MQTTMonitor: monitoraggio di broker MQTT

Samuele Intaschi

Gestione di Reti 2018/2019

1 Introduzione

MQTT (Message Queue Telemetry Transport) è un protocollo di messaggistica di tipo publish-subscribe che opera sopra TCP/IP e viene utilizzato per la sua leggerezza principalmente su sistemi non molto potenti o quando la banda è limitata. Per lo smistamento dei messaggi ai destinatari ha bisogno di un message broker, i client infatti si iscrivono a dei topic e ricevono i messaggi pubblicati su questi, oppure possono pubblicare qualcosa su di essi, spesso vengono usate publish anche per impartire comandi a dispositivi remoti, ad esempio in ambito domotica.

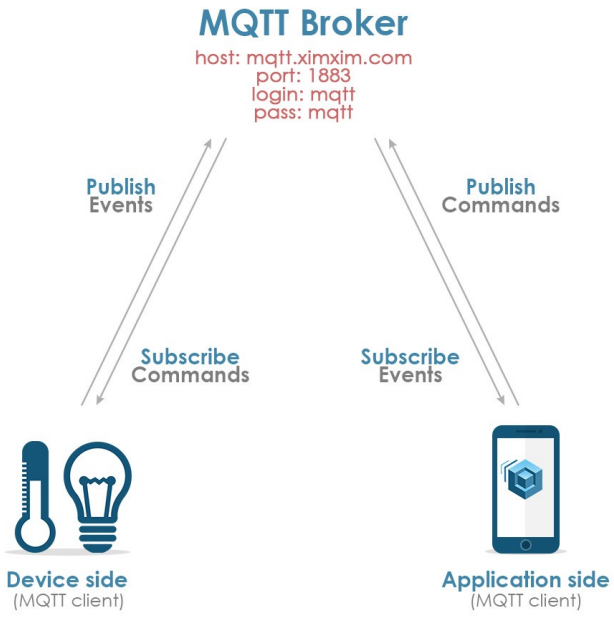

Figura 1: Schema di funzionamento del protocollo MQTT

Molti di questi broker implementano i cosiddetti *"SYS-topics"*, topic usati per comunicare ai client che vi si iscrivono informazioni sul broker stesso e sul suo stato, che sono di sola lettura.
Il broker pubblica periodicamente valori per questi argomenti, di default ogni 60 secondi, i client MQTT ricevono un messaggio appena si iscrivono e poi ogni volta che un valore viene pubblicato dal broker.
La sicurezza per questo protocollo è del tutto opzionale, infatti possono essere richiesti un username e una password al momento della connessione ma non viene fatto dalla maggior parte dei broker e, quando fatto, quasi sempre le cre-

denziali viaggiano in chiaro nel pacchetto di connessione.
MQTTMonitor è un programma che monitora lo stato e il funzionamento di un broker MQTT utilizzando i *"SYS-topics"* e inserendo i valori relativi a questi argomenti in un database per serie temporali, al fine di rilevare comportamenti "strani" sia da parte del server, sia da parte degli utenti iscritti che usufruiscono del servizio, con il limite del periodo usato dal broker per pubblicare le informazioni, che quindi non consente di notare tempestivamente la presenza di un problema.

2 Funzionamento

La creazione del client MQTT, la connessione ad esso e la ricezione dei messaggi vengono effettuate tramite le funzioni della libreria Eclipse Paho specifica per il linguaggio C, mentre la creazione e la distruzione del time series database vengono effettuate tramite l'interfaccia API di InfluxDB, così come le query per inserire i valori nel database. Per interagire con l'interfaccia API ho usato la libreria *libcurl*.
InfluxDB è un database per serie temporali gestibile con un linguaggio molto simile a SQL, che utilizza come chiave primaria il timestamp del momento in cui viene inserito un dato, in questo caso un valore numerico, e quindi consente di inserire solo il valore numerico e formare automaticamente una coppia chiave-valore (timestamp di raccolta del dato - valore numerico). Queste coppie poi vengono viste come una serie di punti ed è possibile visualizzarle su un grafico come serie temporale.
Il programma lavora con un solo thread ed inizia installando un gestore per il segnale SIGINT, unico modo per chiuderlo correttamente.
Subito dopo esegue il parsing del file *"mqttmonitor.conf"*, in cui l'utente ha inserito le informazioni che consentono al programma di funzionare.
A questo punto ha tutte le informazioni di cui necessita e quindi genera un client MQTT, si connette ad esso e crea un time series database.
Qui inizia l'elaborazione dei messaggi ricevuti: il programma cicla fino a che non viene ricevuto il segnale SIGINT, ogni volta che riceve un messaggio prende il valore, crea una query per il database ed esegue una richiesta HTTP di tipo POST per inserire questo valore. Se durante l'attesa di un messaggio il client si disconnette, il programma prova subito a riconnettersi.
Quando viene ricevuto il segnale SIGINT viene settata una variabile che provoca la terminazione del ciclo di attesa dei messaggi entro tre secondi (il tempo massimo di attesa di un messaggio) e inizia la fase terminale: viene distrutto il database creato all'inizio tramite apposita query, viene disconnesso e distrutto il client MQTT e viene liberata tutta la memoria occupata dalle strutture dati al momento della chiusura.
Mentre il programma è in esecuzione possono essere visualizzati grafici tramite *Chronograf*, uno strumento di InfluxData, con il quale possono essere create anche delle dashboard.

3 Utilizzo

Il programma, per essere compilato ed eseguito, ha bisogno di un'installazione preliminare della libreria *libcurl* e del pacchetto *Eclipse Paho* per C, che fornisce le funzioni per creare un client MQTT.
Entrambi hanno bisogno delle librerie del pacchetto *libssl-dev*, che si installa eseguendo in sequenza i comandi[1]:

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

Per installare *Eclipse Paho C* invece è necessario scaricare da github la repository specifica con il comando:

```
git clone https://github.com/eclipse/paho.mqtt.c.git
```

Poi entrare nella cartella con:

```
cd paho.mqtt.c
```

E infine installare il contenuto con:

```
make
sudo make install
```

A questo punto è possibile installare anche *libcurl* con il comando:

```
sudo apt-get install libcurl4-openssl-dev
```

Il programma non riceve parametri in input, ma li recupera dal file di configurazione *"mqttmonitor.conf"*, che l'utente è quindi invitato a modificare con le proprie preferenze, cioè l'indirizzo del server MQTT da monitorare, la quality of service (il livello di affidabilità richiesto per la ricezione dei messaggi), l'indirizzo dell'host su cui è ospitato il database InfluxDB, l'identificativo del client MQTT e i topic da monitorare.
Per testare il programma senza avere un host che ospita un database, si può usare *localhost* ospitando InfluxDB sul proprio PC, per l'installazione del quale rimando alla guida ufficiale.
Finito di installare queste librerie è necessario tornare nella cartella che contiene i file sorgente di *mqttmonitor* ed eseguire il comando:

```
make
```

Fatto questo il programma può essere eseguito da terminale con il comando:

```
./mqttmonitor
```

[1] Tutta la seguente procedura di installazione è stata testata sul sistema operativo *Lubuntu 19.10*, su altre distribuzioni di Linux potrebbe non essere funzionante.

Sul terminale verranno visualizzati inizialmente i topic a cui il programma si sta iscrivendo e poi appariranno di volta i valori ricevuti collegati ai topic a cui si riferiscono.
In questo modo tali valori non sono molto utili, ma utilizzando servizi come *Chronograf* (si veda la guida per l'installazione e la connessione al database) o *Grafana* e connettendoli al database InfluxDB possono essere visualizzati su grafici, come mostra la figura sottostante.

Figura 2: Un esempio di grafici, ottenuti monitorando il servizio di test *mqtt.eclipse.org:1883* su quattro valori qualunque (in particolare numero di clients connessi e non connessi e numero messaggi ricevuti e inviati dal broker in un minuto) visualizzati in una dashboard mediante *Chronograf*.

Questo tipo di visualizzazione consente all'utente di accorgersi subito di eventuali comportamenti non previsti da parte sia degli utenti che del server o in generale di avere una visione più chiara di come sta funzionando il sistema.
Ad esempio possono essere osservati picchi dei valori monitorati in alcuni momenti della giornata, oppure un numero di utenti connessi superiore a quelli che dovrebbero usare il sistema oppure ancora capire se il sistema si è riavviato.
Quando desidera interrompere il processo l'utente deve inviare un segnale di tipo SIGINT e il modo più semplice per farlo è premere i tasti Ctrl+C, fatto questo verrà disconnesso il client MQTT, distrutto il database e liberata la memoria occupata dalle strutture dati, terminando correttamente il programma.
La configurazione predefinita è fornita per testare il programma ed è funzionante a patto di aver installato e in esecuzione sulla porta 8086 del proprio PC un' istanza di InfluxDB. Utilizza il broker MQTT di test *mqtt.eclipse.org:1883* monitorando quattro topic: clients connessi, non connessi, messaggi ricevuti in un minuto e messaggi inviati in un minuto.
Il risultato finale utilizzando *Chronograf* per la visualizzazione dei valori è quello mostrato in figura 2.