

通訊網路實驗

ØMQ

112學年度 第一學期

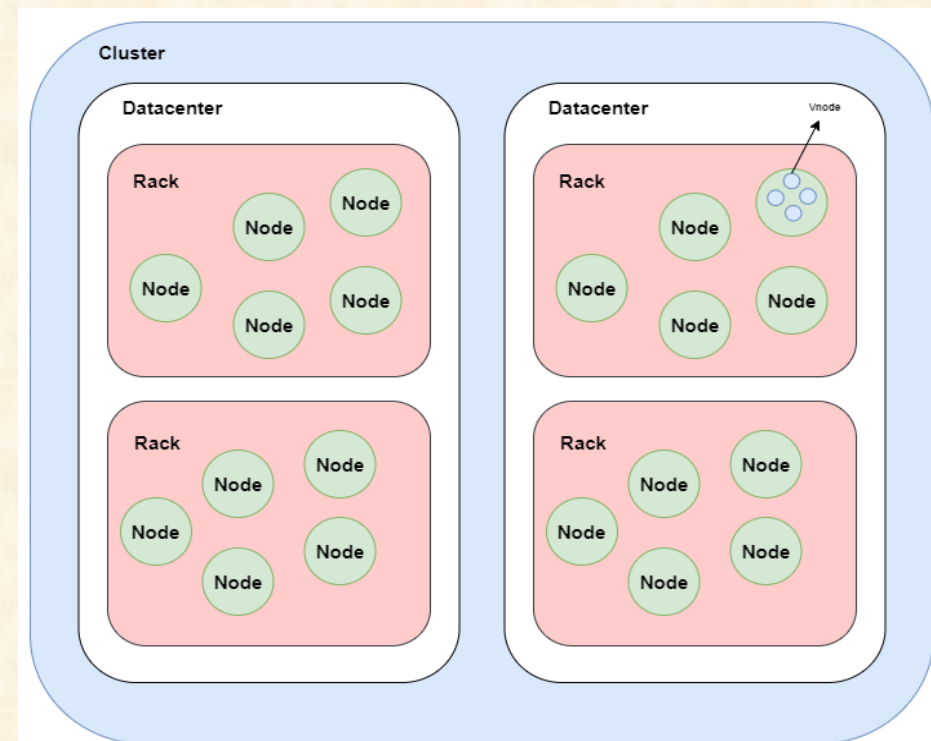
Dept. of Electrical and Computer Engineering (ECE)
National Yang Ming Chiao Tung University

ØMQ 介紹

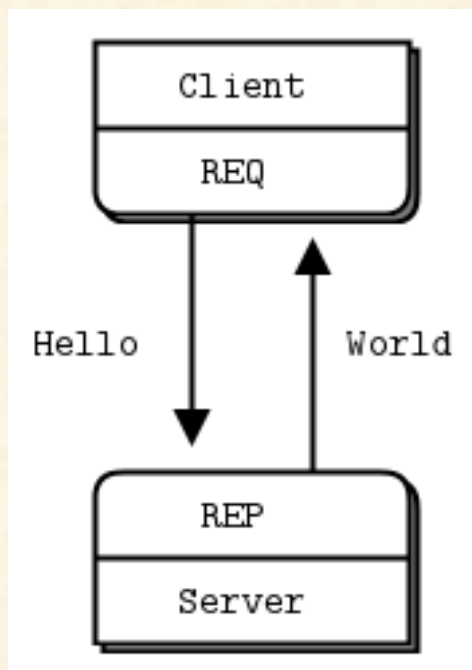
- ØMQ (也拼寫作ZeroMQ，0MQ 或 ZMQ)
- ZeroMQ 是由 iMatix 公司和大量貢獻者組成的社群共同開發
- 是一個非同步網路函式庫，提供 IPC、TCP 等通訊方式
- 就像是一個更多功能的 socket library，也可以說是 mailboxes with routing

ØMQ 的特點

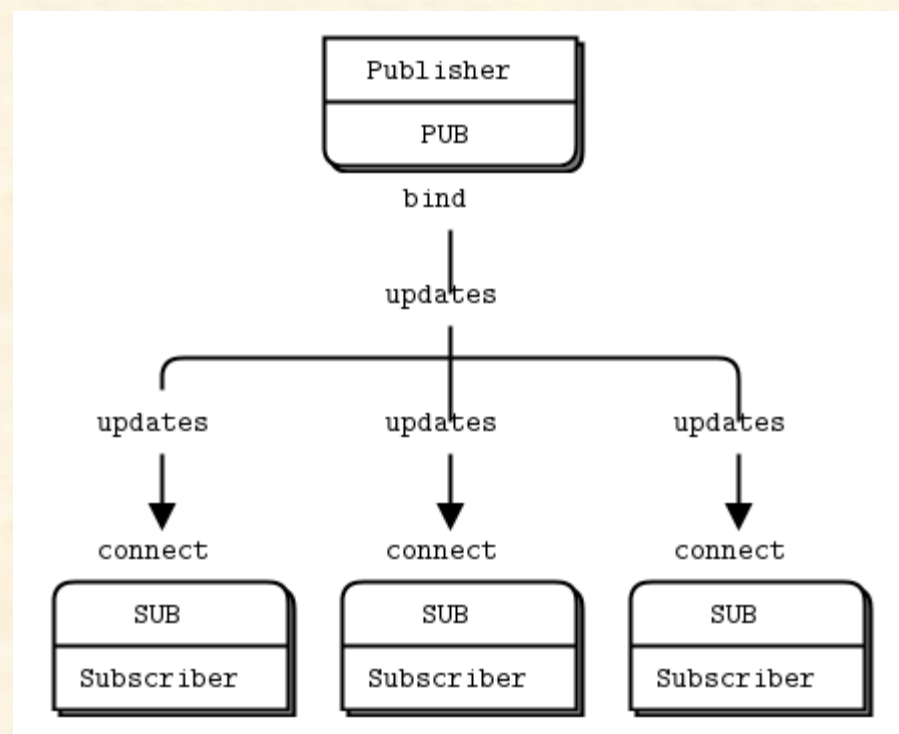
- 速度快，適用於大型集群和分散式計算
- 提供多種訊息傳遞機制，ex: in-process、TCP 等等
- 支援大部分流行的程式語言
 - ◆ Java
 - ◆ Python
 - ◆ C/C++



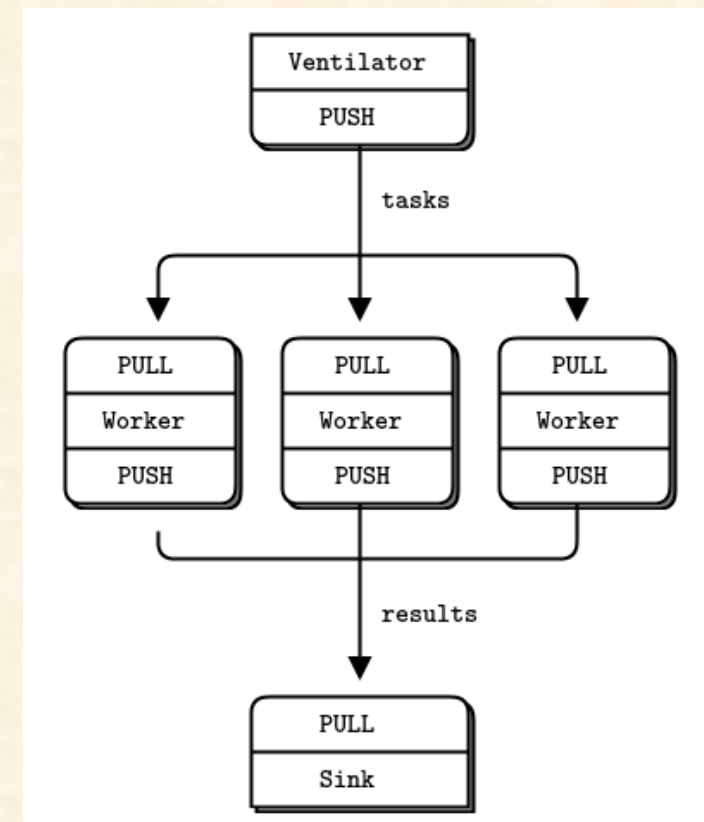
三種常見的 ØMQ 模式



1. Request – Reply 模式



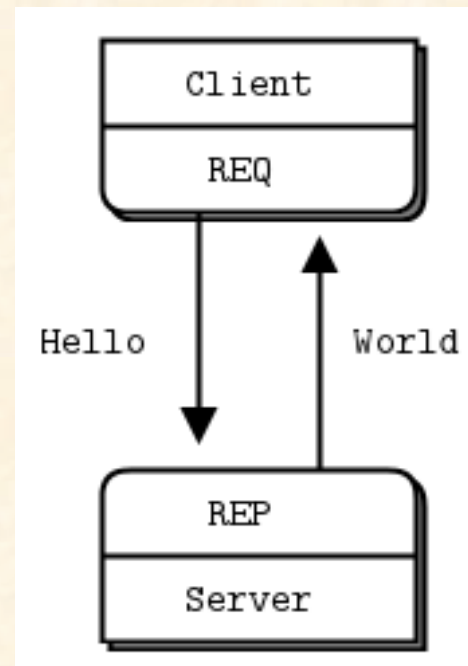
2. Publish – Subscribe 模式



3. Parallel – Pipeline 模式

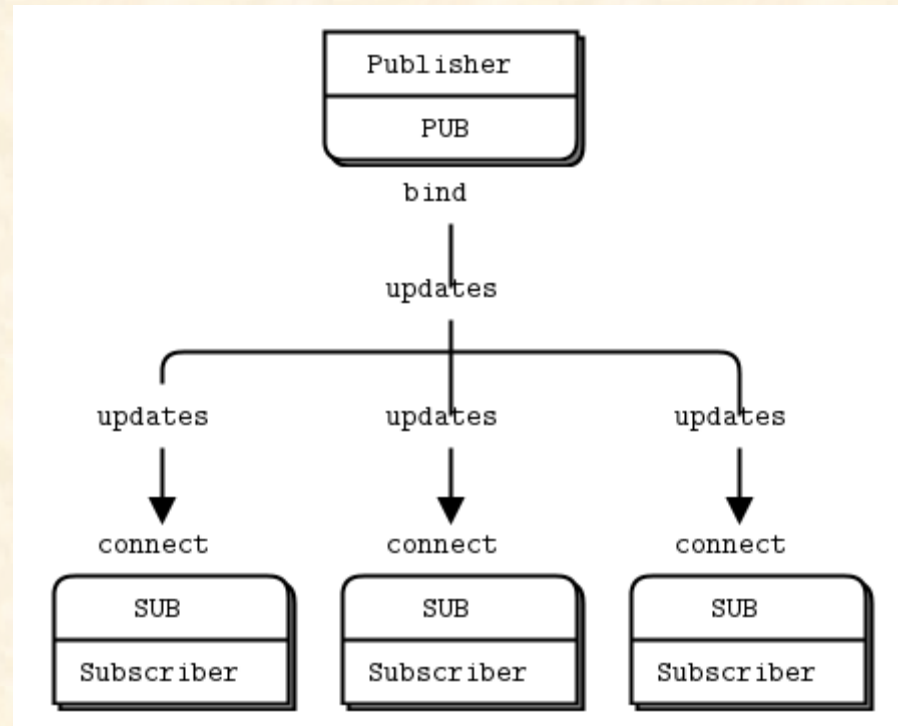
Request – Reply 模式

- 最基本的一種模式，Server 與 Client 互相收送資料
- Client 端在發出 Request 後，Server 端需要 Reply
- Round-robin (REQ)



Publish – Subscribe 模式

- 第二種常見的模式是單向的 data distribution
- Server 會持續發送資料給有訂閱相關主題的 Client

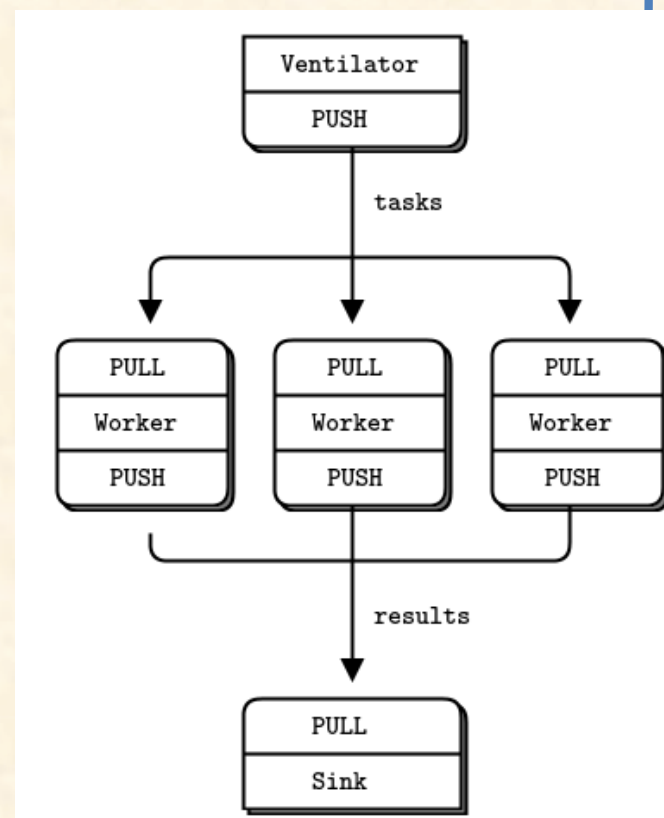


Publish – Subscribe 模式

- Subscriber 可以一次訂閱多個主題，且不會一直收到同一個 Publisher 的資料，而是會依次接收(fair-queued)
- 如果 Publisher 沒有連接任何的 Subscriber，則會丟棄所有資料

Parallel – Pipeline 模式

- 傳輸者將任務「平均」 PUSH 給多個 workers
 - 此方式稱為 **load balancing**
- 數個 workers 會處理任務並將結果由接收者 PULL 接收
 - 此方式稱為 **fair queueing**
- Worker 數量越多處理的時間越短



網路設定

- 請根據上一堂課 (lab 1) 的 wifi 設定教學，將 Pi 板與電腦連上同個 AP
 - SSID : bunexp
 - 密碼 : 7111177117

- 若使用桌機需插上無線網卡

ZMQ 常用 Function 介紹

□ 建立 context

```
context = zmq.Context()
```

□ 建立 socket

- zmq.SUB 為模式參數，有 REQ、REP、PUB...等

```
socket = context.socket(zmq.SUB)
```

□ socket.connect / socket.bind 用法跟一般的 socket 程式一樣

□ socket.setsockopt(zmq.SUBSCRIBE, topicfilter)

- topicfilter: 訂閱的主題
- 如果沒有設定的話就接收不到任何資料
- **topicfilter** 須為 **string** 型態變數

□ publish 方式

- socket.send_multipart([**topic**, **data**])
- socket.send("**topic data**...")

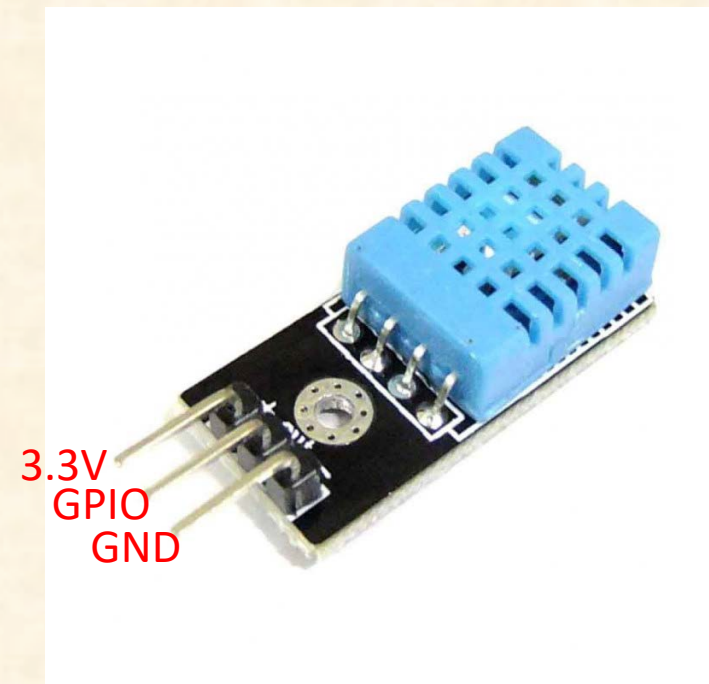
□ socket.send(), socket.recv()

- 傳輸/接收data
- socket.send(string_data)
- socket.send_multipart([string1, string2])
- list = socket.recv_multipart()

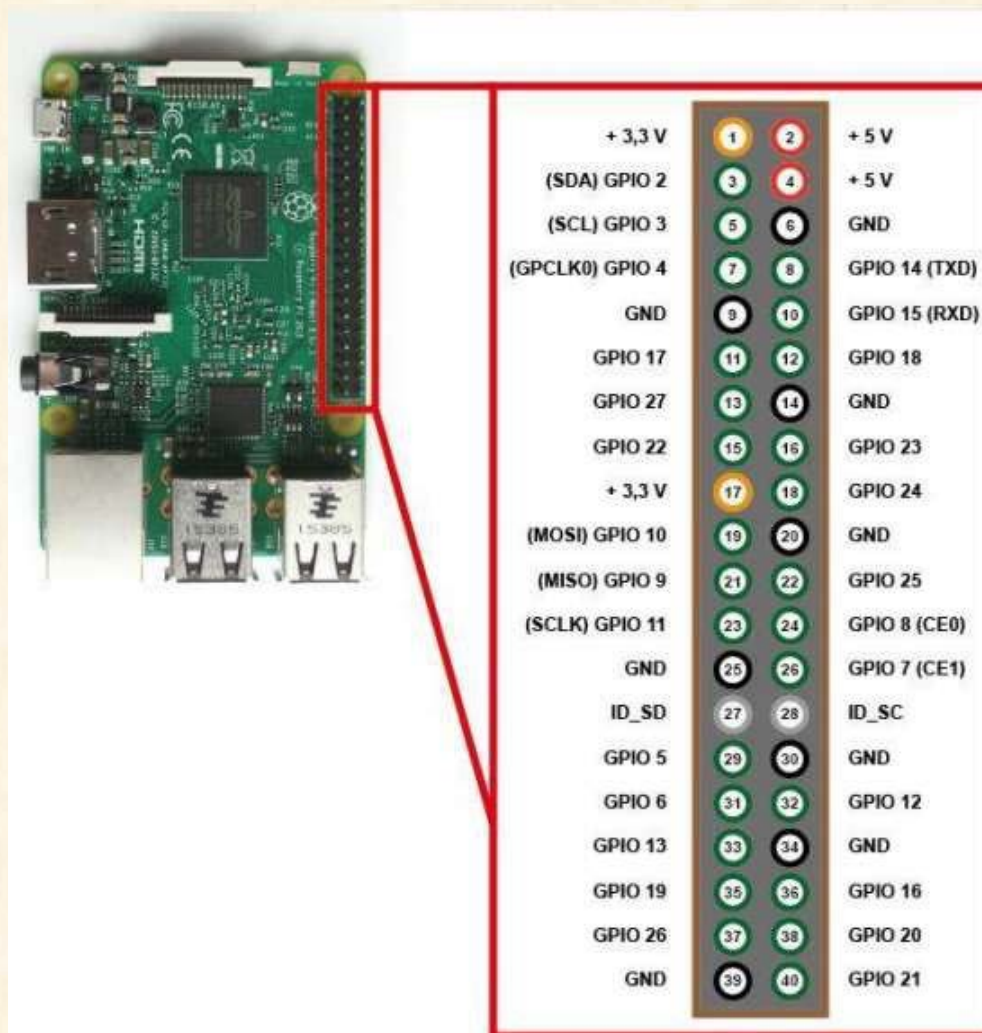
溫溼度感測器

□ DHT11 溫溼度感測器

- ◆ 溫度: $0 \sim 50^{\circ}\text{C}$, 誤差 $\pm 2^{\circ}\text{C}$
- ◆ 濕度: $20 \sim 90\%$, 誤差 $\pm 5\%$
- ◆ 使用三個腳位: Data , VCC , GND (out 、 + 、 -)
- ◆ Data 腳位統一連接到 RPi 板上的 GPIO4 (Pin 7)
- ◆ VCC 連接到 RPi 板上的 3.3 V 位置
- ◆ GND 則接地



腳位參考圖



下載本次實驗函式庫

□ DHT11

- ◆ git clone https://github.com/adafruit/Adafruit_Python_DHT.git
- ◆ cd Adafruit_Python_DHT
- ◆ sudo python setup.py install

□ GPIO

- ◆ sudo pip install rpi.gpio

□ sudo pip install pyzmq

□ 程式碼已放在E3

溫溼度感測器功能測試

- 執行函式庫提供的測試檔
 - ◆ `cd Adafruit_Python_DHT/examples`
 - ◆ `sudo ./AdafruitDHT.py 11 4`
 - 11 為 DHT11 (也有 22 的型號)
 - 4 為GPIO 4 (也就是Pin 7)

```
pi@raspberrypi ~ $ cd Adafruit_Python_DHT/examples/  
pi@raspberrypi ~/Adafruit_Python_DHT/examples $ sudo ./AdafruitDHT.py 11 4  
Temp=26.0* Humidity=37.0%
```

※若測試結果有任何錯誤或是無結果請先自行
檢查溫溼度計模組是否有接線錯誤，待確認後
再跟助教反應需更換材料或其他處置

Code 解釋

```
1  import Adafruit_DHT
2
3  # Setup DHT11
4  sensor_args = {'11' : Adafruit_DHT.DHT11,
5                 '22': Adafruit_DHT.DHT22,
6                 '2302': Adafruit_DHT.AM2302}
7
8  sensor = sensor_args['11']
9
10 # GPIO#, ex: GPIO4 = Pin7
11 gpio = 4
12
13 #Get humidity & temperature
14 humidity, temperature = Adafruit_DHT.read_retry(sensor, gpio)
15 humidity = str(humidity)
16 temperature = str(temperature)
```

本次實驗 Demo

- Q1: 使用助教提供的程式完成一個分散式加法計算的架構
 - 共要同時運行三個 servers 與一個 client(共四個Session，先執行servers最後執行client)
 - Lab2_1_client.py 共會執行 9 次迴圈
 - 每一次迴圈亂數產生 a 與b 兩個數值(介於 0 ~ 100)
 - Lab2_1_client.py 傳輸給 Lab2_1_server.py 計算，再由 Server 傳輸回來並顯示在 terminal上
 - 先執行三個 Lab2_1_server.py 檔，再執行 Lab2_1_client.py 檔
 - 每個 Lab2_1_server.py 都會被平均分配到 3 個 a, b 數值做計算
 - client、servers 之間傳遞 a、b 兩數字請用 `socket.send_multipart()`、`socket.recv_multipart()`, client、servers 之間傳遞運算結果請使用 `socket.send_string()`、`socket.recv()`
 - 請注意變數型態後再做運算或傳送

本次實驗 Demo

```

pi@raspberrypi:~ $ sudo python Lab2_1_server.py
Worker 2377 is running ...
Compute 27 + 76 and send response
Compute 21 + 3 and send response
Compute 92 + 96 and send response

pi@raspberrypi:~ $ sudo python Lab2_1_server.py
Worker 2362 is running ...
Compute 14 + 34 and send response
Compute 65 + 93 and send response
Compute 57 + 31 and send response

pi@raspberrypi:~ $ sudo python Lab2_1_server.py
Worker 2347 is running ...
Compute 26 + 74 and send response
Compute 6 + 94 and send response
Compute 42 + 84 and send response

```

1. 先執行3次server

2. 再執行1次client

```

pi@raspberrypi:~$ sudo python Lab2_1_client.py
Compute 27 + 76
= 103 (from worker 2377)
Compute 26 + 74
= 100 (from worker 2347)
Compute 14 + 34
= 48 (from worker 2362)
Compute 21 + 3
= 24 (from worker 2377)
Compute 6 + 94
= 100 (from worker 2347)
Compute 65 + 93
= 158 (from worker 2362)
Compute 92 + 96
= 188 (from worker 2377)
Compute 42 + 84
= 126 (from worker 2347)
Compute 57 + 31
= 88 (from worker 2362)

```


□ 如果出現下圖的 Error

- 1. 請重新開啟 SSH
- 2. 或將 Lab2_1_client.py、Lab2_1_server.py 中的 socket port 修該成其他數字

```
pi@raspberrypi:~/yen/kevin/ZMQ $ sudo python Lab2 1 client.py
Traceback (most recent call last):
  File "Lab2_1_client.py", line 8, in <module>
    socket.bind("tcp://*:7788")
  File "zmq/backend/cython/socket.pyx", line 550, in zmq.backend.cython.socket.Socket.bind
  File "zmq/backend/cython/checkrc.pxd", line 26, in zmq.backend.cython.checkrc._check_rc
zmq.error.ZMQError: Address already in use
```

本次實驗 Demo


- Q2: 使用助教提供的程式實現 Pub/Sub 模式訂閱來自助教的溫溼度數值
 - 根據助教提供的 `Lab2_2_client.py`及`Lab2_2_server.py`，並自行完成
 - `connect`及`bind`函式修改為自己server的 IP 位置：`192.168.xxx.xxx:5556`
(使用`ifconfig`指令查詢IP)
 - 同時訂閱且接收溫度跟濕度
 - 先執行server，再執行client
 - server端的輸出格式為
 - `socket.send("temp = %d" % (temp))`
 - `socket.send("humidity = %d" % (humidity))`
 - 溫度和濕度各收 10 筆資料後做平均並將結果印出
 - 請顯示每一次接收的溫度和濕度數據

本次實驗 Demo

```
pi@raspberrypi:~ $ sudo python Lab2_2.py
temp = 129
humidity = 44
temp = 89
humidity = 57
temp = -74
humidity = 22
temp = 41
humidity = 41
temp = -34
humidity = 46
temp = 43
humidity = 17
temp = 93
humidity = 46
temp = 108
humidity = 26
temp = -20
humidity = 22
temp = 11
humidity = 42
average temp = 38.6
average humidity = 36.3
```

□ string.split(token)

```
1 msg = "I love NYCU"  
2 words = msg.split()  
3  
4 for word in words:  
5     print(word)
```



```
I  
love  
NYCU
```

本次結報內容

- 1. 附上本次實驗 Q1 4 個 terminals 的結果圖，Q2 溫溼度接收的結果圖
- 2. 考慮本次 Q1 和 Q2 的實驗流程下來，請問 ZMQ 的 client 與 server 間的 connect、bind 順序不同的話會對實驗內容有影響嗎？，請說明原因。
- 3. 列舉 ZMQ 的三種常見模式的優缺點。
- 4. 試想 ZMQ 的三種常見模式 Request - Reply、Publish - Subscribe、Parallel - Pipeline 分別有哪些應用？(愈詳細且創新分數越高)
- 5. 本次實驗心得，你學到了什麼東西？

評分標準 & 注意事項

- 出席 30 %
- Demo 30 %
- 結報 40 %
- 請繳交 .pdf 檔，檔名為 學號_姓名_Labx.pdf
- 結報遲交一天，分數扣10%，以此類推

Reference

- <https://zh.wikipedia.org/wiki/%C3%98MQ>
- <https://cloud.tencent.com/developer/article/1129302>
- https://blog.csdn.net/mydear_11000/article/details/75218543
- <https://blog.maxkit.com.tw/2019/09/zeromq.html>
- [ZeroMQ 教學 | Medium](#)