Background information:

MQTT - Message Query Telemetry Protocol

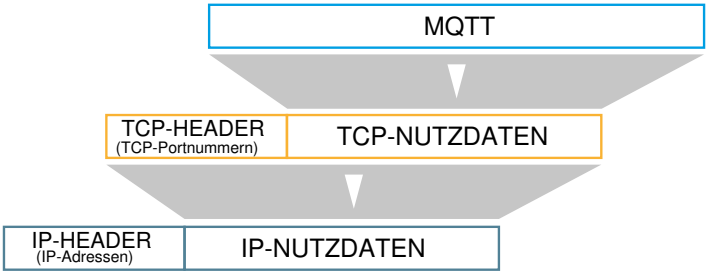
Data exchange without direct connection

The unique feature of MQTT is that two communication partners exchanging data with each other never have a direct connection.

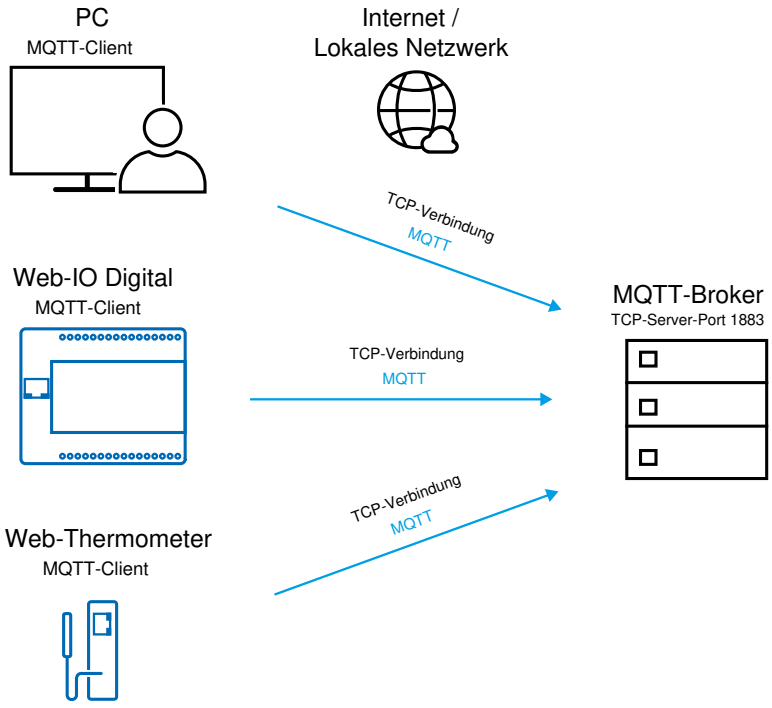
Instead there is a central data broker. The broker receives data from an MQTT user and passes it on to others.

MQTT - Transmission on the network level

MQTT uses TCP as the base protocol and therefore operates on the client/server principle.



The broker is the server, with by convention can accept connections on TCP port 1883. The MQTT users act as TCP clients and connect with the broker when needed.



MQTT - Data exchange on the protocol level

In MQTT there are two roles which a client can assume.

Publisher
As a publisher the client sends data to the broker. This can be measurement values, switching states or any desired process data. MQTT allows both readable text content and binary data. Whether the publisher sends its data to the broker when there is a change or cyclically depends upon the application.

Subscriber
The subscriber receives data from the broker. In the role of subscriber the client tells the broker after a connection is opened which data it would like to receive or subscribe to.

Any MQTT client can be a publisher, subscriber or both. There is no obligatory master/slave principle as with other industrial protocols. All MQTT terminal devices and clients are equal at first on the MQTT level. Which one delivers data and which one receives it is determined solely by the application using the publisher/subscriber arrangement.

Topic
The MQTT broker manages the exchanged data according to data end points. The data end points are named using topics. These are character strings which can be structured similar to the URL when the web page is opened.

Example:

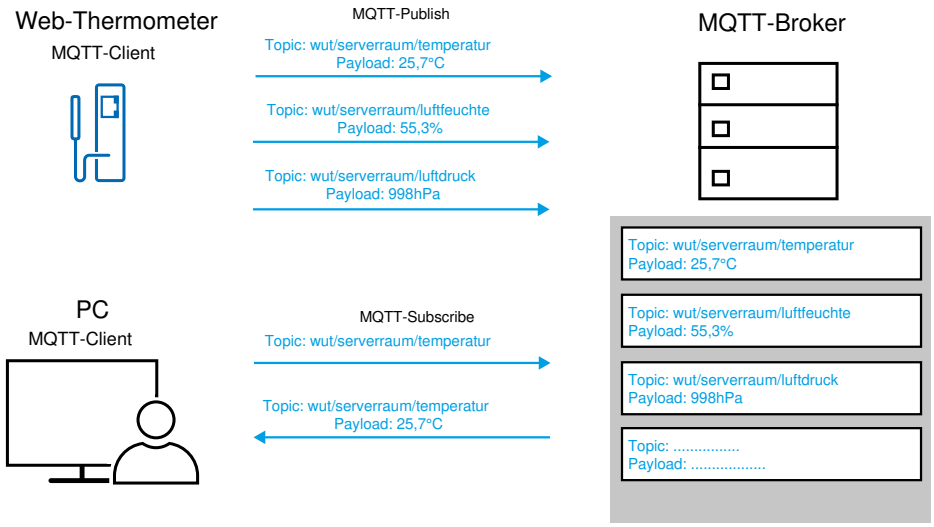
A W&T Web-Thermo-Hygrobarometer measures the temperature, relative humidity and barometric pressure in the server room of the W&T headquarters. Each of the three values represents a data end point.

The topics can look as follows:

`wut/serverraum/temperatur`
`wut/serverraum/luftfeuchte`
`wut/serverraum/luftdruck`

The Web-Thermo-Hygrobarometer uses MQTT publish to send the measured values as a payload, i.e. as data contents, to the MQTT broker under these topics.

Any MQTT client can now send a subscribe to the MQTT broker by indicating the desired topic. Here the MQTT client sends a subscribe to `wut/serverraum/temperatur` and thereby subscribes to the temperature value of the Web-Thermo-Hygrobarometer.



As long as the PC is connected as an MQTT client to the broker, it automatically get the value sent by the Web-Thermo-Hygrobarometer via Publish.

A "#" can also be used as a wildcard when the subscriber indicates topics.

Beispiel: `wut/serverraum/#` abonniert Temperatur, Luftfeuchte und Luftdruck für den Serverraum.

In addition to "#" you can also use "+" as a wildcard. Using "+" subscribes only to end topics on the corresponding level - topics from levels lying behind are ignored.

Each subscribed value has its own data send.

The data are sent in bytes and thereby bit-transparent - in other words any desired contents can be sent. UTF8 is used as a format for simplifying all text contents.

MQTT - Special features

MQTT offers special options for transporting and exchanging the data which use flags to set the option for each connection and each subscription.

Quality of Service - QoS
Values between 0 and 2 determine the reliability with which a publish message is sent to the broker.

0. At most once
   The MQTT client does not expect a confirmation for sent data.
   This procedure is less reliable but very fast.

1. At least once
   The MQTT client sends the data repeatedly if necessary
   until it gets a received confirmation from the MQTT broker.
   This procedure ensures that the data have arrived at the broker,
   if necessary but repeatedly

2. Exactly once
   Each data transmission must be explicitly acknowledged by the MQTT broker.
   This ensures that nothing is send in duplicate.
   This procedure is the most reliable, but also the slowest.

With QoS1 and 2 the question might be asked why a single data transmission is more reliable than a multiple. But this becomes quickly clear when we look at two examples.

When sending a measurement value - e.g. a temperature - it makes no difference whether the value arrives repeatedly at a subscriber. But if the angle a robot arm needs to move is sent - for example 5° - and the data transmission arrives at the subscriber three times, the robot arm would move 15°, which could have fatal consequences.

Last Will and Testament
A publisher can specify that in case the MQTT connection is lost, the broker instead of the subscribed values/data sends a specific message to the subscriber(s).

Retained Message
By setting this flag the publisher tells the broker to buffer store the last sent value / last sent data and immediately sends it to a newly connecting subscriber.

All three features are especially useful when sending over transmission paths that are not always reliable, such as mobile networks.

Characteristics and advantages of MQTT

MQTT is considered to be the standard transmission protocol for the Internet of Things and offers various advantages especially for exchanging data over the internet:

- Since all terminal devices using MQTT are clients, establishing firewalls and security measures is usually possible without or with minimum effort (firewalls require no port releases and no NAT routing / port forwarding. .
- The publisher does not have to worry about which recipients actually get the provided data.
- As opposed to other internet-based protocols, binary data can be sent without Base64 or other encoding.

Characteristics and drawbacks

- The data provider is not told who is actually receiving his data (no end-to-end acknowledgement).
- No real-time capability

Basics of common industrial protocols

Web-IO - Automation using standard protocols

Integrating signals with Modbus-TCP, OPC UA/DA, REST or MQTT

Data formats and protocols

From traditional data exchange to IoT

Modbus-TCP

Open standard for industrial communication

REST - REpresentational State Transfer

Industrial communication based on standardized HTTP requests

OPC UA -
OPC Unified Architecture

OPC support out of the box

OPC DA -
OPC Data Access

The process data interpreter

Products for industrial applications with standard protocols

Web-IO 4.0 Digital
2xIn, 2xOut

Power via PoE also when needed

Web-IO 4.0 Digital
12xIn, 12xOut

12x outputs (6-30V),
12x inputs (8-30V)

Other Web-IOs

All W&T Web-IO Digital 24V

We are available to you in person:

Wiesemann & Theis GmbH
Porschestr. 12
42279 Wuppertal
Phone: +49 202/2680-110 (Mon.-Fri. 8 a.m. to 5 p.m.)
Fax: +49 202/2680-265
info@wut.de

© Wiesemann & Theis GmbH, subject to mistakes and changes: Since we can make mistakes, none of our statements should be applied without verification. Please let us know of any errors or misunderstandings you find so that we can become aware of and eliminate them.

Data Privacy