Paweł Kraszewski <root+LinuX-LAB@linuxpedia.pl>

Krótko o protokole MQTT

- Skrót MQTT rozwija się jako **Message Queuing Telemetry Transport**
- Opracowany przez **IBM** w 1999, obecnie jest standardem ISO/IEC PRF 20922, zarządzanym przez konsorcjuim **OASIS** (to od m.in. formatu OpenDocument OpenOffice'a)
- Strona domowa to **`http://mqtt.org/`**

Topologia MQTT

- Protokół typu **klient-serwer** (zasadniczo gwiazda)
- Serwer nazywany jest **brokerem komunikatów**
- Klienci mogą **publikować** komunikaty w hierarchii tematów i/lub **subskrybować** wybrane tematy.
- Komunikaty są blobami o rozmiarze do ~256MB
- Tematy przypominają ścieżki w systemie plików (struktura drzewa)

Założenia projektowe MQTT

- MQTT transportowany jest przez **TCP**, jednak występuje wariant MQTT-SN do pracy w sieciach nie-IP, np. **Zigbee** albo **Bluetooth**.
- Ramki komunikatów mają **szczątkowy narzut** protokołu (bardzo efektywny binarnie, łatwy do zaprogramowania w IoT)
- Protokołu nie interesuje **zawartość komunikatu** (binarka, JSON, XML, plain text)

Dostarczanie MQTT

- Komunikaty mogą mieć flagę **retain** (zachowaj), czekają wtedy na odbiorcę w brokerze
- Komunikaty mają poziom obsługi **QoS**:
  - 0 – dostarcz nie więcej niż raz (być może wcale)
  - 1 – dostarcz nie mniej niż raz (być może wielokrotnie)
  - 2 – dostarcz dokładnie jeden raz

Złożone topologie MQTT

- Możliwa jest łączność **broker-broker**. Technicznie jeden broker jest wtedy klientem dla drugiego, może wtedy subskrybować zdalne tematy i publikować własne.
- Taką konfigurację nazywa się **mostem** (bridge)
- MQTT **nie ma zabezpieczenia przed pętlą** stworzoną przez mosty, choć niektóre brokery (m.in. mosquitto) obsługują dwukierunkową replikację między dwoma brokerami w konfiguracji asymetrycznej (tj tylko jeden równocześnie publikuje i subskrybuje na drugim)

Implementacje

- Referencyjna implementacja brokera i biblioteki klienckiej C/C++ MQTT to **Apache Mosquitto** **`https://mosquitto.org`**
- Multi-językowa biblioteka kliencka (Java, C#, Go, C, Python i JavaScript) **Eclipse Paho** **`https://www.eclipse.org/paho/`**
- Inne implementacje są wymienione na Wikipedii

Zastosowanie

- Dużo klientów i nieduże komunikaty to ulubione środowisko życia MQTT
  - Komunikatory (prawdopodobnie Facebook korzysta za jakiejś mutacji MQTT)
  - Monitorowanie urządzeń i aplikacji
  - Systemy Command&Control

Przykład

- **Komunikator** oparty o MQTT w Pythonie3
- Biblioteka **`python3-paho-mqtt`** dla protokołu
- Biblioteka **`python3-tk`** dla GUI
- Użycie:

```
komunikator.py [-p PORT] [-s SERVER] username
```