УДК 621.391.8

АНАЛИЗ ИСПОЛЬЗОВАНИЯ ПРОТОКОЛОВ ВЗАИМОДЕЙСТВИЯ ВО ВСТРАИВАЕМЫХ СИСТЕМАХ

Житковский Е.А., Ельников Е.П., Тонко И.А.

Белорусский государственный университет информатики и радиоэлектроники, г. Минск, Республика Беларусь

Научный руководитель: Пискун Г.А. – канд.техн.наук, доцент

Аннотация. В настоящее время существует потребность в соединении устройств, находящих на расстоянии друг от друга. Осуществить данную задачу можно при помощи сети интернет. Для передачи данных по сети существует множество различных протоколов, но не все из них выгодно использовать во встраиваемых системах. В статье проведено сравнение некоторых наиболее популярных протоколов взаимодействия.

Ключевые слова: встраиваемые системы, интернет вещей, сетевой протокол

Введение. В настоящее время Интернетом Вещей охватывается огромный спектр отраслей, начиная от промышленности и заканчивая продуктами питания. Данные от устройств передаются на облачные сервера. Для передачи данных используются протоколы, которых в настоящий момент насчитывается около двадцати пяти. Среди существующих протоколов Интернета Вещей наибольшее распространение получили протоколы *MQTT*, *CoAP*, *HTTP*/2, которые используются для сбора и передачи данных между устройствами серверной инфраструктурой.

Основная часть. *MQTT* (*Message Queuing Telemetry Transport*) – простой протокол обмена сообщениями, реализующий модель «публикации/подписки» (*publish*/*subscribe*) и предназначенный для связи компьютеризированных устройств, подключённых к локальной или глобальной сети, между собой и различными публичными или приватными веб-сервисами.

Протокол создавался, чтобы обеспечить открытость, простоту, минимальные требования к ресурсам и удобство внедрения. *MQTT* располагается поверх *TCP/IP* и работает с моделью «клиент/сервер», где каждый датчик является клиентом и подключен к серверу, который является брокером. Протокол *MQTT* требует обязательного наличия брокера, который управляет распределением данных подписчикам. Все устройства или подписчики посылают данные только брокеру и принимают данные тоже только от него.

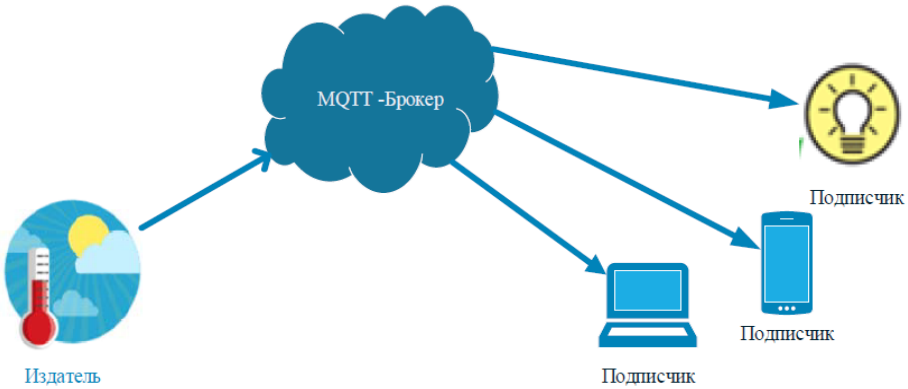

Рисунок 1 – Структура работы протокола *MQTT*

В сети на базе протокола *MQTT* различают 3 объекта (рисунок 1):

– издатель (*Publisher*) – *MQTT*-клиент, который при возникновении определенного события передает брокеру информацию о нём, публикуя соответствующие топики;

– брокер (*Broker*) – *MQTT*-сервер, который принимает информацию от издателей и передает ее соответствующим подписчикам, в сложных системах может выполнять также различные операции, связанные с анализом и обработкой поступивших данных. Разные брокеры могут соединяться между собой, если они подписываются на сообщения друг друга;

– подписчик (*Subscriber*) – *MQTT*-клиент, который после подписки к брокеру большую часть времени «слушает» его и постоянно готов к приему и обработке входящего сообщения на интересующие топики от брокера.

То есть, когда один клиент, так называемый издатель, передает сообщение М на определенную тему Т, то все клиенты, которые подписываются тему Т, получают это сообщение М. Например, три клиента подключены к брокеру, клиенты Б и С подписываются на топик «*temperature*». В какое-то время, когда клиент А передает значение «30» на топик «*temperature*», сразу после получения, брокер передает это сообщение к подписавшимся клиентам (рисунок 2).

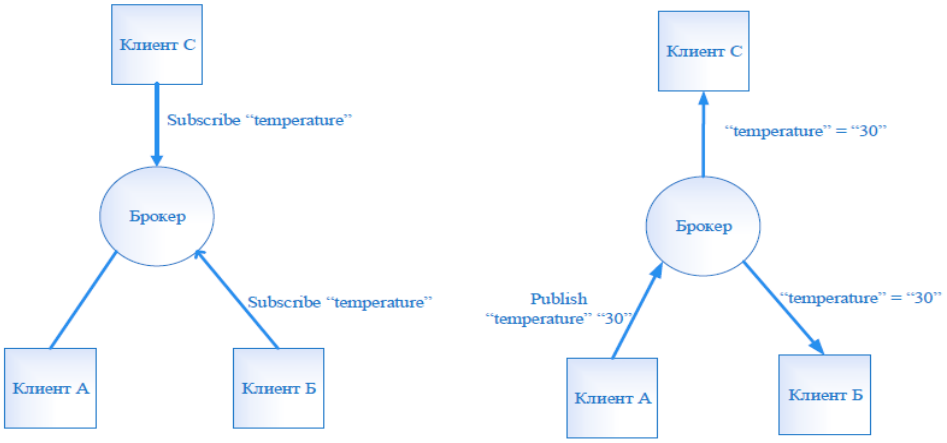

Рисунок 2 – Схема обмен сообщениями между клиентами

В *MQTT* предусматривается 3 выбора надежности обмена сообщениями, которые обеспечиваются тремя уровнями качества обслуживания (*QoS, Quality of Service*):

– *QoS0* – сообщение передается только один раз и не требует подтверждение;

– *QoS1* – сообщение отправляется минимум один раз и требует подтверждение;

– *QoS2* – для доставки сообщения используется механизм четырехэтапного рукопожатия.

Кроме того, поверх уровня *TCP* стоит уровень стандартной безопасности *TLS* (*Transport Layer Security*), ранее известный как *SSL* (*Secure Sockets Layer*). Порт 8883 обеспечивает безопасность связи, если адрес брокера работает с этим портом, то трафик передаётся с шифрованием.

CoAP (*Constrained Application Protocol*) – протокол, разработанный Инженерным советом Интернета (*IETF, Internet Engineering Task Force*) и описан в документе *RFC* 7252. Протокол работает на прикладном уровне, и предназначен для передачи данных по линиям с ограниченной пропускной способностью. *CoAP* был разработан на основе протокола *HTTP*, представляет собой двоичную его версию, но не является слепым его сжатием. *CoAP* состоит из подмножества *HTTP* функциональных возможностей, которые были вновь разработаны с учетом низкой мощности и малого потребления энергии ограниченных встраиваемых устройств, например, такие как датчик уровня пыли в помещении. Кроме того, были изменены различные механизмы и добавлены некоторые новые возможности, чтобы протокол подходил для Интернета Вещей. Стеки протоколов *HTTP* и *COAP* показаны на рисунке 3.

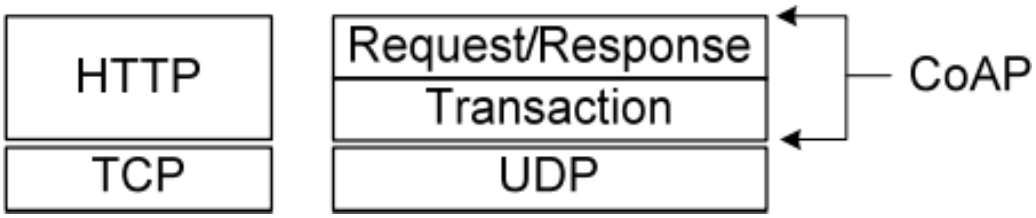

Рисунок 3 – Стеки протоколов *HTTP* и *COAP*

Так, в отличие от протокола *HTTP*, который является текстовым и использует *TCP*, *CoAP* – это бинарный протокол, который транспортируется через *UDP*, что уменьшает размер его служебных данных и повышает гибкость в моделях связи.

CoAP организован в два слоя: слой транзакций и слой «*Request*/*Response*».

Слой транзакций обрабатывает единый обмен сообщениями между конечными точками. Сообщения обмена на этом слое могут быть четырех типов:

– «*Confirmable*» – требует подтверждение;
– «*Non-confirmable*» – не требует подтверждение;
– квитирование – подтверждает получение «*Confirmable*» сообщения;
– «*Reset*» – указывает на то, что «*Confirmable*» сообщение было получено, но контекст, подлежащий обработке, отсутствует.

Данными сообщениями *CoAP* обеспечивает механизм собственной надежности. Когда получателю приходит «*Confirmable*» сообщение, он всегда возвращает подтверждение. Передающая сторона, в свою очередь, повторно отправляет сообщение, если подтверждение не возвращается в течение определенного периода времени, заданного по умолчанию и с каждым разом возрастающего экспоненциально, пока получатель не посылает сообщение подтверждения приема. Кроме того, это дает возможность асинхронной связи, которая является ключевым требованием для Интернета Вещей.

Но некоторые сообщения не требуют подтверждения. Тогда в качестве более легкой альтернативы, сообщение может быть передано с меньшей надежностью, помеченное как «*Non-confirmable*».

Слой «*Request* / *Response*» представляет модель взаимодействия «запрос/ответ» (клиент/сервер) для манипулирования ресурсами и передачи. Сенсорный узел обычно является сервером, но может реализовывать оба стиля общения, что сокращает время разработки и ресурсы, необходимые на устройствах.

В *CoAP* поддерживается шифрование, но без *TCP* стандартный *TLS* не может быть использован для обеспечения безопасности связи. Поэтому в *CoAP* используется *DTLS* (*Datagram Transport Layer Security*), более новая производная *TLS*, которая за счет дополнений позволяет ему работать на вершине своего *UDP* транспортного протокола.

Протокол *HTTP/2* – это обновленный протокол *HTTP* версии 2, который был разработан *IETF* и описан в документе *RFC* 7540. Он полностью совместим со своим предшественником – *HTTP*/1.1, который в свою очередь не хорошо подходит для встраиваемых устройств, так как тратит много оперативной памяти и буферного пространства. Кроме этого, он затрачивает больше энергии, из-за чего встает проблема с батарейным питанием. Новая версия протокола *HTTP*/2 создана, чтобы справляться с данными проблемами. Конечно, он не будет столь же эффективным и идеальным для встраиваемых устройств, как протокол, специально предназначенный для этого, но при проектировании протокола *HTTP*/2 была рассмотрена данная возможность. Технология *HPACK* работает путем присвоения имен заголовков и значений записей в таблицах. Затем, используется только необходимый номер записи. *HPACK* позволяет уменьшить размер заголовка *HTTP*, за счет чего эффективно используется пропускная способность канала, и также уменьшается время необходимое для обработки анализа, не жертвуя исходным синтаксисом *HTTP*.

Одним из недостатков *HTTP*/1.1 является тот факт, что, когда *HTTP* сообщение отправлено с заголовком *Content-Length* определённой длины, просто так его остановить не представляется возможным. Конечно, зачастую можно (но не всегда) разорвать *TCP*-соединение, но ценой повторного согласования нового *TCP*-соединения. Гораздо лучше просто отменить отправку и начать новое сообщение. Это может быть достигнуто отправкой *HTTP*/2фрейма *RST_STREAM*, который таким образом предотвратит растрату полосы пропускания и необходимость разрыва каких-либо соединений.

*HTTP/*2 является бинарным протоколом, т. е. поток делится на фреймы, имеющие фиксированную структуру и размер. Бинарный формат помогает уменьшить размер отправляемого пакета. В спецификации *HTTP*/2 существует десять различных типов фреймов, но наиболее важными, которые связывают с *HTTP*/1.1 являются: *DATA* (данные) и *HEADERS* (заголовки).

Таблица 1 – Основные различия в протоколах *CoAP, MQTT и HTTP*/2

Протокол	***MQTT***	***CoAP***	***HTTP*/2**
Транспортный уровень	*TCP*	*UDP*	*TCP*
Безопасность	*TLS/SSL*	*DTLS*	*TLS/SSL*
Обмен сообщениями	Издатель/Подписчик	Запрос/Ответ	Запрос/Ответ
Надежность	3 типа: *QoS0, QoS1, QoS2*	2 типа: *Confirmable, Non-Confirmable*	–

Структурируем основные различия между протоколами *CoAP*, *MQTT* и *HTTP*/2 и представим в таблице 1. Протоколы *CoAP* и *MQTT* предполагаются для связи шлюза к серверу. В настоящее время многочисленное количество протоколов используется для этих целей, но данные протоколы получили наибольшее распространение при разработке Интернет Вещей. Также возможно эффективное использование *CoAP* и *MQTT*, когда необходимо отправлять короткие сообщения. Протокол *HTTP*/2 больше предполагает использование для Веб Вещей (*WoT, WEB of Things*).

Список литературы

1. Росляков А. В., Ваняшин С. В., Гребешков А. Ю., Самсонов М. Ю. Интернет вещей / под ред. А. В. Рослякова. Самара : ПГУТИ, ООО «Издательство Ас Гард», 2014. 340 с.
2. Кучерявый А. Е. Интернет Вещей // Электросвязь. 2013. № 1. С. 21–24.

UDC 621.391.8

ANALYSIS OF PROTOCOL USE INTERACTION IN EMBEDDED SYSTEMS

Zhitkovsky E.A., Elnikov E.P. , Tonko I.A.

Belarusian State University of Informatics and Radioelectronics, Minsk, Republic of Belarus (style T-institution)

Piskun G.A. – PhD, aasoooociate professor

Annotation. Currently, there is a need to connect devices that are at a distance from each other. You can accomplish this task using the Internet. There are many different protocols for transmitting data over a network, but not all of them are beneficial to use in embedded systems. The article compares some of the most popular communication protocols.

Keywords. embedded systems, internet of things, network protocol