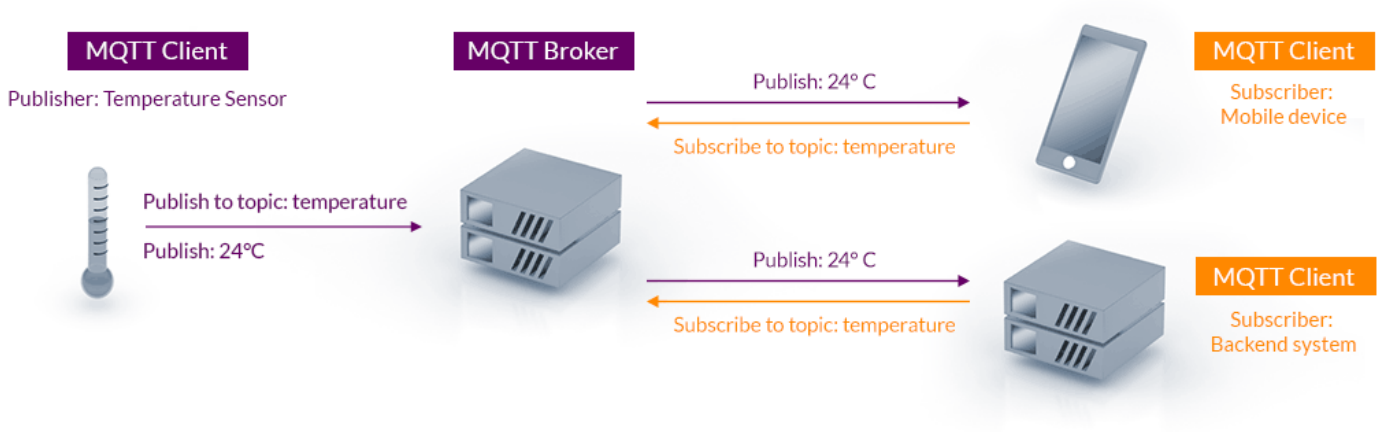
MQTT

Diese Seite beschreibt den Einsatz von MQTT um Datenpakete zwischen Applikationen zu vermitteln

MQTT Theorie

Wikipedia zu MQTT: ein offenes Netzwerkprotokoll für Machine-to-Machine-Kommunikation (M2M), das die Übertragung von Telemetriedaten in Form von Nachrichten zwischen Geräten ermöglicht, trotz hoher Verzögerungen oder beschränkter Netzwerke. Entsprechende Geräte reichen von Sensoren und Aktoren, Mobiltelefonen, Eingebetteten Systemen in Fahrzeugen oder Laptops bis zu voll entwickelten Rechnern.



In der MQTT Architektur werden Daten von einem Client zu einem Server (*Broker*) geschickt (*published*). Dieser puffert die Pakete für eine gewisse Zeit.

Pakete werden einem Thema (*topic*) zugewiesen. Topics sind hierarchisch angeordnet (z.B. projekte/connecting_peaks/v2023/miesbach-pxc-01 oder home_automation/lab_gmund/sensoren/tür01), womit man eine gewisse Struktur erstellen kann.

Andere Clients melden sich beim Broker an, abonnieren (*subscribe*) dedizierte Themen oder horchen allem zu. Sobald ein Datenpaket zu einem abonnierten Thema verfügbar ist, sendet ihn der Broker dem Client (Push-Verfahren).

Dank des Zwischenpufferns können zeitversetzt abgeholt werden, bzw können verzögert zugestellt werden falls die Verbindung instabil ist.

Die Kommunikation ist ggf bidirektional, d.h. ein Sensor published ein Paket, welches empfangen wird, und eine Reaktion auslöst, die als weiteres Paket über MQTT published wird, entweder zum

Sensor zurück, oder zu einem weiteren Aktuator.

MQTT und TTN

The Things Network bildet die Brücke zwischen IoT Geräte, die per LoRa funken, und Applikationen. TTN stellt einen MQTT Server zur Verfügung, der die Datenpakete weiterleitet.

Die Topics folgen dem Schema: v3/{application id}@{tenant id}/devices/{device id}, wobei

- *application id* ist z.B.oberlab-counter-sandbox, also der Name der Applikation wie der Benutzer sie vergeben hat
- *tenant id* ist “ttn”
- *device id* ist nicht die DeviceEID, sondern der DeviceName

Clients

- Grafische Oberfläche mqtt-explorer für Windows, Mac, Linux
- Kommandozeile unter Linux : z.B. mosquitto-client Paket, welches die Befehle mosquitto_sub und mosquitto_pub bereitstellt.
- Telegraf: Client, um die Datenpakete an eine Influx Datenbank weiterzuleiten

Ober-MQTT Server

Auf dem OberServer im Lab läuft ein zentraler MQTT Server (mit der Mosquitto Software). Er ist unter mqtt.oberlab.de erreichbar. Der Zugang ist allerdings nur mit Login/Passwort möglich. Es sollte pro Client ein eigener Satz an Credentials erstellt werden!

Er verwaltet nciht nur eigene Daten, sondern meldet sich als Client bei anderen Server als Client an (z.B. bei TTN), sammelt und konzentriert die Datenpakete. Somit reicht es, sich mit dem OberMTT Server zu verbinden, um auf alle MQTT-Daten zuzugreifen.

Daten von TTN zu OberMQTT spiegeln

Die Daten der Sandbox Applikation werden zu unseren Server gespiegelt, dabei wird der Topic zu connecting_peaks/sandbox geändert

Der passende Eintrag wird in `/docker/conf/mosquitto.conf` vorgenommen:

```
connection ttn_mqtt_01
address eu1.cloud.thethings.network:1883
remote_password <API Key>
remote_username <API Key User>
local_password <password>
local_username <username>
notifications true
# topic pattern [[[ out | in | both ] qos-level] local-prefix remote-prefix]
# subscribes to the remote topic $SYS/broker/clients/total and republishes the messages
received to the local topic test/mosquitto/org/clients/total
# topic clients/total in 0 test/mosquitto/org/ $SYS/broker/
topic # in 0 connecting_peaks/sandbox/ v3/oberlab-counter-sandbox@ttn/
```

Credentials

Die lokalen Credentials werden in `/docker/conf/mosquitto.conf` abgelegt

Version #2
Erstellt: 2024-05-14 19:58:23 UTC von Joel Hatsch
Zuletzt aktualisiert: 2024-05-20 14:52:50 UTC von Joel Hatsch