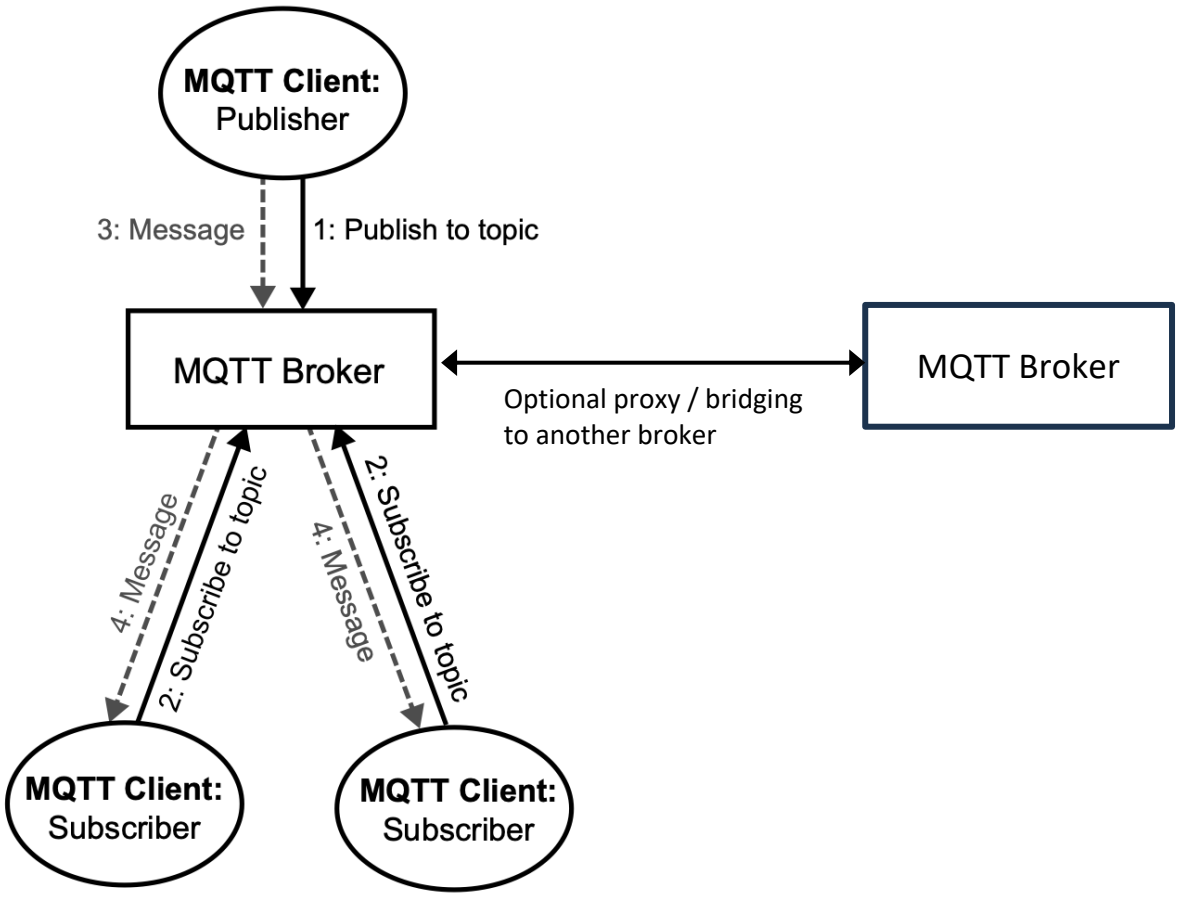
Introduction

You can integrate xxter with MQTT. The xxter controller can be used as a MQTT broker, with optional bridging support to connect to another broker, and as a MQTT client.

Overview

MQTT is a messaging protocol designed for transferring messages and based on a publish and subscribe model. MQTT is often used in Internet of Things (IoT) and Machine-to-Machine (M2M) environments to allow devices to communicate with each other and with back-end servers. MQTT is useful for enabling efficient and reliable communication between devices, particularly in environments where device resources are limited.
In MQTT a publisher publishes messages on a topic, and a subscriber must subscribe to that topic to view the message. MQTT requires the use of a central Broker and its as shown in the diagram below:



Main features:

- MQTT uses TCP/IP to connect to the broker.
- MQTT clients publish a keepalive message at regular intervals which tells the broker that the client is still connected
- Clients do not have addresses like in email systems, and messages are not sent to clients.
- Messages are published to a broker on a topic.
- The job of an MQTT broker is to filter messages based on topic, and then distribute them to subscribers.
- A client can receive these messages by subscribing to that topic on the same broker
- There is no direct connection between a publisher and subscriber.
- All clients can publish (broadcast) and subscribe (receive).
- MQTT brokers do not normally store messages.

MQTT can be used for:

- Transmit data between KNX or IoT devices via backend servers, for example to collect data from sensors or control actuators.
- Trigger real-time actions, such as notifying an app when a sensor detects a change in the state of the environment.
- Support two-way communication between devices, for example to allow an app to send commands to a KNX or IoT device.
- Manage data security through the use of authentication and encryption.

Setting up MQTT Broker

The MQTT broker is a service in the xxter controller, to support a local broker in the network.
The broker can optionally connect to another broker. A use case can be for example that several other clients (DALI gateways, MQTT sensors) connect to the xxter controller broker locally, and the broker in the xxter controller forwards the message to an external dashboard platform. This would be much easier to setup and allow better security.

General MQTT broker settings	(Local device protocol settings)
MQTT broker enabled	y/n
MQTT listen port	8883
MQTT versions supported	5.0 + 3.1.1 + 3.1
MQTT max connection timeout	300
MQTT enable TLS	y/n
MQTT certificate	auto generate / manual
MQTT custom certificate / key	upload PEM
Usernames / passwords	local users with MQTT access rights*
max_packet_size:	4096 (4k)

*) The usernames and password used in the broker are defined on my.xxter.com by adding local users with MQTT access rights. Please note that if you disable TLS for the broker, only users with no other access rights will be allowed to authenticate, for security reasons.

Setting up MQTT proxy / bridge

The MQTT broker in the controller can also be used as a bridge to forward message to another broker.

General MQTT proxy settings	(Local device protocol settings)
MQTT proxy enabled	y/n
Host	localhost
Port	8883
ClientID	[ ]
Add remote prefix	[ ]
Version to use	5.0 / 3.1.1
Use TLS	y/n
Validate certificate	y/n
Use TLS version	[tls1.1 / tls1.2 / tls1.3]
Root CA to use (empty default CA)	[upload pem]
Username	[ ]
Password	[ ]

Setting up MQTT client

The client connects the xxter Controller (and it components, status, logic) to the MQTT broker.
This can be the locally build in broker, but you can also connect directly to an external broker.

General MQTT client settings	(Local device protocol settings)
Enabled	y/n
Host	localhost
Port	8883
Client ID	xxter-12345
Version to use	5.0 / 3.1.1
Use TLS	y/n
Validate certificate	y/n
Root CA to use (empty default CA)	[upload pem]
Username	[ ]
Password	[ ]
MQTT Project settings	(Project MQTT settings on my.xxter.com)
Enable MQTT in project	y/n
Topic setup	
Prefix	xxter/%S
Status / Publish (xxter => MQTT)	/%T/sts
Send / Subscribe (MQTT => xxter)	/%T/cmd

Message setup

Status / Publish (xxter => MQTT)	%V	
Send / Subscribe (MQTT => xxter)	%V	(currently fixed setting)

Variable setup

Custom settings 1 (%1)	[ ]	[ ]	set through LUA
Custom settings 2 (%2)	[ ]	[ ]	set through LUA
Custom settings 3 (%3)	[ ]	[ ]	set through LUA
Custom settings 4 (%4)	[ ]	[ ]	set through LUA
Custom settings 5 (%5)	[ ]	[ ]	set through LUA
Custom settings 6 (%6)	[ ]	[ ]	set through LUA
Custom settings 7 (%7)	[ ]	[ ]	set through LUA
Custom settings 8 (%8)	[ ]	[ ]	set through LUA
Custom settings 9 (%9)	[ ]	[ ]	set through LUA

These variables can be used as replacements for %1 to %9 in message body or topics, and can be set as a static text in the project, or can be set using a LUA function dynamically from a LUA script.

Global messages

Birth message (after connect)

Topic:	/connected
Retain:	Y
QoS:	1
Message:	[0]

Will message (irregular disconnect)

Topic:	/connected
Retain:	Y
QoS:	1
Message:	[1]

MQTT project components settings (Project components settings on my.xxter.com)

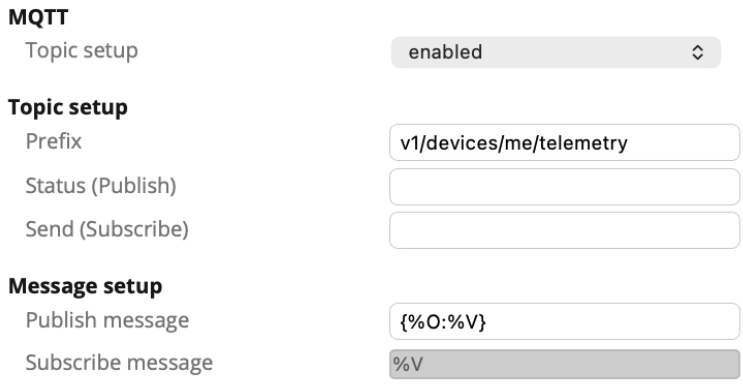
For each component (object) you can setup:

- MQTT available: none, to MQTT (publish), from MQTT (subscribe), both
- MQTT topic: /123 [%T]
- MQTT object identifier: light123 [%O] can be used in message body e.g.
- Retain: y/n
- QoS: 0/1/2

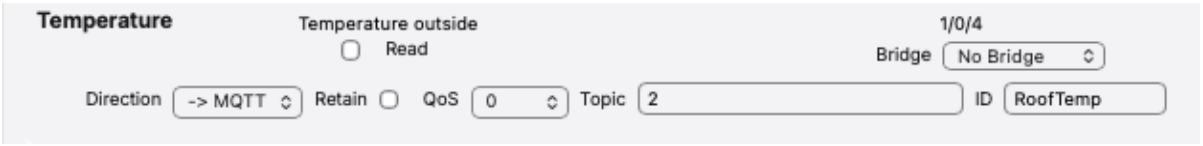
Example MQTT-client integration with Thingsboard©

When using the xxter controller as an MQTT-client, an integration can be made to Thingsboard for example. Log in to the Thingsboard cloud, and create a “Device” for your xxter controller, together with the credentials for it (see below).

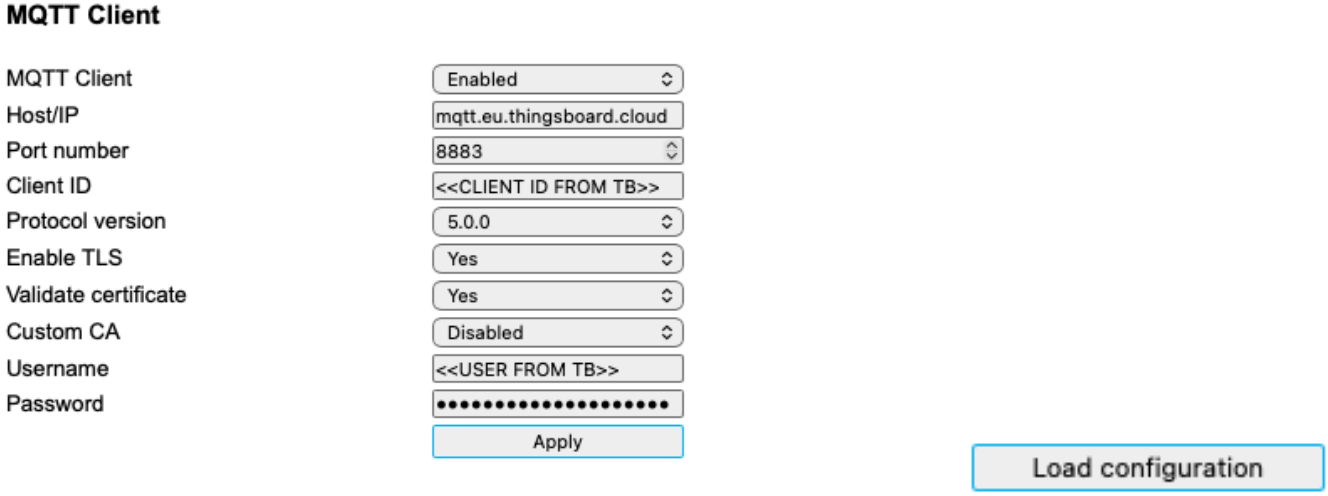
Then, in My xxter, enable MQTT and setup the Topic and message as follows:



Subsequently, you can enable the components you want to publish to MQTT and set a topic and ID:



Lastly on the xxter controller itself, on the Protocol page, enable the MQTT client, enter the Thingsboard server information and credentials, and press apply.



When you then load the configuration, the xxter controller will start publishing the data to Thingsboard.

On Thingsboard you can create dashboard and much more. For more information about Thingsboard, visit https://thingsboard.io.

MQTT variables

The xxter controller has support for several variables which will be replaced on publish (or subscribe) of a message / topic.

%S	Serial number of the controller
%M	Product model (HK04F e.g.)
%D	ClientID
%T	Object topic name (setup on the project components)
%t	Object type (switch, temperature etc)
%o	object identifier (fixed id from within controller)
%O	object identifier (setup on the project components)
%n	Object name (the name of the object)
%V	raw value (depending on type)
%v	value including name, can be multiple values. (e.g. dimmer: "value":50, "onoff":1)

Special variable options:

%1 … %9	variable can be set in project, or be set with LUA
%U	timestamp UTC (ms precision)
%u	timestamp UTC (sec precision)
%L	timestamp local time (ms precision)
%l	timestamp local time (sec precision)
%RUID	Unique message ID – generated