MQTT Broker TBMQ

- Einleitung
  - Was ist TBMQ
  - Wichtige Funktionen & Eigenschaften
  - Architekturintegration
  - TODO: TBMQ-Automated-Client-Generation
  - Nutzer (manuell) anlegen
  - MQTT Client Verbindungen
  - Subscriptions
  - Sessions

Einleitung

Was ist TBMQ

TBMQ ist ein Open-Source-MQTT-Broker von Thingsboard, mit dem Ziel, sehr große IoT-Installationen effizient und zuverlässig zu betreiben.
Einige Eckdaten:

- Unterstützt über 4 Mio. gleichzeitige Clients in Einzelknoten-Deployment, in Clusterbetrieb gar 100 Mio+. ThingsBoard+2GitHub
- Durchsatz: mindestens 3 Mio. Nachrichten pro Sekunde in einer Knoteninstallation. ThingsBoard
- Vollständige Unterstützung für MQTT Version 3.x und 5.0. ThingsBoard

Wichtige Funktionen & Eigenschaften

Die Kern-Funktionalitäten lassen sich in mehrere Kategorien unterteilen:

Skalierbarkeit & Verfügbarkeit

- Horizontal skalierbar: Clusterbetrieb mit identischen Knoten ohne Master-Single-Point. ThingsBoard
- Fehlertoleranz: kein zentraler Koordinator, jeder Knoten kann Clients bedienen und Nachrichten routen. Medium
- Dauerhaftigkeit (Durability): Nachrichten und Sessions können auch bei Node-Ausfällen gesichert werden. ThingsBoard

Kommunikationsmuster
TBMQ unterstützt typische IoT-Szenarien:

- „Fan-in“: Viele Geräte senden Daten an wenige Anwendungen.
- „Fan-out“: Wenige Quellen senden Updates, viele Geräte abonnieren.
- P2P (Point-to-Point): Direkte Kommunikation Gerät ↔ Gerät oder Gerät ↔ Anwendung über dedizierte Topics.

Sicherheits- & Managementfunktionen

- Authentifizierung: z. B. via JWT, X.509 Zertifikate. ThingsBoard
- Authorisierung/ACL: Zugriffskontrollen basierend auf ClientID, Username oder Zertifikat. ThingsBoard
- Monitoring & Management: Sessions- und Subscription-Überwachung, Clients-Metriken. ThingsBoard+1

Integration mit externen Systemen

- Weiterleitung von MQTT-Nachrichten an externe Systeme wie HTTP, Kafka oder andere MQTT-Broker. GitHub
- Unterstützung von MQTT über WebSocket. ThingsBoard

Technische Architektur

- Intern nutzt TBMQ Plattformen wie Apache Kafka und Redis je nach Einsatzszenario für persistente Sessions, hohe Last, geringe Latenz. DEV Community
- Datenbank für Metadaten (z. B. PostgreSQL) wird eingesetzt, aber nicht als Bottleneck im Messaging-Pfad.

Architekturintegration

Der MQTT-Broker ist selbst über MQTT an Thingsboard angebunden. Sensorpakete werden von Chirpstack aufgenommen und in den Applications mit Tags versehen. Diese Tags werden in Thingsboard verwendet um die Sensorpakete Gerätegruppen zuzuweisen. Dies wird im jeweiligen Data-Converter der verbundenen Integration umgesetzt. Ein gerät kann hierbei mehreren Gruppen zugeordnet werden. Jede Gruppe erhält später ihren eigenen Topic - so können Zugriffe über Abonnements von Kunden/Nutzern gesteuert werden.

Um Sensordaten über MQTT an Kunden zu übertragen, werden diese über einen Rulechain-Flow mittels eines eigenen MQTT Clients (bspw. PUB_stg_F&E_01) an TBMQ gesendet. Der Flow ruft hierbei für jedes Datenpaket mehrere Systematribute ab.

- Die Tenant-ID
- Den Application-Name (aus Chirpstack)
- Die Gruppen-ID

Aus diesen Informationen wird im MQTT-Node der Topic dynamisch generiert. Da dieser an die Kunden ausgespielt wird, sind Tenant und Gruppen durch die IDs (anstatt der Klarnamen) verschleiert. Zur besseren Übersicht werden die Application-Namen im Klartext eingebunden. Ein Topic kann somit beispielsweise so aussehen:

```
7f1857b0-51a9-11f0-8d0d-9b856896f048/df633c20-a8f2-11f0-85bc-
9bdb41bca33f/Pumpwerküberwachung/Dragino CS01-LB - Stromüberwachung NSA/#
```

Hierbei wird für jede Gerätegruppe (definiert über Tags in Chirpstack) ein eigener Topic generiert. Wie im obigen Schaubild zusehen, werden einzelne Datenpakete hierdurch auf mehrere Topics dupliziert und können so unterschiedlichen Clients zur Verfügung gestellt werden (Datenkapselung).

Dadurch lassen sich beispielsweise Sensordaten eines Kunden einem anderen zur Verfügung stellen ohne, dass dieser Kunde die übrigen Sensordaten des anderen Kunden einsehen kann.

Die Erstellung der Topics erfolgt somit in Thingsboard vollautomatisch und wird lediglich über Tags in Chirpstack gesteuert.
Wichitg ist hierbei, dass künftig alle Data-Converter so implementiert werden müssen, dass die Tags aus Chirpstack die Sensoren in Gerätegruppen organisieren.
Hier ein Beispiel-Baustein:

TbLib.js · main · SWHL Digital / IoT / Scripte_Parser / TB Data Converter / Basismodule für Konverter · GitLab

---

image.png

Rulechain-Flow zur Anreicherung der Datenpakete mit Systemparametern, Topicgenerierung und Übertragung via MQTT-Node.

Der “Publish”-Client zur Verbindung von TB mit TBMQ folgt folgender **Namenskonvention**:

PUB_[**Umgebungskennung**]_[**Tenant**]_[**Indexnummer**]

Beispiel:

- Name: PUB_stg_F&E_01
- Client_ID: PUB_stg_F&E_01
- Passwort: frei wählbar

image.png

Um die Gruppen-IDs zu erhalten ist ein API-call auf das eigene System notwendig. Hierfür ist in jedem Tenant ein eigener User einzurichten, über den auf die API zugegriffen werden kann.

Idealerweise heißt der User in jedem Tenant gleich.

image.png

Einleitung

TODO: TBMQ-Automated-Client-Generation

In TBMQ sollen basierend auf den durch Chirpstack erstellenten Topics automatisch MQTT Clients angelegt werden.

Hierzu ist ein Skript erforderlich, welches sich selbst per Client mit dem Broker verbindet und alle vorhandenen Topics überwacht. Ist ein Topic unbekannt, erstellt das Skript automatisch einen Client, welcher diesen Topic abonniert. Dieser Client kann anschließend dem Kunden zur Verfügung gestellt werden. Das Skript vergibt hierbei zunächst kein Passwort für den Client. Dieses wird bei der Ausrollung an den Kunden vergeben.

Nutzer (manuell) anlegen

Die Nutzerverwaltung in TBMQ befindet sich unter dem Reiter "Users". Hier sind alle angelegten Nutzer einsehbar. Es existiert lediglich die Nutzergruppe "Administratoren" - entsprechend ist jeder angelegte Nutzer automatisch Administrator mit Vollzugriff auf das gesamte System.

image.png

Um einen neuen Nutzer anzulegen wird über das + rechts oben das Nutzerformular geöffnet. Notwendig ist lediglich eine Email-Adresse.
Nach dem Erstellen des Nutzers ist eine erste Anmeldung möglich. Der Nutzername entspricht der Email, das Passwort ist initial immer `sysadmin`

Username: EMAIL

Password: `sysadmin`

Anschließend ist die Vergabe eines eigenen Passworts möglich.
Auch wenn dieser Schritt übersprungen werden kann ist er durchzuführen!

image.png

Einleitung

MQTT Client Verbindungen

Die Verwaltung der MQTT-Clients erfolgt über den Reiter "Credentials".

Auch wenn Clients gleichberechtigt sind wird zwischen folgenden Clients unterschieden. Die entsprechende Nomenklatur ist zu verwenden.

- **Clientverbindung**
  - Credencials für Kunden und Anwender, primär zur Subscription von Topics
  - Nomenklatur: `client_[Kunde]_[Tenant9_[Anwendung]_[Nummerierung]`
  - Beispiel: `client_Symcon_TRAVENETZ_wärmezähler_01`
- **Publisher Service**
  - Credencials für Anwendungen die Daten an den Broker publishen (bspw. Tenants aus Thingsbpard) aber keine Topics subscriben
  - Nomenklatur:
    - Name: `PUB_[Tenant]_[Anwendung oder Kunde]_[Nummerierung]`
    - Username: `PUB_[TB-Instanz]_[Tenant]_[Nummerierung]`
    - Topic restrictions: keine
  - Beispiel: SUB_dnno_planon_01
- **Subscriber Service**
  - Credencials für Nutzer/Anwendungen die Sensordaten aus Thingsboard subscriben (aber nicht publishen können)
  - Nomenklatur:
    - Name: `SUB_[Tenant]_[Anwendung oder Kunde]_[Nummerierung]`
    - Username: `[Thingsboard-Tenant-ID]_[Empfänger/Kunde]`
    - Topic restrictions:
      - keine publish-Authorisierung
      - Subscribe-pattern: `[Thingsboard-Tenant-ID]/[Empfänger/Kunde]/.*`
  - Beispiel anhand des Clients DNNO an Planon:image.png

Subscriptions

Subscriptions (Abonnements)

Verfügbar seit TBMQ Version 2.0

In MQTT ist ein *Subscription*-Eintrag ein Mechanismus, der es Clients ermöglicht, Nachrichten zu empfangen, die für bestimmte Topics bestimmt sind. Wenn ein Client ein Topic abonniert, signalisiert er damit sein Interesse daran, alle Nachrichten zu erhalten, die auf dieses Topic publiziert werden.

Tabelle: Subscriptions

Auf der Seite **Subscriptions** können alle Abonnements des Brokers beobachtet, analysiert oder gefiltert werden. Die Tabelle enthält folgende Informationen zu jedem einzelnen Subscription-Eintrag:

- **Client ID** – Die Kennung des Clients, der Eigentümer des Abonnements ist.
- **Topic Filter** – Der MQTT-Topic-Filter.
- **QoS** – *Quality of Service* des Abonnements.
- **Retain as Published** – Wenn „true“, behalten weitergeleitete Nachrichten das RETAIN-Flag, mit dem sie ursprünglich veröffentlicht wurden.
- **Retain Handling** – Bestimmt, wie der Broker Retained Messages behandelt, wenn ein Client ein Topic abonniert:
  - **0** – Retained Messages beim Abonnieren senden
  - **1** – Retained Messages beim Abonnieren senden, sofern das Abo noch nicht existiert
  - **2** – Keine Retained Messages beim Abonnieren senden
- **No Local** – Wenn „true“, sendet der Broker Nachrichten dieses Clients nicht zurück auf die Verbindung, über die das Abonnement erfolgt ist.
- **Subscription Identifier** – Eindeutiger numerischer Wert, der dem Abonnement zugeordnet ist. Er ermöglicht es dem Client, Nachrichten unterschiedlicher Abonnements zu unterscheiden.

Die Subscriptions-Tabelle zeigt alle aktuellen Abonnements des Brokers an.
Das **Filter**-Fenster ermöglicht eine einfache Filterung der Tabelle nach jeder Spalte.

image.png

Subscriptions verwalten

Abonnements können direkt im Fenster **Session details** hinzugefügt, entfernt oder bearbeitet werden.

Vorgehen:

1. Öffnen Sie im linken Menü die Seite **Subscriptions**.
2. Klicken Sie auf einen Eintrag um die **Session details** zu öffnen
3. Öffnen Sie den Tab **Subscriptions**, um die Session-Abonnements zu verwalten.
4. Fügen Sie Abonnements hinzu, bearbeiten oder löschen Sie diese.
5. Klicken Sie auf **Update**, um Änderungen zu speichern.

Subscriptions Chart

Auf den Seiten **Monitoring** und **Home** können Sie die Anzahl aktueller Abonnements und weitere Broker-Aktivitäten nachverfolgen.

Leeren von Subscription?Knoten

Subscriptions werden im Broker in einer **Trie**-Datenstruktur gespeichert, die besonders effiziente Suchvorgänge ermöglicht.
Ein Trie (oder Präfixbaum) ordnet Topic-Filter hierarchisch, wobei jeder Knoten eine Topic-Ebene repräsentiert.
Der Broker kann dadurch anhand eines publizierten Topic-Namens schnell die zugehörigen Client-Abonnements finden – was die Performance deutlich verbessert.

Wenn ein Client ein Abonnement entfernt (*unsubscribe*), löscht der Broker die Daten aus dem Speicher und markiert den entsprechenden Knoten als „leer“.
Mit der Zeit können sich viele solcher leeren Knoten ansammeln, was:

- Speicher verschwendet
- die Verarbeitung von Abgleichvorgängen (Subscription Matching) verlangsamen kann

Das Bereinigen leerer Knoten gibt Speicher frei und beschleunigt die Verarbeitung.

Leere Subscription-Knoten löschen:
Klicken Sie oben rechts auf die Schaltfläche **Clear empty subscription nodes** (Papierkorb-Symbol) und bestätigen Sie die Aktion.

Einleitung

Sessions

Sessions (Sitzungen)

Die zugehörige Seite bietet die Möglichkeit, alle im Broker gespeicherten Sitzungen zu beobachten und zu analysieren. Dies umfasst sowohl aktuelle **Online-Sessions** – also Clients, die derzeit mit dem Broker verbunden sind – als auch **Offline-Sessions persistenter Clients**.
Durch den Zugriff auf diese Seite erhalten Benutzer einen umfassenden Überblick über alle gespeicherten Sitzungen und können sowohl aktive als auch historische Client-Interaktionen mit dem Broker überwachen und verwalten.

Zugriff auf detaillierte TBMQ? Session?Informationen

Gehen Sie wie folgt vor:

1. Öffnen Sie im linken Menü die Seite **Sessions**.
2. Klicken Sie in der Tabelle auf die entsprechende Session-Zeile, um das Fenster **Session Details** zu öffnen.

Im Bereich **Session Details** können Benutzer:

- **Client trennen:**
  Klicken Sie auf **Disconnect client**.
  Hinweis: Nur verbundene Clients können getrennt werden.
- **Session löschen:**
  Klicken Sie auf **Remove session**.
  Hinweis: Nur getrennte (disconnected) Clients können gelöscht werden.

image.png

Session Details

Der Tab **Details** enthält folgende Informationen:

- **Connected Status** – Status der Verbindung (Connected/Disconnected).
- **Connected At** – Zeitpunkt, zu dem sich der Client verbunden hat.
- **Disconnected At** – Bei persistierenden Offline-Clients: Zeitpunkt, zu dem die Verbindung getrennt wurde.
- **Keep Alive (Sekunden)** – Zeitraum, den Broker und Client ohne Kommunikation bestehen dürfen, bevor die Session geschlossen wird.
- **Node ID** – Der Broker-Knoten, mit dem der Client verbunden war/ist.
- **Clean Start** – (Clean/Persistent Session) Wenn *true*, verwirft der Broker alle zuvor gespeicherten Sitzungsdaten und Nachrichten und beginnt eine neue Session.
- **Session Expiry Interval** – Zeitspanne, für die Session-Informationen nach einem Verbindungsverlust gespeichert werden.
- **Session End** – Zeitpunkt, zu dem die Session-Informationen und Nachrichten endgültig gelöscht werden.
- **Client ID** – Identifikator des Clients.
- **Client IP** – IP-Adresse des Clients.
- **Client Type** – Gerät oder Applikation.
- **Client Credentials** – Die Anmeldedaten, die für die aktuelle Session verwendet wurden.
- **MQTT-Version** – Verwendete Protokollversion: MQTT 3.1 (3), MQTT 3.1.1 (4) oder MQTT 5.0 (5).

Subscriptions

Im Tab **Subscriptions** können Benutzer die dem Client zugeordneten Abonnements einsehen, darunter:

- **Topic Filter** – Der MQTT-Topic-Filter.
- **QoS** – Quality of Service des Abonnements.
- **Retain as Published** – Wenn *true*, behalten Nachrichten das ursprüngliche RETAIN-Flag.
- **Retain Handling** – Legt fest, wie der Broker Retained Messages beim Abonnieren verarbeitet:
  - **0** – Retained Messages beim Abonnieren senden
  - **1** – Retained Messages senden, wenn das Abo noch nicht existiert
  - **2** – Keine Retained Messages beim Abonnieren senden
- **No Local** – Wenn *true*, sendet der Broker Nachrichten dieses Clients nicht an dessen eigene Verbindung zurück.
- **Subscription Identifier** – Eindeutige numerische ID zum Unterscheiden verschiedener Abonnements.

Obwohl die Session-Details im Allgemeinen schreibgeschützt sind, können Benutzer Abonnements dennoch verwalten (hinzufügen, entfernen, bearbeiten):

Verfügbare Aktionen:

- **Neues Abonnement hinzufügen:** Button **Add Subscription**
- **Abonnement entfernen:** Klick auf das Lösch-Symbol neben dem Eintrag
- **Topic Filter oder QoS bearbeiten:** Änderungen im Formular vornehmen und **Update** klicken

---

Metrics

Der Tab **Metrics** liefert detaillierte Informationen zum Nachrichtenfluss innerhalb der Sitzung. Dies unterstützt die Überwachung der Leistung und Zuverlässigkeit der Nachrichtenübertragung:

- **Received PUBLISH Messages** – Gesamtanzahl empfangener PUBLISH-Nachrichten
- **Received QoS 0 Messages** – Empfangen mit QoS 0 (*AT_MOST_ONCE*)
- **Received QoS 1 Messages** – Empfangen mit QoS 1 (*AT_LEAST_ONCE*)
- **Received QoS 2 Messages** – Empfangen mit QoS 2 (*EXACTLY_ONCE*)
- **Sent PUBLISH Messages** – Gesamtanzahl gesendeter PUBLISH-Nachrichten
- **Sent QoS 0 Messages** – Gesendet mit QoS 0
- **Sent QoS 1 Messages** – Gesendet mit QoS 1
- **Sent QoS 2 Messages** – Gesendet mit QoS 2

Zum Zurücksetzen der Statistik klicken Sie oben rechts auf das **Delete**-Symbol.