quit()
19
20 print("Email sent!")
21
22
```

Shell

```
Email sent!
>>> %Run smtp.py
Email sent!
```

如果沒有開啟存取權的帳號，信件就會跑到垃圾郵件中：

