MQTT

MQTT 介紹

IBM 1999年開發

隨物聯網興起而興起

解決問題：

1. 服務器要實現成千上萬客戶端接入
2. 單次數據量小，但不能出錯
3. 適應高延遲、斷網等通訊不可靠風險
4. 根據數據的重要程度和特性設置不同等級的服務

特性

1. 針對移動中端設備，基於TCP/IP， 發佈/訂閱 的消息協議
2. 可以連接大量遠程傳感器和控制設備
3. 可以保持長連接，具有一定的實時性
4. 雙向實時通信
5. 適合需要實時控制的場合，尤其適合執行器
6. 保持長連接心跳，耗電
7. 低功耗場合不適合MQTT
8. MQTT建立在TCP基礎上，對設備要求比UDP高
9. 發佈/訂閱模式，提供一對多的消息發佈，減除應用程序耦合
10. 三種消息發佈服務質量（QOS0、QOS1、QOS2）
    - 至多一次，完全依賴TCP/IP網絡，會發生消息丟失和重複，適用於對採集數據要求不高的場景

    - 至少一次，確保消息到達，消息可能會重複
    - 只有一次，確保消息只到達一次，消息重複或丟失都不允許，適用於計費系統等要求嚴格的場景
11. 開銷小
12. 使用Last Will 和 Testament特性通知有關各方客戶端異常中斷的機制
13. 允許用戶動態創建主題，零運維成本
14. 把底帶寬，不穩定，高延遲的網絡因素考慮在內
15. 假設數據不可知，不強求傳輸數據的類型與格式，保持靈活性
16. 官網： http://mqtt.org

架構

包含三角色：服務器、發佈者、訂閱者

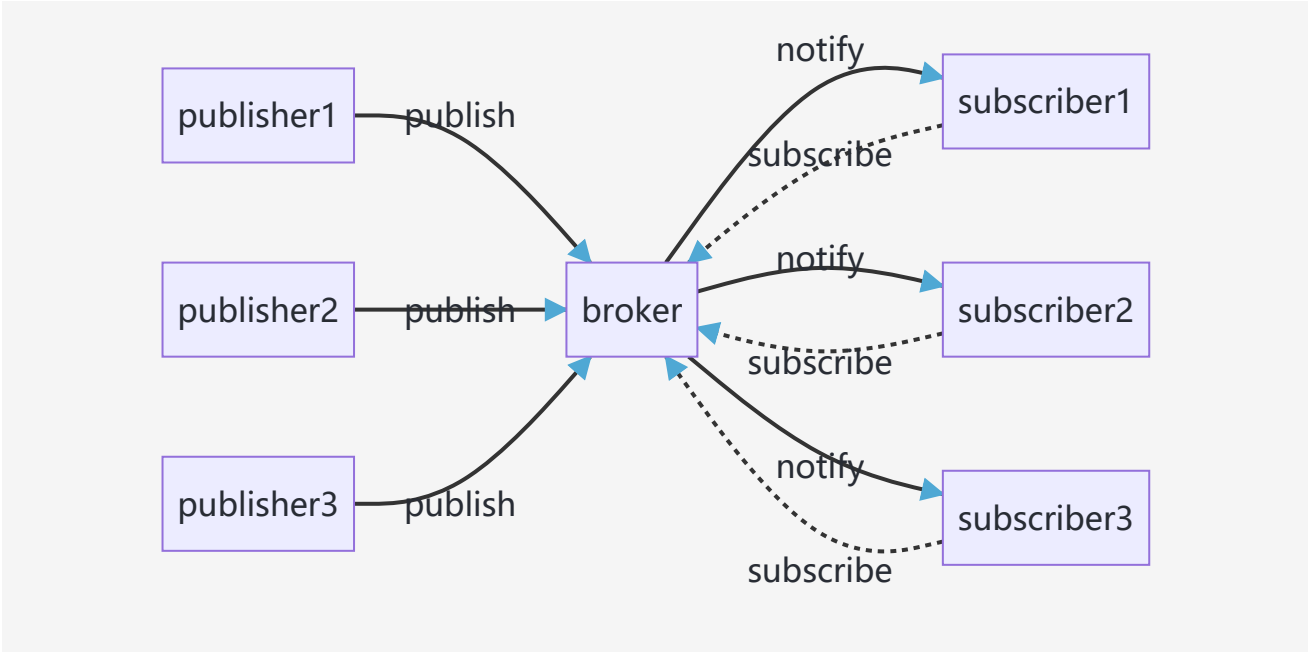
MQTT服務器或者broker是協議的核心

服務器職責：

1. 轉發消息
2. 收集數據，對數據進行處理和存儲

發佈者： 發佈消息

訂閱者：訂閱消息



代替了傳統的服務器/客戶端 模型，可以實現如下解耦:

1. 空間解耦， 雙方不需要知道對方
2. 時間解耦， 雙方不需要同時在線，即支持離線消息
3. 同步解耦， 異步通訊，無需停止任何處理

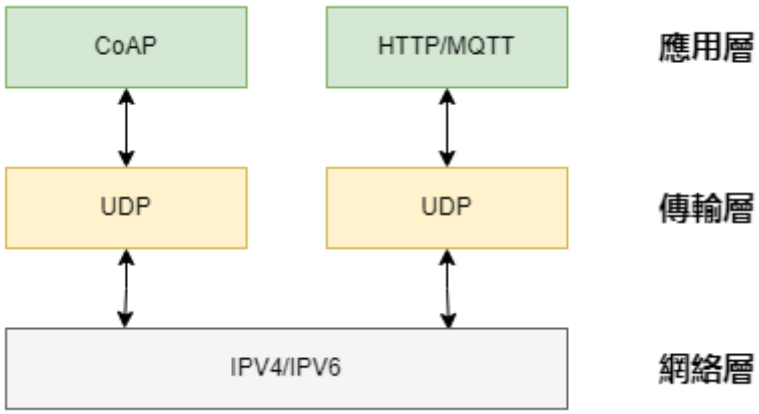
主題

1. MQTT通過主題對消息進行分類
2. 主題本質是UTF-8字符串
3. 通過 '/' 表示多個層級關係
4. 不需要創建，直接使用
5. 通過通配符進行過濾
    - '+' 過濾一個層級
    - '*' 過濾任意級別的層級
        - building-b/floor-5: 表示B樓5層的設備
        - +/floor-5: 表示任一樓的5層的設備
        - building-b/*: 表示B樓所有設備
        -

MQTT協議

協議層

CoAP是另外一種物聯網協議



報文格式

Control Header	Remaining Length	Variable Header	Payload
1 byte	1-4 bytes	0-x bytes	0-y bytes

1. 固定報頭

Figure 2.2 - Fixed header format

Bit	7	6	5	4	3	2	1	0
byte 1	MQTT Control Packet type				Flags specific to each MQTT Control Packet type			
byte 2...	Remaining Length							

Table 2.1 - Control packet types

Name	Value	Direction of flow	Description
Reserved	0	Forbidden	Reserved
CONNECT	1	Client to Server	Client request to connect to Server
CONNACK	2	Server to Client	Connect acknowledgment
PUBLISH	3	Client to Server or Server to Client	Publish message
PUBACK	4	Client to Server or Server to Client	Publish acknowledgment
PUBREC	5	Client to Server or Server to Client	Publish received (assured delivery part 1)
PUBREL	6	Client to Server or Server to Client	Publish release (assured delivery part 2)
PUBCOMP	7	Client to Server or Server to Client	Publish complete (assured delivery part 3)
SUBSCRIBE	8	Client to Server	Client subscribe request
SUBACK	9	Server to Client	Subscribe acknowledgment
UNSUBSCRIBE	10	Client to Server	Unsubscribe request
UNSUBACK	11	Server to Client	Unsubscribe acknowledgment
PINGREQ	12	Client to Server	PING request
PINGRESP	13	Server to Client	PING response
DISCONNECT	14	Client to Server	Client is disconnecting
Reserved	15	Forbidden	Reserved

Table 2.2 - Flag Bits

Control Packet	Fixed header flags	Bit 3	Bit 2	Bit 1	Bit 0
CONNECT	Reserved	0	0	0	0
CONNACK	Reserved	0	0	0	0
PUBLISH	Used in MQTT 3.1.1	DUP[1]	QoS[2]	QoS[2]	RETAIN[3]
PUBACK	Reserved	0	0	0	0
PUBREC	Reserved	0	0	0	0
PUBREL	Reserved	0	0	1	0
PUBCOMP	Reserved	0	0	0	0
SUBSCRIBE	Reserved	0	0	1	0
SUBACK	Reserved	0	0	0	0
UNSUBSCRIBE	Reserved	0	0	1	0
UNSUBACK	Reserved	0	0	0	0
PINGREQ	Reserved	0	0	0	0
PINGRESP	Reserved	0	0	0	0
DISCONNECT	Reserved	0	0	0	0

DUP[1] = Duplicate delivery of a PUBLISH Control Packet

QoS[2] = PUBLISH Quality of Service

RETAIN[3] = PUBLISH Retain flag

See Section 3.3.1 for a description of the DUP, QoS, and RETAIN flags in the PUBLISH Control Packet.

1. 剩餘數據長度

剩餘數據長度最長四個字節，也就是最大能傳輸256M的內容

```
// encoding
do

             encodedByte = X MOD 128

             X = X DIV 128

            // if there are more data to encode, set the
top bit of this byte

```

```
            if ( X > 0 )

                encodedByte = encodedByte OR 128

            endif

            'output' encodedByte

 while ( X > 0 )
```

```
//decoding
multiplier = 1

value = 0

do

            encodedByte = 'next byte from stream'

            value += (encodedByte AND 127) * multiplier

            if (multiplier > 128*128*128)

              throw Error(Malformed Remaining Length)

            multiplier *= 128

while ((encodedByte AND 128) ≠ 0)
```

2. 可變長度報頭
    - 報文標識
    - 協議名稱

- 協議級別
- 連接標誌
- 保活時間：客戶端在發送兩個報文之間最大間隔，如果沒有報文發送，則需要發送PINGREQ
- 連接標識

Package Identifier used to identify packages belonging to same 'session'

Figure 2.3 - Packet Identifier bytes

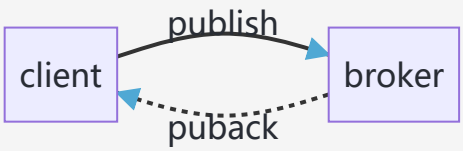
Bit	7	6	5	4	3	2	1	0
byte 1	Packet Identifier MSB							
byte 2	Packet Identifier LSB							

3. 有效數據載荷 有控制頭和可變頭參數決定

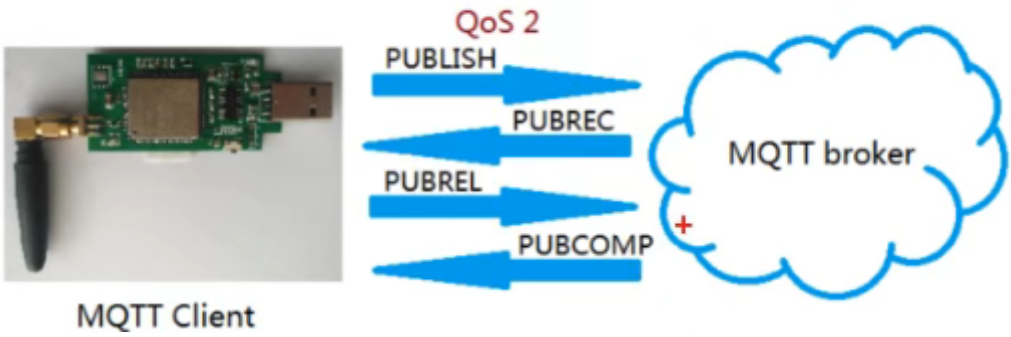
通訊過程

QoS0

QoS1



QoS2



MQTT 應用

基本用法

mosquitto 是一個高質量輕量級開源MQTT broker， 目前支持MQTT3.1和3.1.1協議，同時提供一個c語言動態連接庫

mosquitto 由MQTT協議創始人之一的Andy Stanford-Clark參與開發

https://mosquitto.org

```
#update
apt-add-repository ppa:mosquitto-dev/mosquitto-ppa
apt-get update

```

```
#install server
apt-get install mosquitto

#install client
apt-get install mosquitto-clients

#start broker
mosquitto -c /etc/mosquitto/mosquitto.conf -p 1883

#subscribe
mosquitto_sub -t "tmp"
#or
mosquitto_sub -t "tmp" -m "hello"

# show debug info
mosquitto_sub -t "tmp" -m "hello" -d

```

非匿名登錄配置

修改配置文件 /etc/mosquitto/mosquitto.conf

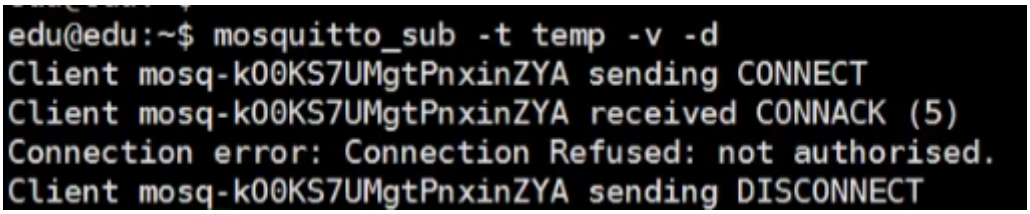
```
allow_anonymous false
password_file /etc/mosquitto/passwd.conf  //accounts file

```

服務器端創建用戶

```
sudo mosquitto_passwd -c /etc/mosquitto/passwd.conf tom //hide passwd
sudo mosquitto_passwd -b /etc/mosquitto/passwd.conf tom 111111 //visible passwd
```

無授權用戶訪問失敗



授權用戶訪問

```
mosquitto_sub -t temp -v -d -u tom -P 111111 //-u username, -P passwd
```

安全通訊

應用層：用過用戶名密碼驗證客戶端

傳輸層： 通過TLS加密，既可以實現身份驗證，有可以保證數據安全

網絡層： 採用VPN專線

安裝openssl

```
sudo apt-get install openssl
sudo apt-get install libssl-dev
```

生成證書和密鑰

```

```

mosquitto.conf中添加ssl信息

```
cafile /xx/ca.crt
certfile /xx/server.crt
keyfile /xx/server.key
```

客戶端訪問

```
mosquitto_sub -t temp -u tom -P 111111 --cafile /xx/ca.crt
```

雙向驗證

```
# mosquitto.conf
require_certificate true
use_identity_as_username true

mosquitto_sub -t temp -u tom -P 111111 --cafile /xx/ca.crt --cert
/xx/client.crt
--key /xx/client.key
```

mosquitto 移植

mosquitto 應用編程

其他資源

//其他服務器

http://www.steves-internet-guide.com/mqtt-hosting-brokers-and-servers/

//客戶端

https://mosquitto.org/download/

https://github.com/mqtt/mqtt.org/wiki/libraries

//hivemq，java embedded

https://github.com/hivemq/hivemq-community-edition