

Практическая работа №6 – Основы работы с протоколом MQTT. Брокераж сообщений

MQTT протокол, основные особенности

MQTT (англ. message queuing telemetry transport) — упрощённый сетевой протокол, работающий поверх TCP/IP, ориентированный на обмен сообщениями между устройствами по принципу издатель-подписчик.

Протокол ориентируется на простоту в использовании, невысокую нагрузку на каналы связи, работу в условиях постоянной потери связи, лёгкую встраиваемость в любую систему. Основное предназначение — работа с телеметрией от различных датчиков и устройств. Использование шаблона подписчика обеспечивает возможность устройствам выходить на связь и публиковать сообщения, которые не были заранее известны или предопределены, в частности, протокол не вводит ограничений на формат передаваемых данных.

Главные особенности протокола MQTT:

- Компактный и легковесный — минимальные накладные расходы на пересылку данных, для экономии трафика.
- Устойчивость к потерям — гарантированная доставка в условиях нестабильных сетевых подключений.
- Асинхронный — позволяет обслуживать большое количество устройств, и не зависит от сетевых задержек.
- Поддержка QoS — возможность управлять приоритетом сообщений и гарантировать доставку сообщения адресату.
- Динамическая конфигурация — не требует предварительно согласования полей и форматов данных, может конфигурироваться «на лету».
- Работает за NAT — клиенты могут находиться за NAT, только сервер (брокер) должен иметь реальный IP. Позволяет обойтись без VPN и пробрасывания портов.
- Удобная адресация — поля данных имеют текстовые названия, понятные для человека. Не нужно запоминать цифровые адреса и битовые смещения.

Протокол MQTT работает на прикладном уровне поверх TCP/IP и использует по умолчанию 1883 порт (8883 при подключении через SSL).

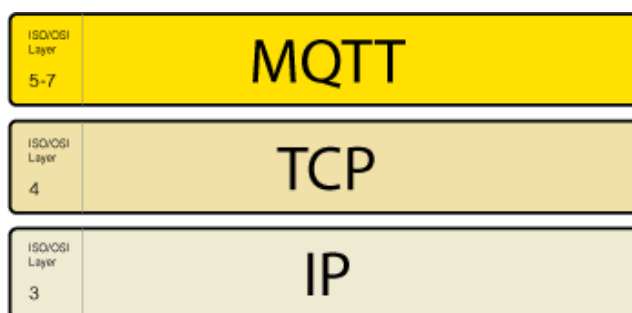


Рисунок 4 – уровни модели OSI

Принцип работы MQTT

Обмен сообщениями в протоколе MQTT осуществляется между клиентом (client), который может быть издателем или подписчиком (publisher/subscriber) сообщений, и брокером (broker) сообщений (например, Mosquitto MQTT).

Брокер (Broker) — это центральный узел MQTT, обеспечивающий взаимодействие клиентов. Обмен данными между клиентами происходит только через брокера. В качестве брокера может выступать серверное ПО или контроллер. В его задачи входит получение данных от клиентов, обработка и сохранение данных, доставка данных клиентам, и контроль за доставкой сообщений.

Publisher/Subscriber

Для понимания разницы между Publisher и Subscriber разберем простой пример: датчик влажности измеряет влажность в помещении, и? если она опустилась ниже определенного уровня, включается увлажнитель воздуха.

В данном случае датчик влажности выступает в роли Publisher: его задача сводится только к публикации данных в сторону брокера. Увлажнитель воздуха выступает в роли Subscriber: он подписывается на обновления данных о влажности и получает от брокера актуальные данные, при этом увлажнитель может сам решать, в какой момент включать увлажнение.

В этой схеме MQTT-клиенты, то есть датчик и увлажнитель, не знают о существовании друг друга, и не взаимодействуют напрямую. Брокер может получать данные из разных источников, проводить над ними манипуляции, например, рассчитывать среднее значение от нескольких датчиков, и уже обработанные данные возвращать подписчику.

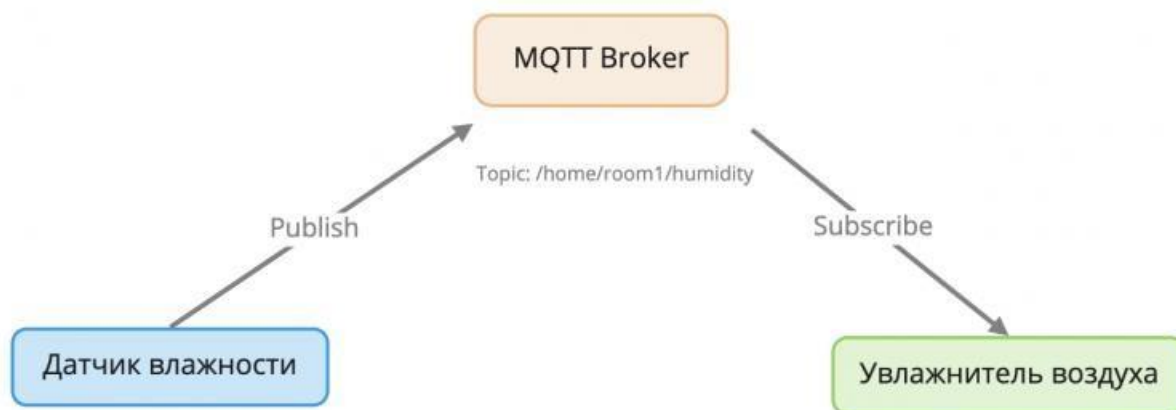


Рисунок 5 – Publisher посылает данные брокеру, Subscriber подписывается на обновления этих данных

При этом, асинхронность протокола MQTT предусматривает, что датчик и увлажнитель могут быть онлайн в разное время, терять пакеты, и быть недоступны. Брокер позаботится о том, чтобы сохранить в памяти последние данные, полученные от датчика, и обеспечить их доставку на увлажнитель.

Издатель отправляет данные на MQTT брокер, указывая в сообщении определенную тему, топик (topic). Подписчики могут получать разные данные от множества издателей в зависимости от подписки на соответствующие топики.

Устройства MQTT используют определенные типы сообщений для взаимодействия с брокером, ниже представлены основные:

- Connect – установить соединение с брокером
- Disconnect – разорвать соединение с брокером
- Publish – опубликовать данные в топик на брокере
- Subscribe – подписаться на топик на брокере
- Unsubscribe – отписаться от топика



Рисунок 6 – Схема простого взаимодействия между подписчиком, издателем и брокером

Семантика топики

Для идентификации сущностей в MQTT используются топики (или каналы). Топики состоят из UTF8-символов, и имеют древовидную структуру, похожую на файловую систему в UNIX. Это удобный механизм, позволяющий называть сущности в человекопонятном виде.

Пример топики в MQTT:

```
# Датчик температуры на кухне
home/kitchen/temperature
# Датчик температуры в спальне
home/sleeping-room/temperature
# Датчик освещенности на улице
home/outdoor/light
```

Такой подход позволяет наглядно видеть, какие данные передаются, и удобно разрабатывать и отлаживать код, без необходимости запоминать цифровой адрес размещения данных,

Иерархическая структура топики имеет формат «дерева», что упрощает их организацию и доступ к данным. Топики состоят из одного или нескольких уровней, которые разделены между собой символом «/».

Пример топика, в который датчик температуры, расположенный в спальном комнате, публикует данные брокеру:

```
/home/living-space/living-room1/temperature
```

Подписчик может так же получать данные сразу с нескольких топиков, для этого существуют **wildcard**. Они бывают двух типов: одноуровневые и многоуровневые. Для более простого понимания рассмотрим в примерах каждый из них:

1. Одноуровневый wildcard. Для его использования применяется символ «+»

К примеру, нам необходимо получить данные о температурах во всех спальнях комнаты:

```
/home/living-space+/temperature
```

В результате получаем данные с топиков:

```
/home/living-space/living-room1/temperature  
/home/living-space/living-room2/temperature  
/home/living-space/living-room3/temperature
```

2. Многоуровневый wildcard. Для его использования применяется символ «#»

К примеру, чтобы получить данные с различных датчиков всех спален в доме:

```
/home/living-space/#
```

В результате получаем данные с топиков:

```
/home/living-space/living-room1/temperature  
/home/living-space/living-room1/light1  
/home/living-space/living-room1/light2  
/home/living-space/living-room1/humidity  
/home/living-space/living-room2/temperature  
/home/living-space/living-room2/light1
```

Работа с MQTT в WirenBoard

Подробно о работе MQTT на контроллере WirenBoard вы можете ознакомиться в документации к оборудованию: <https://wirenboard.com/wiki/MQTT>

Задание практической работы №6

Часть 1. SSH-подключение

Подключитесь к консоли WirenBoard по протоколу SSH. Для этого используйте SSH-клиент PuTTY (или другой клиент):

1. Запустите Терминал.
2. Введите в поле терминала `ssh user@192.168.X.Y`, где X, Y номер чехла.

3. При появлении ошибки доступа сгенерируйте fingerprint: `ssh-keygen -R 192.168.X.Y`, проведите повторное подключение.
4. Введите пароль пользователя 123123(вводимые символы не будут отображаться).
5. Появится приветственное сообщение - вы подключились к консоли контроллера.

```
C:\Users\romaa>ssh user@192.168.2.25
user@192.168.2.25's password:

Welcome to Wiren Board 8.5.2 (s/n AH4QUHWR), release wb-2507 (as stable)
Linux wirenboard-AH4QUHWR 6.8.0-wb140 #1 SMP Fri Jul 18 08:08:05 UTC 2025 aarch64 GNU/Linux

System load:  1.35 1.20 1.11   Up time:       54 min
Memory usage: 13% of 1.94G    Usage of /:    43% of 2.0G    /mnt/data:    15% of 13G

11 package updates are available; type 'apt update && apt upgrade' to update them.
```

Рисунок 9 – Приветственный баннер контроллера

Часть 2. Подписка на топик

При помощи утилы **mosquitto_sub** подпишитесь на сообщения нескольких датчиков стенда, согласно варианту. Первое задание выполняются на wb-demo-kit v2, второе на wb-demo-kit v3:

№ варианта	Датчики
1	1. Датчик температуры устройства WB-MSW v.3 (5) 2. Яркость лампочки EL1
2	1. Датчик CO ₂ устройства WB-MSW v.3 (5) 2. Кнопка SB2
3	1. Датчик температуры 1-wire DS18B20 (2) 2. Скорость вращения вентилятора WB-MDM3
4	1. Датчик движения устройства WB-MSW v.3 (5) 2. Датчик протечки WB-MWAC v.2
5	1. Датчик влажности устройства WB-MS v.2 (12) 2. Кнопка SB1
6	1. Датчик движения устройства WB-MSW v.3 (5) 2. Датчик температуры WB-M1W2
7	1. Световой индикатор (25) 2. Датчик шума устройства WB-MSW v.4

Пример получения сообщений в результате успешной подписки:

```
~# mosquitto_sub -t '/devices/<Имя устройства>/controls/<Идентификатор датчика>' -v -h 192.168.X.Y //команда на получение сообщения из топика
/devices//<Имя устройства>/controls/<Идентификатор датчика> 22.75 //в этой строке и ниже – вывод утилиты, полученные сообщения
/devices//<Имя устройства>/controls/<Идентификатор датчика> 22.75
/devices//<Имя устройства>/controls/<Идентификатор датчика> 22.75
```

Часть 3. Управление устройствами

Для управления устройством (изменения значения канала), необходимо отправить сообщение в топик `/devices/<device-id>/controls/<control-id>/on` (обратите внимание на `/on` в конце). Название топика можно также посмотреть в веб-интерфейсе контроллера во вкладке *Settings – MQTT Channels*. Это делается с помощью консольной команды `mosquitto_pub`. Например, данная команда отправляет сообщение «1» (логическую единицу, «включить») в топик, соответствующий подключённому по RS-485 релейном модуле WM-MRM2 с адресом 130.

```
~# mosquitto_pub -t "/devices/wb-mrm2_130/controls/Relay 1/on" -m "1" -p 1883
```

Включите или измените поведение устройств посредством отправки сообщение в соответствующий топик согласно вариантам:

№ варианта	Действие
1	1. Включите звуковой сигнал 2. Включение и изменение скорости вентилятора
2	1. Откройте и закройте шаровой кран 2. Включение и изменения яркости лампочки накаливания
3	1. Включите индикатор 25 2. Изменение цвета RGB-ленты
4	1. Включите подсветку кнопки 29 2. Изменение насыщенности RGB-ленты
5	1. Включите и измените уровень громкости звукового сигнала 2. Включение и выключение системы водоснабжения
6	1. Включите индикатор устройства WB-MSW v.3 (5) 2. Включите подсветку кнопки SB2
7	1. Включите и выключите RGB-ленту 2. Включение и выключение аварии системы водоснабжения

Часть 4*. Сообщения MQTT с внешнего устройства

**Примечание. Работа выполняется только в случае подключения стендов к сети Интернет.*

Настройте MQTT-мост (т.е. пересылку всех сообщений MQTT на любой облачный MQTT брокер и обратно) или создайте собственный брокер MQTT с помощью брокера **Clusterfly**.

Обязательно верните файл конфигурации 20bridges.conf контроллера к изначальному состоянию после выполнения работы!

Процесс настройки приведен в [документации](#) к контроллеру WirenBoard.

ВНИМАНИЕ! Необходимо редактировать файл конфигурации, находящийся в домашней директории пользователя **user** (а не /etc/mosquitto/mosquitto.conf) !