**СТАНДАРТ ПРОТОКОЛА MQTT v.3.1.1.**

*Вольный перевод:* [*Zolandv@gmail.com*](mailto:Zolandv@gmail.com) *20/01/2018*

Протокол MQTT – Message Queuing Telemetry Transport – протокол для передачи последовательности сообщений с телеметрическими данными, то есть информации от датчиков температуры, влажности, освещенности и др.

MQTT был предложен в 1999 г. Энди Стандфордом-Кларком в качестве протокола, который бы служил для передачи данных о состоянии нефтепровода и газопровода в реальном времени. Разработка велась компанией IBM для нового трубопровода крупнейшей американской нефтяной компании ConocoPhillips. В рамках создания диспетчерской системы управления и сбора данных (SCADA) необходимо было обеспечить гарантированный сбор самой различной информации: состояние насосов, температура подшипников, скорость потоков, состояние клапанов, уровни в баках и т.д. При этом необходимо было учесть дороговизну каналов связи и узкую полосу пропускания. Ни один из существующих протоколов не подходил под эти задачи, таким образом, сформировались требования к новому протоколу: качество обслуживания, двусторонняя связь, эффективное использование полосы пропускания [1].

Впервые протокол MQTT был опубликован консорциумом OASIS (Organization for the Advancement of Structured Information Standards) в октябре 2014 г. Данный стандарт находится в открытом доступе [2].

В июне 2016 г. стандарт был признан Международной организацией по стандартизации (ISO). MQTT Version 3.1.1 был зарегистрирован техническим комитетом по информационным технологиям ISO (JTC1) под номером ISO/IEC 20922 [3].

**ТЕРМИНОЛОГИЯ**

В переводах Стандарта используются различные определения для ключевых терминов, поэтому я ввел свои определения для своего удобства прочтения и понимания моего перевода, что не заменяет уже более-менее сложившуюся терминологию.

**MQTT-сообщение (Message)**

Информационный байтовый поток, сформированный и передаваемый в соответствии с правилами MQTT в определенной последовательности. Состоит из ***фиксированной части заголовка***, а также в определенных типах MQTT-сообщений может дополнительно содержать ***переменную часть заголовка*** и ***данные (payload),*** состоящих из одной и более строк в кодировке UTF-8, которые, однако, могут быть и нулевой длины. Их формат излагается в последующих разделах.

**Издатель (Publisher)**

**Абонент (Subscriber)**

**Клиент (Client)**

Конечное устройство IoT, поддерживающих протокол MQTT, издающего или использующего MQTT-сообщения в соответствии с определенным протоколом. Может выступать в качестве Издателя или Абонента

**Сервер(Server)**

Компьютер или микроконтроллер, который обеспечивает взаимодействие Издателей и Абонентов путем выполнения протокола обмена MQTT-сообщений и использующего в качестве базового протокола TCP/IP. Может действовать самостоятельно, так и в качестве драйвера для систем визуализации и управления, использующих протокол MQTT. Может выступать в качестве Клиента для других Серверов.

**Публикация (Subscribe)**

Частный случай MQTT-сообщения, сформированный командой PUBLISH, состоящего из себя Постоянной и Переменной частей Заголовка и Данных, структура которых определяется конкретным устройством или группой устройств, связанных физически с Клиентом или Сервером. Состоят из одной и более строк в кодировке UTF-8, которые, однако, могут быть и нулевой длины.

**Тема (Topic)**

Строка в формате UTF-8, длиной не более 32767 байт, указывающая пространство или адрес, построенный, возможно, по иерархической системе, по которому будут размещаться Публикации, созданные Издателем в соответствии с условиями размещения и доступными для подписки на них Абонентами

**Таймаут (Keep Alive timer)**

Таймаут измеряется в секундах, определяет максимальный интервал между сообщениями, получаемыми от клиента. Это позволяет серверу определить, что соединение с клиентом прервано не дожидаясь более длительного таймаута от TCP/IP.

Клиент несет ответственность за отправку сообщений в пределах времени, указанного в значении Таймаута. В течение этого времени, если нет значимой информации, клиент должен послать команду PINGREQ, которую сервер подтверждает командой PINGRESP.

Если сервер не получит никаких сообщений или команд от клиента в течении времени, в полтора раза дольше заданного в Таймауте, он отсоединяет клиента как если бы клиент сам послал команду DISCONNECT. Это действие никак не влияет на подписки клиента.

Если клиент не получил команду PINGRESP в течение времении, заданного в Таймауте после издания им команды PINGREQ, он должен закрыть сессию TCP/IP.

Таймаут – это 16-битное число, задающее в секундах время. Фактическое значение зависит от конкретного приложения, но обычно оно составляет несколько минут. Максимальное значение не должно превышать 18 часов. Значение 0 означает, что клиент не должен отключаться.

**Качество доставки (QoS Level – Quality of Service level)**

В протоколе MQTT обмен Публикациями производится в соответствии с установленным уровнем качества.

***QoS level 0: At most once delivery – не более одной***

Публикация доставляется наиболее эффективно для базового протокола TCP/IP без подтверждения средствами MQTT. Издатель не требует, чтобы сообщение было гарантировано доставлено Абонентам. Публикация может и не дойти, но это не это не считается критичным. Например, при постоянном мониторинге температуры поеря единичного измерения не изменит картины в целом.

Пример

|  |  |  |
| --- | --- | --- |
| **Клиент**  **ИЗДАТЕЛЬ** | **Сервер** | **Клиент**  **АБОНЕНТ** |
| --- PUBLISH --🡪 |  |  |
|  | --- PUBLISH --🡪 | Публикация доставлена |

***QoS level 1: At last once delivery – хотя бы одна***

Получение Публикации с подтверждением. Если возникло непредвиденное отключение или после определенного времени, заданного в параметере Keep Alive Timer, не поступает ожидаемое подтверждение, отправитель повторно посылает Публикацию с параметром DUP=1, пока не получит подтверждение от получателя. Команды SUBSCRIBE и UNSUBSCRIBE используют QoS level =1.

Команды с этим качеством доставки в заголовке имеют MessageID.

Пример команды PUBLISH, когда нет ожидаемого ответа

|  |  |  |
| --- | --- | --- |
| **Клиент**  **ИЗДАТЕЛЬ** | **Сервер** | **Клиент**  **АБОНЕНТ** |
| Сохранение Публикации с MessageID = X |  |  |
| DUP = 0  --- PUBLISH --🡪 | Публикация доставлена |  |
|  | DUP = 0  --- PUBLISH --🡪 | Публикация доставлена |
|  | Потеряно соединение |  |
| DUP = 1  --- PUBLISH --🡪 | Доставлена копия Публикации |  |
|  | DUP = 1  --- PUBLISH --🡪 | Доставлена копия Публикации |
| Подтверждение доставки | 🡨-PUBACK --- |  |
| Удаление Публикации с MessageID = X |  |  |

***QoS level 2: Exactly one delivery – гарантированно один***

Обеспечивает исключение доставки в приложение дублированных сообщений. Применяется, когда нужно исключить любые потери и дублирование информации. Это увеличивает нагрузку трафика сети, но важно, когда от полученного сообщения срабатывает сигнализация и обеспечивается вызов экстренных служб.

Команды с этим качеством доставки в заголовке имеют MessageID.

Пример, когда неустойчивая связь Сервера с Абонентом

|  |  |  |
| --- | --- | --- |
| **Клиент**  **ИЗДАТЕЛЬ** | **Сервер** | **Клиент**  **АБОНЕНТ** |
| Сохранение Публикации с MessageID = X |  |  |
| DUP = 0  --- PUBLISH --🡪 | Публикация доставлена Сохранение Message ID |  |
|  | Потеря связи, нет своевременного ответа |  |
| DUP = 1  --- PUBLISH --🡪 | Публикация доставлена повторно |  |
| Готовность к публикации | 🡨- PUBREC --- |  |
| DUP = 0  --- PUBREL -🡪 | Можно публиковать |  |
|  | DUP = 0  --- PUBLISH --🡪 | Публикация доставлена |
|  | Потеря связи, нет своевременного ответа |  |
|  | DUP = 1  --- PUBLISH --🡪 | Публикация доставлена повторно |
|  | Готовность к публикации | 🡨- PUBREC --- |
|  | DUP = 0  --- PUBREL -🡪 | Можно использовать Публикацию |
|  | Потеря связи, нет своевременного ответа |  |
|  | DUP = 1  --- PUBLISH --🡪 | Публикация доставлена повторно |
|  | DUP = 1  --- PUBREL -🡪 | Можно использовать Публикацию, если она до этого еще не была использована |
|  | Публикация завершена | 🡨- PUBCOMP --- |
| Публикация завершена | 🡨- PUBCOMP --- |  |
| Удаление Публикации с MessageID = X | Удалить MessageID |  |

***Допущения для QoS level 1 и 2***

В любой сети устройства могут терять связь друг с другом по разным причинам. Когда такое случается, одна из сторон протокола может не знать, что что-то случилось с другой стороной. В таком случае необходимо предусмотреть механизмы контроля надежности устройств и сетей, участвующих в доставке сообщений.

MQTT предполагает, что Клиент и Сервер как устройства надежны и проблемы могут возникнуть только на уровне ненадежной связи. Если потеряна связь с Клиентом, это, как правило, нештатная ситуация. Возможность восстановить информацию с устройство низка. Некоторые устройства имеют область долговременного хранения, например, Flash-ПЗУ. Предоставление более надежного хранилища на клиентском устройстве защищает наиболее важную информацию на момент непредвиденных ситуаций.

Помимо базовых сбоев бывают более сложные ситуации, что приводит к разнообразным сценариям которые протокол MQTT не сможет обработать.

***Повторные попытки доставки информации***

Не смотря на то, что обычно базовый протокол TCP/IP обеспечивает доставку MQTT-сообщений, существуют ситуации, когда они теряются и не обрабатываются протоколом MQTT. В случае, когда MQTT предполагает получение подтверждения действий ( QoS level > 0 для команд PUBLISH, PUBREL, SUBSCRIBE, UNSUBSCRIBE) и подтверждение не поступила за разумный период ожидания, отправитель может повторить доставку MQTT-сообщения. При этом отправитель помечает MQTT-сообщение как повторное, установив флаг DUP = 1.

Разумное время ожидания (таймаут) – параметр настраиваемый. Однако необходимо учитывать пограничную ситуацию, когда MQTT-сообщение находится в процессе доставки, а таймаут уже наступил. Например, отправка большого MQTT-сообщения по медленной сети займет значительно больше времени, чем короткое по быстрой сети. Поэтому желательно корректировать время таймаута после нескольких неудачных попыток.

Когда связь восстановлена, а информация о предыдущей сессии не очищена, рекомендуется Клиенту и Серверу повторить доставку ранее пересланных MQTT-сообщений. Для Сервера это более желательная логика, чем для Клиента.

***Последовательность отправки сообщений***

Последовательность отправки MQTT-сообщений может зависеть от целого ряда факторов, включая количество одновременно посланных команд MQTT-сообщений, клиент допускает параллельную обработку или нет и т.п. Для настоящего обсуждения предполагается последовательная обработка транзакций.

Реализация MQTT предполагает предоставление гарантий того, что каждая транзакция будет обработана до полного своего завершения. Например, два запроса с QoS level = 2, команды PUBREL должны быть посланы в той же последовательности, как и изначальные PUBLISH команды.

Пример

|  |  |
| --- | --- |
| Клиент | Сервер |
| PUBLISH 1 -🡪  PUBLISH 2 -🡪  PUBLISH 3 -🡪 |  |
|  | 🡨- PUBREC 1  🡨- PUBREC 2 |
| PUBREL 1 -🡪 |  |
|  | 🡨- PUBREC 3 |
| PUBREL 2 -🡪 |  |
|  | 🡨- PUBCOMP 1 |
| PUBREL 3 -🡪 |  |
|  | 🡨- PUBCOMP 2  🡨- PUBCOMP 3 |

Количество одновременно обрабатываемых MQTT-сообщений также влияет а тип предоставляемых гарантий:

* Если каждая команда обрабатывается последовательно до своего завершения, можно гарантировать, что все MQTT-сообщения будут установлены в порядке, котором были предоставлены.
* Если одновременно обрабатывается более одной команды, последовательность следования MQTT-сообщений могут быть гарантированы только соответствующим QoS level

**Формат MQTT-сообщения**

***Фиксированная часть заголовка***

Все данные высшего порядка расположены перед данными низшего порядка. В 16-битном слове сначала расположен старший байт (Most Significant Byte – MSB), затем младший ( Least Significant Byte – LSB).

Фиксированная часть заголовка состоит из 2 байт

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Бит** | **7** | **6** | **5** | **4** | | **3** | **2** | | **1** | | **0** |
| Байт 1 | Тип MQTT-сообщения | | | | DUP | | | уровень QoS | | RETAIN | |
| Байт 1 | Remaining Length | | | | | | | | | | |

***БАЙТ 1***

***ПОЗИЦИЯ: Байт 1, Биты 7-4 – Тип MQTT-сообщения (Message type)***

Представлен в виде 4-битового положительного значения в соответствии с таблицей

|  |  |  |
| --- | --- | --- |
| Название | Код | Описание |
| Reserved | 0 | резерв |
| **CONNECT** | 1 | Запрос Клиента на ПОДКЛЮЧЕНИЕ к Серверу |
| **CONNAK** | 2 | Подтверждение ПОДКЛЮЧЕНИЯ Сервера |
| **PUBLISH** | 3 | ОПУБЛИКОВАТЬ |
| **PUBACK** | 4 | ПУБЛИКАЦИЯ подтверждена |
| **PUBREC** | 5 | ПУБЛИКАЦИЯ получена (заверение в доставке 1) |
| **PUBREL** | 6 | ПУБЛИКАЦИЯ доставлена (заверение в доставке 2) |
| **PUBCOMP** | 7 | ПУБЛИКАЦИЯ размещена (заверение в доставке 3) |
| **SUBSCRIBE** | 8 | ПОДПИСАТЬСЯ |
| **SUBACK** | 9 | Подтверждение ПОДПИСКИ |
| **UNSUBSCRIBE** | 10 | ОТПИСАТЬСЯ |
| **UNSUBACK** | 11 | Подтверждение ОТПИСКИ |
| **PINREQ** | 12 | Запрос ПИНГА |
| **PINRESP** | 13 | Подтверждение ПИНГА |
| **DISCONNECT** | 14 | Извещение Клиента о плановом ОТКЛЮЧЕНИИ |
| Reserved | 15 | резерв |

***ПОЗИЦИЯ: БАЙТ 1, бит 3 – флаг повторного сообщения DUP (Duplicate)***

Этот флаг указывает получателю, что полученное MQTT-сообщение передается повторно и, возможно, уже было получено им ранее. Этот флаг играет важную роль при передаче информации по ненадежным каналам, где возможна потеря сигнала.

Флаг устанавливается в 1 когда Клиент или Сервер повторно пересылает MQTT-сообщения типа PUBLISH, PUBREL, SUBSCRIBE и UNSUBSCRBE. Применяется, когда значение QoS больше 0. Когда DUP установлен в 1, в переменной части заголовка содержится MessageID.

***ПОЗТЦИЯ: БАЙТ 1, бит 2-1 – Quality of Service (QoS level)***

Этот флаг определяет уровень качество доставки Публикаций.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Код | Бит2 | Бит1 | Описание |  |
| 0 | 0 | 0 | At most once | Выстрелить и забыть |
| 1 | 0 | 1 | At least once | Подтверждение доставки |
| 2 | 1 | 0 | Exactly once | Гарантированная доставка |
| 3 | 1 | 1 | резерв |  |

***ПОЗИЦИЯ: БАЙТ 1, бит 0 – флаг сохранения RETAIN***

Этот флаг используется только в MQTT-сообщениях типа PUBLISH (Публикация). Если при передаче Публикации на Сервер флаг установлен, она будет сохранена Сервером после доставки существующим Абонентам.

Если флаг установлен в 1, то при появляении нового Абонента все Публикации из Тем, на которые новый Абонент подписывается, будут переданы новому Абоненту, что позволяет новому Абоненту мгновенно получить прежде сохраненные в Теме Публикации. Если в Тему ничего не поступало, то новому Абоненту ничего не отправляется.

Когда Сервер запрашивает PUBLISH у Клиента как результат существующей подписки, и запрошенная Публикация доставлена, флаг RETAIN должен быть сброшен вне зависимости от первоначального PUBLISH. Это позволяет Клиенту различать сообщения по Теме, которые были получены, тк были ранее сохранены от «живых»

Сохраненные Публикации должны храниться до перезагрузки Сервера.

Сервер может удалить сохраненные Публикации, если получает сообщение с Данными нулевой длины и установлен флаг RETAIN на эту Тему

***ПОЗИЦИЯ: БАЙТ 2 – оставшаяся длина (Remaining Lenth)***

Представляет количество байтов, оставшихся в текущей Публикации, включая информацию в *переменной части заголовка* и *данных*.

***Переменная часть заголовка***

Некоторые типы MQTT-сообщений также содержат компонент ПЕРЕМЕННУЮ ЧАСТЬ ЗАГОЛОВОКА (variable header). Он находится между ФИКСИРОВАННОЙ ЧАСТЬЮ ЗАГОЛОВКА (fixed header) и ДАННЫМИ (payload).

Поле “ОСТАВШАЯСЯ ДЛИНА” (Remaining Length), расположенный после ФИКСИРОВАННОГО ЗАГОЛОВКА, не является частью ПЕРЕМЕННОГО ЗАГОЛОВКА и содержит информацию о количестве байт только в ПЕРЕМЕННОГО ЗАГОЛОВКА и ДАННЫХ, а его значение расссчитывается, как это было описано выше в описании ФИКСИРОВАННОГО ЗАГОЛОВКА.

**В MQTT-сообщениях типа** PUBLISH, PUBACK, PUBREC, PUBREL, PUBCOMP, SUBSCRIBE, SUBACK, UNSUBSCRIBE, UNSUBACK **присутствует Идентификатор (Message ID)**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|  | Message ID MSB | | | | | | | |
|  | Message ID LSB | | | | | | | |

Message ID существует только в сообщениях 1 и 2 уровня QoS (Quality of Service – качества обслуживания).

Message ID – 16-битное неотрицательное число, соответствующее коду команды MQTT и, как правило, после использования увеличивается на единицу, назначая следующую команду в соответствии с протоколом, но это зависит от исполняемой задачи.

Message ID клиента и сервера не совпадают, клиент можт издать команду PUBLISH с Message ID = 1 одновременно получив другую команду PUBLISH с таким же Message ID

Message ID = 0 зарезервировано и может вызвать ошибку

***Данные***

Следующие команды MQTT содержат в себе Данные

- CONNECT – Данные содержат одну или более строк в кодировке UTF-8. Они определяют уникальный идентификатор клиента, Will topic, Will message, User Name and User Password. Наличие этих строк определяется флагами, значения которых определены в переменной части заголовка.

- SUBSСRIBE – Данные содержит одну или более строк в кодировке UTF-8, определяющих список Тем, на которые клиент может подписаться и уровень QoS

- SUBNAK Данные содержит список предоставленных уровней QoS. Предоставленные уровни QoS соответсвуют Темам, на которые клиент подписался командой SUBSCRIBE

- PUBLISH Данные содержат содержит только данный конкретного приложения. Эта информация обрабатывается исключительно как BLOB. Являются основной информационной составляющей Публикации.

Если вы планируете применить сжатие данных в Payload, необходимо в приложении заложить информацию для обработки деталей сжатия. Эта информация специфична для конкретного приложения и не может быть размещена в постоянной или фиксированной части заголовка.

**3. ТИПЫ MQTT-сообщений (Команды)**

**3.1. Команда CONNECT – запрос клиента на соединение с сервером**

После осуществления соединение по протоколу TCP/IP должно быть установлено соединение по протоколу MQTT с использованием команды CONNECT

***Фиксированная часть заголовка***

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| byte 1 | Message type (1) | | | | DUP flag | QoS level | | RETAIN |
|  | 0 | 0 | 0 | 1 | x | x | x | x |
| byte 2 | Remaining Length (2) | | | | | | | |
|  |  |  |  |  |  |  |  |  |

QoS level

Не используется

DUP flag

Не используется

RETAIN flag

Не используется

Remaining Length = 12 байтам, состоящим из переменной части заголовка и Payload.

***Переменная часть заголовка***

Пример переменной части заголовка со значением Keep Alive timer 10 секунд (0x000A)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Description | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Protocol Name | |  | | | | | | | |
| byte 1 | Length MSB (0) | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| byte 2 | Length LSB (6) | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| byte 3 | 'M' | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 |
| byte 4 | 'Q' | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 |
| byte 5 | 'I' | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |
| byte 6 | 's' | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 |
| byte 7 | 'd' | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 |
| byte 8 | 'p' | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| Protocol Version Number | |  | | | | | | | |
| byte 9 | Version (3) | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| Connect Flags | |  | | | | | | | |
| byte 10 | User Name flag(1)  Password flag (1)  Will RETAIN (0)  Will QoS (01)  Will flag (1)  Clean Session (1) | 1 | 1 | 0 | 0 | 1 | 1 | 1 | х |
| Keep Alive timer | |  | | | | | | | |
| byte 11 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| byte 12 | | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |

***ИМЯ ПРОТОКОЛА (Protocol Name)***

Имя протокола - строка в кодировке UTF-8, представляющую название протокола “MQTsdp” заглавными буквами.

***ВЕРСИЯ ПРОТОКОЛА*** ***(Protocol Version Number)***

В беззнаковом байте содержится текущая ревизия уровня протокола, используемого Клиентом. В настоящее аремя это значение 0х03, как показано в таблице

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Бит** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
|  | Версия протокола | | | | | | | |
|  | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |

***БАЙТ ФЛАГОВ СОЕДИНЕНИЯ (Connection flags )***

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Бит | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|  | User Name Flag | Password Flag | Will Retain | Will QoS | | Will Flag | Clean Session | Reserved |
|  | x | x | x | x | x | x |  | x |

Бит 0 в данной версии протокола не используется и зарезервирован для будущего.

***ПОЗИЦИЯ: Бит1 Байта Connection flags – Очистить сессию (Clean Session)***

Если не сброшен в 0, Сервер должен хранить подписки Клиента после его отключения. Это означает, что необходимо продолжать доставлять сообщения уровня QoS1 и QuS2 подписанных тем после отключения Клиента. Сервер также должен поддерживать состояние сообщения, находящегося в процессе доставки на момент разрыва связи. Информация должна храниться до следующего подключения Клиента.

Если установлен в 1, Сервер должен всю ранее полученную информацию о Клиенте и рассматривать как “чистое” подключение. Сервер обязан также отменить все состояния, когда Клиент отключается.

Как правило, Клиент будет неизменно работать либо в одном, либо в другом режиме. Выбор будет зависеть от конкретной задачи. В “чистой” сессии Клиент не получит при подключении устаревшую информацию и его необходимо будет подписывать каждый раз, когда он подключается. В протвном случае при подключении Клиент увидит сообщения уровня QoS1 и QoS2, которые были опубликованы за время его отсутствия. Сообщения уровня QoS0 никогда не сохраняются после публикации.

Ранее этот флаг назывался «Чистый Старт», но впоследствии уточнен его смысл, в связи с тем, что он применим не только к первоначальному запуску, но и ко всем последующим сессиям.

Сервер может предоставить административный механизм для очистки информации о Клиенте, когда известно, что Клиент больше не вернется.

***ПОЗИЦИЯ Бит2 Байта Connection flags - Will флаг***

Определяет, что сообщение публикуется сервером от имени клиента при возникновении ошибки ввода/вывода во время общения с клиентом, или клиент не укладывается в установленное время сохранения. Отправка Will-сообщения не исполняется, если сервер получил сообщение DISCONNECT от клиента.

Если Will установлен, Will QoS и Will Retain поля имеют значение. Поля Will Topic и Will Message должны быть представлены в блоке данных playload.

***ПОЗИЦИЯ Бит 4 и 3 Байта Connection flags - Will QoS флаг***

Имеет значение при выставленном флаге Will.

Подключающийся клиент задает уровень QoS (0x00 или 0x01 или 0x02) для Will message, которое будет издано в случае непредвиденного отключения клиента. Содержание Will message задается в блоке данных payload команды CONNECT.

***ПОЗИЦИЯ Бит 5 Байта Connection flags - Will Remain флаг***

Имеет значение при выставленном флаге Will.

Определяет необходимость сохранение Will message, опубликованное на сервере от имнени клиента в случае непредвиденного отключения.

Если флаг сброшен, необходимости сохранения Will message нет.

***ПОЗИЦИЯ Бит 6 и 7 Байта Connection flags - User Name и User Password флаги***

Подсоединяющийся клиент может определить имя пользователя и пароль, установленные флаги указывают, что в блоке данных payload обязательно должна содержатся информация об имени пользователя и, возможно, пароле, заданным клиентом.

Флаг User Password не может быть установлен, если не установлен флаг User Name

***ПОЗИЦИЯ 2 БАЙТА Keep Alive timer (Таймаут)***

Смысл Таймера описан ранее

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|  | Keep Alive time MSB | | | | | | | |
|  | Keep Alive time LSB | | | | | | | |

***Данные***

Данные команды CONNECT содержит одну или более строк в кодировке UTF-8 исходя из флагов, выставленных в переменной части заголовка. Строки, если определены, должны появляться в следующем порядке

**Client Identifier**

Первая строка в кодировке UTF-8. Client ID – идентификатор клиента длиной до 23 символов, уникально идентифицирующих клиента для сервера. Он должен быть уникальным для всех клиентов, подключенных к одному серверу и является ключевым в обработке сообщений уровня QoS1 и QoS2. Если размер Client Identifier более 23 символов, команда CONNAK от сервера вернет ошибку (2) Code Rejected:Identifier Rejected

**Will Topic**

Cтрока в кодировке UTF-8. Если установлен Will Flag, то в теме, указанной в Will Topic будет опубликовано Will Message. Уровень QoS определен флагом Will QoS и статус RETAIN определен флагом Will RETAIN в переменной части заголовка

**Will Message**

Cтрока в кодировке UTF-8. Если установлен Will Flag, то содержит текст, которй будет передан в Will Topic в случае непредвиденного разрыва связи с клиентом. Может быть нулевой длины

Не смотря на то, что в команде CONNECT Will Message в кодировке UTF-8, публикация в Will Topic должна происходить в формате 7-битных ASCII символов без указания в начале строки 2 байт длины сообщения

**User Name**

Cтрока в кодировке UTF-8. Если установлен User Name flag – идентифицирует имя присоединяющегося клиента, которое будет использовано для идентификации. Рекомендуемая длина не более 12 символов, но не строго.

Обратите внимание на то, что для совместимости с оригинальной спецификацией MQTT V3, значение оставшейся длины поля из фиксированной части заголовка имеет приоритет над флагом User name. Реализация сервера должна предусматривать возможность того, что в имени пользователя установлен флаг, но отсутствует строка имени пользователя. Это допустимо, и соединения должно быть разрешено продолжить.

**Password**

Cтрока в кодировке UTF-8. Если установленPassword flag – идентифицирует пароль присоединяющегося клиента, которое будет использовано для идентификации. Рекомендуемая длина не более 12 символов, но не строго.

Обратите внимание на то, что для совместимости с оригинальной спецификацией MQTT V3, значение оставшейся длины поля из фиксированной части заголовка имеет приоритет над флагом Password. Реализация сервера должна предусматривать возможность того, что в Password установлен флаг, но отсутствует строка пароля. Это допустимо, и соединения должно быть разрешено продолжить.

***Ожидаемая реакция***

Сервер посылает команду CONNACK в ответ на команду CONNECT от клиента.

Если сервер не получил команду CONNECT за разумное время после установки соединения по протоколу TCP/IP, сервер завершает соединение.

Если клиент не получил команду CONNACK от сервера за разумное время, клиент завершает TCP/IP соединение и повторяет запрос CONNECT.

В обоих сценариях термин «разумное время» зависит от особенностей приложения и структуры коммуникаций.

Если на сервере уже есть клиент с таким же ClientID, сервер прекращает соединение и не соединяется, пока не отсоединится предыдущий клиент с таким ClientID.

Если клиент посылает неверную команду CONNECT, сервер прекращает соединение. Это может произойти в случае указания неверного Name of Protocol или Protocol Version Number, определенных в фиксированной части заголовка сообщения, о чем сервер сообщает, вернув в команде CONNACK код “(1) Connection Refused: unnacceptible protocol version”

**3.2. Команда CONNACK – Подтверждения запроса на соединение**

Команда CONNACK посылается Сервером Клиенту в ответ на команду CONNECT, посланной Клиентом.

Фиксированная часть заголовка

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| byte 1 | Message type (2) | | | | DUP flag | QoS level | | RETAIN |
|  | 0 | 0 | 1 | 0 | x | x | x | x |
| byte 2 | Remaining Length (2) | | | | | | | |
|  | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

QoS level

Не используется

DUP flag

Не используется

RETAIN flag

Не используется

Remaining Length = 2 байтам, состоящим из переменной части заголовка, определяющих размер переменной части сообщения.

***Переменная часть заголовка***

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Description | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Topic Name Compression Response | | | | | | | | | |
| byte 1 | Зарезервирован | x | x | x | x | x | x | x | x |
| Connect Return Code | | | | | | | | | |
| byte 2 | Код возврата Connect |  |  |  |  |  |  |  |  |

Код возврата 0 обычно означает успешное соединение, остальные значения приведены в таблице

Коды возврата на запрос клиента командой CONNECT

|  |  |  |
| --- | --- | --- |
| Код | HEX | Значение |
| 0 | 0x00 | Connection Accepted: соединение подтверждено |
| 1 | 0x01 | Connection Refused: недопустимая версия протокола |
| 2 | 0x02 | Connection Refused: идентификатор отклонен |
| 3 | 0x03 | Connection Refused: сервер недоступен |
| 4 | 0x04 | Connection Refused: неверное имя пользователя или пароль |
| 5 | 0x05 | Connection Refused: не авторизован |
| 6-255 |  | Зарезервироваровано |

Код 2 Connection Refused: идентификатор отклонен устанавливается, если ClientID нулевой или более 23 байт

***Данные***

Отсутствуют

**3.3. Команда PUBLISH – запрос на Публикацию по Теме**

Посылается Издателем Серверу с указанием Темы, после чего полученная Публикация пересылается Абонентам, подписанным на эту Тему, командой PUBLISH с тем же содержанием от Сервера в соответствии с установленным уровнем качества обслуживания QoS level.

***Фиксированная часть загоовка***

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| byte 1 | Message type (3) | | | | DUP flag | QoS level | | RETAIN |
|  | 0 | 0 | 1 | 1 | x | x | x | x |
| byte 2 | Remaining Length (2) | | | | | | | |
|  |  |  |  |  |  |  |  |  |

QoS level = 1

Уровень обслуживания сообщений равен 1

DUP flag = 0

Указывает на то, что Публикация делается в первый раз

RETAIN flag = 0

Означает отсутствие необходимости сохранять сообщение. См.описание RETAIN

Remaining Lenth

Указывает значение количества байт в переменной части заголовка и информационной части сообщения. Может содержать несколько списков.

***Переменная часть заголовка***

Topic Name (Название Темы)

Представлено в переменной части заголовка команды PUBLISH и является ключом, который идентифицирует информационный канал, на котором публикуется информация и данные. Подписчики (абоненты) используют ключ для для идентификации информационных каналов, по которым хотят получать опубликованную информацию.

Topic name – это строка в кодировке UTF-8 длиной не более 32767 символов. Для получения дополнительной информации см. Приложение MQTT и UTF.

Строки не должны содержать подстановочные знаки, которые могут задаваться Абонентом для получения информации по разным подтемам, те название Темы должно быть точным.

Например

|  |  |
| --- | --- |
| Поле | Значение |
| Topic Name: | "a/b" |
| QoS level | 1 |
| Message ID: | 10 |

В этом случае вид переменной части заголовка будет следующим

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Description | 7 | 6 | | 5 | | 4 | 3 | 2 | 1 | 0 |
|  | Topic Name | | | | | | | | | | |
| byte 1 | Length MSB (0) | 0 | | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |
| byte 1 | Lenth LSB (3) | 0 | | 0 | | 0 | 0 | 0 | 0 | 1 | 1 |
| byte 1 | !a! (0x61) | 0 | | 1 | | 1 | 0 | 0 | 0 | 0 | 1 |
| byte 1 | !/! (0x2F) | 0 | | 0 | | 1 | 0 | 1 | 1 | 1 | 1 |
| byte 1 | ‘b’ (0x62) | 0 | | 1 | | 1 | 0 | 0 | 0 | 1 | 0 |
|  | Message ID | | | | | | | | | | |
| byte 1 | Message ID MSB (0) | 0 | | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |
| byte 1 | Message ID LSB (10) | 0 | | 0 | | 0 | 0 | 1 | 0 | 1 | 0 |

***Данные***

Содержит публикуемые данные. Содержание и формат определяется конкретным приложением. В некоторых случаях, определянных приложением, может быть нулевой длины.

***Ожидаемая реакция***

Зависят от установленного значения QoS

|  |  |  |
| --- | --- | --- |
| QoS Level | Ожидаемая команда | Действие |
| QoS 0 | None | Сервер открывает сообщение всем Абонентам. Издателю ничего не возвращается. |
| QoS 1 | PUBACK | Сервер открывает сообщение всем Абонентам, зарегистрировав его в постоянном хранилище и возвращает Издателю команду PUBACK |
| QoS 2 | PUBREC | Сервер откладывет сообщение, оно пока не доступно всем Абонентам, зарегистрировав его в постоянном хранилище и возвращает Издателю команду PUBREC |

Для QoS level 0 и QoS level 1 после получения Сервером команды PUBLISH от Издателя, Сервер дублирует ее всем Абонентам, подписанным на получение Публикаций по Теме, указанной в этой команде PUBLISH.

Для QoS2 Абонент не открывает доступ к использованию Публикации, тк предварительно должен подтвердить свою готовность.

**3.4. Команда PUBACK – подтверждение публикации**

Это вторая команда в протоколе с уровнем обслуживания QoS level 1.

Посылается Сервером Издателю в ответ на команду PUBLISH от Издателя или посылается Абонентом Серверу в ответ на команду PUBRLISH от Сервера.

***Фиксированная часть заголовка***

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| byte 1 | Message type (4) | | | | DUP flag | QoS level | | RETAIN |
|  | 0 | 1 | 0 | 0 | x | x | x | x |
| byte 2 | Remaining Length (2) | | | | | | | |
|  | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

QoS level

Не используется

DUP flag

Не используется

RETAIN flag

Не используется

Remaining Lenth = 2

Указывает значение количества байт в переменной части заголовка. Может содержать несколько списков.

***Переменная часть заголовка***

Содержит Message ID подтверждаемой Побликации

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Байт 1 | Message ID MSB | | | | | | | |
| Байт 2 | Message ID LSB | | | | | | | |

***Данные***

Отсутствуют

***Ожидаемая реакция***

Когда Издатель получает команду PUBACK, означающую подтверждение того, что Публикация доставлено и сохранена на Сервере, он может стереть исходную Публикацию.

**3.5. Команда PUBREC - заверение в готовности получить Публикацию**

Это вторая команда в протоколе с уровнем обслуживания QoS level 2.

Посылается Сервером Издателю в ответ на команду PUBLISH от Издателя или посылается Абонентом Серверу в ответ на команду PUBRLISH от Сервера.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| byte 1 | Message type (5) | | | | DUP flag | QoS level | | RETAIN |
|  | 0 | 1 | 0 | 1 | x | x | x | x |
| byte 2 | Remaining Length (2) | | | | | | | |
|  | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

QoS level

Не используется

DUP flag

Не используется

RETAIN flag

Не используется

Remaining Lenth = 2

Содержит значение количества байт в переменной части заголовка. Может содержать несколько списков.

***Переменная часть заголовка***

Содержит Message ID подтверждаемой Побликации

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Байт 1 | Message ID MSB | | | | | | | |
| Байт 2 | Message ID LSB | | | | | | | |

***Данные***

Отсутствуют

***Ожидаемая реакция***

После получения команды PUBREC возвращается команда PUBREL с тем же MessageID, который указан в команде PUBREC

**3.6. Команда PUBREL – заверение в получении Публикации**

Это третья команда в протоколе MQTT с уровнем обслуживания QoS level 2.

Посылается Издателем Серверу в ответ на команду PUBREC от Сервера или посылается Сервером Абоненту в ответ на команду PUBREC от Абонента.

***Фиксированная часть загловка***

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| byte 1 | Message type (6) | | | | DUP flag | QoS level | | RETAIN |
|  | 0 | 1 | 1 | 0 | x | x | x | x |
| byte 2 | Remaining Length (2) | | | | | | | |
|  | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

QoS level

Не используется

DUP flag

Не используется

RETAIN flag

Не используется

Remaining Lenth = 2

Содержит значение количества байт в переменной части заголовка. Может содержать несколько списков.

***Переменная часть заголовка***

Содержит Message ID подтверждаемой Побликации

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Байт 1 | Message ID MSB | | | | | | | |
| Байт 2 | Message ID LSB | | | | | | | |

***Данные***

Отсутствуют

***Ожидаемая реакция***

Когда Сервер получил команду PUBREL от Издателя, Сервер открывает доступ к содержимому Публикации всем подписанным на Тему Абонентам и возвращает команду PUBCOMP Издателю.

Абонент открывает доступ к Публикации для использования только после получения команды PUBREL от Сервера и возвращает Серверу команду PUBCOMP

**3.7. Команда PUBCOMP – подтверждение успешного размещения Публикации**

Это четвертая команда в протоколе MQTT с уровнем обслуживания QoS level 2.

Посылается Сервером Издателю в ответ на команду PUBREL от Издателя или посылается Абонентом Серверу в ответ на команду PUBREL от Сервера.

***Фиксированная часть загловка***

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| byte 1 | Message type (7) | | | | DUP flag | QoS level | | RETAIN |
|  | 0 | 1 | 1 | 1 | x | x | x | x |
| byte 2 | Remaining Length (2) | | | | | | | |
|  | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

QoS level

Не используется

DUP flag

Не используется

RETAIN flag

Не используется

Remaining Lenth = 2

Содержит значение количества байт в переменной части заголовка. Может содержать несколько списков.

***Переменная часть заголовка***

Содержит Message ID подтверждаемой Публикации

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Байт 1 | Message ID MSB | | | | | | | |
| Байт 2 | Message ID LSB | | | | | | | |

***Данные***

Отсутствуют

***Ожидаемая реакция***

Когда Издатель получает команду PUBACK, означающую подтверждение того, что Публикация доставлено и сохранена на Сервере, он может стереть исходную Публикацию.

Когда Сервер получает команду PUBACK, означающую подтверждение того, что Публикация доставлено и сохранена Абоненту, он может стереть исходную Публикацию.

**3.8. Команда SUBSCRIBE – подписаться на Тему**

Команда SUBSCRIBE позволяет Абоненту зафксировать интерес к одной или более Тем, декларированных на Сервере. Публикации, размещенные в этих Темах, будут доставляться Абоненту Сервером в соответствии с установленным уровнем качество обслуживания QoS level командой PUBLISH, иными словами Сервер по отношению к Абоненту является Издателем.

***Фиксированная часть заголовка***

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| byte 1 | Message type (8) | | | | DUP flag | QoS level | | RETAIN |
|  | 1 | 0 | 0 | 0 | 0 | 0 | 1 | x |
| byte 2 | Remaining Length (2) | | | | | | | |
|  | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

QoS level = 1

Требует подтверждения от Абонента на все подписки на Темы, указанные в запросе, ответной командой SUBACK. Обрабатывается аналогично команде PUBLISH

DUP flag = 0

Указывает на то, что Публикация делается в первый раз

RETAIN flag

Не используется

Remaining Lenth = 2

Указывает значение количества байт в переменной части заголовка. Может содержать несколько списков.

***Переменная часть заголовка***

Содержит Message ID Публикации в соответствии с условиями качества обслуживания QoS level.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Байт 1 | Message ID MSB | | | | | | | |
| Байт 2 | Message ID LSB | | | | | | | |

***Данные***

Содержит перечень названий Тем, оформленных как строковые потоки в кодировке UTF-8, на которые Абонент выражает намерение подписаться с учетом возможно присутствующих подстановочных знаков, группирующие Темы по определенным признакам в соответствии с правилами подстановки MQTT и соответствующий каждой из этих тем уровень качества доставки QoS level, размещаемый в байте, расположенном после строки с названием Темы и использующим 2 младших бита в этом байте. Если биты установлены неверно, сессия с Абонентом прекращается Сервером.

Например, 2 темы «a/b» и «c/d»

|  |  |
| --- | --- |
| Поле | Значение |
| Название темы: | "a/b" |
| Запрашиваемый QoS level | 1 |
| Название темы: | "c/d" |
| Запрашиваемый QoS level | 2 |

Будут переданы в формате

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Description | 7 | 6 | | 5 | | 4 | 3 | 2 | 1 | 0 |
|  | Topic Name ("a/b") | | | | | | | | | | |
| byte 1 | Length MSB (0) | 0 | | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |
| byte 2 | Lenqth LSB (3) | 0 | | 0 | | 0 | 0 | 0 | 0 | 1 | 1 |
| byte 3 | ‘a’ (0x61) | 0 | | 1 | | 1 | 0 | 0 | 0 | 0 | 1 |
| byte 4 | ‘/’ (0x2F) | 0 | | 0 | | 1 | 0 | 1 | 1 | 1 | 1 |
| byte 5 | ‘b’ (0x62) | 0 | | 1 | | 1 | 0 | 0 | 0 | 1 | 0 |
|  | Requested QoS level | | | | | | | | | | |
| byte 6 | (1) | x | | x | | x | x | x | x | 0 | 1 |
|  | Topic Name ("c/d") | | | | | | | | | | |
| byte 7 | Length MSB (0) | 0 | | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |
| byte 8 | Lenqth LSB (3) | 0 | | 0 | | 0 | 0 | 0 | 0 | 1 | 1 |
| byte 9 | ‘c’ (0x63) | 0 | | 1 | | 1 | 0 | 0 | 0 | 1 | 1 |
| byte 10 | ‘/’ (0x2F) | 0 | | 0 | | 1 | 0 | 1 | 1 | 1 | 1 |
| byte 10 | ‘d’ (0x64) | 0 | | 1 | | 1 | 0 | 0 | 1 | 0 | 0 |
|  | Requested QoS level | | | | | | | | | | |
| byte 12 | (2) | x | | x | | x | x | x | x | 1 | 0 |

Абонент получает Публикации с уровнем качества обслуживания QoS level не больше запрошенного с учетом QoS Level исходной Публикации. Например, если Абонент запросил QoS level = 1, а QoS level команды PUBLISH равен 0, то доставка Публикации будет QoS level = 0, а вот с QoS level команды PUBLISH равным 2 Публикация Абоненту будет доставлена с QoS level = 1

Это означает, что Издатель определяет максимально возможный для конкретной Публикации уровень качества доставки QoS level, но Абонент может его понизить для себя. Единожды установленный QoS ltevel больше не меняется

Сервер может понизить уровень качества доступа QoS level до 0 если не сможет предоставтиь надежный механизм хранения Публикаций.

***Ожидаемая реакция***

Когда Сервер получает команду SUBSCRIBE от от Абонента, Сервер в ответ высылает команду SUBACK

Сервер может начать посылать команды PUBLSIH на заявленные в подписке Темы до получения Абонентом команды SUBACK.

Обратите внимание на то, что если реализация Сервера на допускает авторизации команды SUBSCRIBE, изданной Абонентом, Сервер командой SUBASK уведомит абонента об этом.

**3.9. Команда SUBACK – подтверждение подписки**

Посылается Абоненту Сервером в качестве подтверждения, что команда SUBSCRIBE Абонента принята и размещена на Сервере.

Команда SUBACK содержит перечень подтвержденных QoS level для каждой запрошенной Темы в соответствии с перечнем, представленным командой SUBSCRIBE.

***Фиксированная часть загловка***

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| byte 1 | Message type (9) | | | | DUP flag | QoS level | | RETAIN |
|  | 1 | 0 | 0 | 1 | x | x | x | x |
| byte 2 | Remaining Length (2) | | | | | | | |

QoS level

Не используется

DUP flag

Не используется

RETAIN flag

Не используется

Remaining Lenth

Содержит значение количества байт в данных. Может содержать несколько списков.

***Переменная часть заголовка***

Содержит Message ID, полученной в команде SUBSCRIBE для его подтверждения и установки соответствия команд SUBSCRIBE и SUBACK по конктретному перечню Тем.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Байт 1 | Message ID MSB | | | | | | | |
| Байт 2 | Message ID LSB | | | | | | | |

***Данные***

Содержит список подтвержденный QoS level по Темам, заказанным в запросе SUBSCRIBE и упорядочен в соответствии с перечнем Тем, указанных в команде SUBSCRIBE. Каждый байт содержит значение соответствующего подтвержденного QoS level для соответствущей Темы. В байте используются младшие 2 бита для указания QoS level

Например, запрошена подписка на 2 Темы с QoS level = 0 для первой Темы и QoS level = 2 для второй

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | Описание | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| byte 1 | Подтвержденный QoS (0) | x | x | x | x | X | x | 0 | 0 |
| byte 2 | Подтвержденный QoS (2) | x | x | x | x | x | x | 1 | 0 |

**3.10. Команда UNSUBSCRIBE – отписаться от Подписки**

Посылается Абонентом Серверу с целью указать перечень Тем, от который Абонент собирается отказаться.

***Фиксированная часть заголовка***

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| byte 1 | Message type (10) | | | | DUP flag | QoS level | | RETAIN |
|  | 1 | 0 | 1 | 0 | 0 | 0 | 1 | x |
| byte 2 | Remaining Length (2) | | | | | | | |

QoS level = 1

Использует QoS level = 1 для подтверждения групповой отписки. Соответствующая команда UNSUBACK идентифицируется Message ID. Обрабатывается аналогично протоколу команды PUBLISH

DUP flag = 0

Указывает на то, что Публикация делается в первый раз

RETAIN flag

Не используется

Remaining Lenth = 2

Указывает значение количества байт в Данных. Может содержать несколько списков.

***Переменная часть заголовка***

Содержит Message ID Публикации в соответствии с условиями качества доставки QoS level = 1.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Байт 1 | Message ID MSB | | | | | | | |
| Байт 2 | Message ID LSB | | | | | | | |

***Данные***

Например, Абонент хочет отписаться от 2 тем «a/b» и «c/d»

|  |  |
| --- | --- |
| Поле | Значение |
| Название темы: | "a/b" |
| Название темы: | "c/d" |

Будут переданы в формате

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Description | 7 | 6 | | 5 | | 4 | 3 | 2 | 1 | 0 |
|  | Topic Name ("a/b") | | | | | | | | | | |
| byte 1 | Length MSB (0) | 0 | | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |
| byte 2 | Lenqth LSB (3) | 0 | | 0 | | 0 | 0 | 0 | 0 | 1 | 1 |
| byte 3 | ‘a’ (0x61) | 0 | | 1 | | 1 | 0 | 0 | 0 | 0 | 1 |
| byte 4 | ‘/’ (0x2F) | 0 | | 0 | | 1 | 0 | 1 | 1 | 1 | 1 |
| byte 5 | ‘b’ (0x62) | 0 | | 1 | | 1 | 0 | 0 | 0 | 1 | 0 |
|  | Topic Name ("c/d") | | | | | | | | | | |
| byte 6 | Length MSB (0) | 0 | | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |
| byte 7 | Lenqth LSB (3) | 0 | | 0 | | 0 | 0 | 0 | 0 | 1 | 1 |
| byte 8 | ‘c’ (0x63) | 0 | | 1 | | 1 | 0 | 0 | 0 | 1 | 1 |
| byte 9 | ‘/’ (0x2F) | 0 | | 0 | | 1 | 0 | 1 | 1 | 1 | 1 |
| byte 10 | ‘d’ (0x64) | 0 | | 1 | | 1 | 0 | 0 | 1 | 0 | 0 |

***Ожидаемая реакция***

Сервер отвечает командой SUBACK, подтверждая запрос на отписку, посланной Агентом командой UNSUBSCRIBE

**3.11. Команда UNSUBACK – подтверждение отписки**

Посылается Агенту Сервером с целью подтверждения запроса на отписку, посланного Агентом командой UNSUBSCRIBE

***Фиксированная часть загловка***

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| byte 1 | Message type (11) | | | | DUP flag | QoS level | | RETAIN |
|  | 1 | 0 | 1 | 1 | x | x | x | x |
| byte 2 | Remaining Length (2) | | | | | | | |
|  | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

QoS level

Не используется

DUP flag

Не используется

RETAIN flag

Не используется

Remaining Lenth = 2

Содержит значение количества байт в переменной части заголовка.

***Переменная часть заголовка***

Содержит Message ID, определенного командой UNSUBSCRIBE, требующей подтверждения исполнения.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Байт 1 | Message ID MSB | | | | | | | |
| Байт 2 | Message ID LSB | | | | | | | |

***Данные***

Отсутствуют

**3.12. Команда PINGREQ – Ping – запрос**

Дословно значает – «Ты жив?» и посылается Абонентом или Сервером, при наступлении события Keep Alive timer

***Фиксированная часть загловка***

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| byte 1 | Message type (12) | | | | DUP flag | QoS level | | RETAIN |
|  | 1 | 1 | 0 | 0 | x | x | x | x |
| byte 2 | Remaining Length (2) | | | | | | | |
|  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

QoS level

Не используется

DUP flag

Не используется

RETAIN flag

Не используется

Remaining Lenth = 0

Не используется

***Переменная часть заголовка***

Отсутствует

***Данные***

Отсутствуют

***Ожидаемая реакция***

Передача команды PINGRESP от отвечающей стороны

**3.13. Команда PINGRESP – Ping – подтверждение**

Дословно означает – «Я жив!» и посылается Абонентом или Сервером как ответ на команду PINGACK в связи с наступлением события Keep Alive timer

***Фиксированная часть загловка***

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| byte 1 | Message type (13) | | | | DUP flag | QoS level | | RETAIN |
|  | 1 | 1 | 0 | 1 | x | x | x | x |
| byte 2 | Remaining Length (2) | | | | | | | |
|  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

QoS level

Не используется

DUP flag

Не используется

RETAIN flag

Не используется

Remaining Lenth = 0

Не используется

***Переменная часть заголовка***

Отсутствует

***Данные***

Отсутствуют

**3.14. Команда DISCONNECT – извещение об отключении**

Посылается Клиентом на Сервер с целью известить о прекращении работы по протоколу TCP/IP. Обеспечивает управляемое отключение, в отличие от непредвиденного разрыва связи.

Если Клиент был присоединен командой CONNECT со сброшенным флагом Clean Session, то вся информация, относящаяся к этому Клиенту будет уничтожена.

Сервер не поддерживает протокол TCP/IP после получения команды DISCONNECT

***Фиксированная часть загловка***

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| byte 1 | Message type (14) | | | | DUP flag | QoS level | | RETAIN |
|  | 1 | 1 | 1 | 0 | x | x | x | x |
| byte 2 | Remaining Length (2) | | | | | | | |
|  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

QoS level

Не используется

DUP flag

Не используется

RETAIN flag

Не используется

Remaining Lenth = 0

Не используется

***Переменная часть заголовка***

Отсутствует

***Данные***

Отсутствует

**ПРИЛОЖЕНИЕ №1 MQTT и UTF-8**

UTF-8 обеспечивает более эффективную кодировку символов/строк, чем ASCII, поддерживаемой для кодировки текстов

Этот формат используется в Java в методах потокового ввода/вывода *writeUTF()* и *readUTF()*

В формате UTF-8 строки начинаются с двумя байтами, определяющими длину строки, следующую за ними

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Байт 1 | Длина строки MSB | | | | | | | |
| Байт 2 | Длина строки LSB | | | | | | | |
| Байт 3… | Кодированные символьные данные в формате UTF-8 | | | | | | | |

Длина строки – количество байт закодированной строки символов, а не количество символов.

Преобразование значения числа, определяющего длину строки (*\_value* ) в строковый поток (\_*stream* ) из символов (*\_digit* ) организовано по следующему алгоритму – очередной символ (*\_digit* ) в потоке равен значению числа по модулю 128 и если результат целочисленного деления числа на 128 больше 0, те значение числа все еще больше 127, то 8 бит в символе устанавливается в 1, символ выводится в поток, итерация повторяется, пока значение числа не станет меньше 128.

*do*

*\_digit = \_value MOD 128*

*\_value = \_value DIV 128*

*// if there are more digits to encode, set the top bit of this digit*

*if ( \_value > 0 )*

*\_digit = \_digit OR 0x80*

*endif*

*'put next digit to\_ stream' \_digit*

*while ( value > 0 )*

|  |  |  |
| --- | --- | --- |
| Цифра | От | До |
| 1 | 0 (0x00) | 127 (0x7F) |
| 2 | 128 (0x80, 0x01) | 16 383 (0xFF, 0x7F) |
| 3 | 16 384 (0x80, 0x80, 0x01) | 2 097 151 (0xFF, 0xFF, 0x7F) |
| 4 | 2 097 152 (0x80, 0x80, 0x80, 0x01) | 268 435 455 (0xFF, 0xFF, 0xFF, 0x7F) |

Обратное преобразование строкового потока (*stream*), состоящего из символов (*\_digit* ) в значение числа (*\_value* ), определяющего длину строки, производится по алгоритму

*\_multiplier = 1*

*\_value = 0*

*do*

*\_digit = 'get next digit from stream'*

*\_value += ( \_digit AND 127 ) \* \_multiplier*

*\_multiplier \*= 128*

*while ( ( \_digit AND 128) != 0 )*

Например, строка” OTWP” кодируется в UTF-8 следующим образом

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| byte 1 | Длина строки MSB (0x00) | | | | | | | |
|  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| byte 2 | Длина строки LSB (0x04) | | | | | | | |
|  | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| byte 3 | 'O' (0x4F) | | | | | | | |
|  | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| byte 4 | 'T' (0x54) | | | | | | | |
|  | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| byte 5 | 'W' (0x57) | | | | | | | |
|  | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 |
| byte 6 | 'P' (0x50) | | | | | | | |
|  | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |

Неиспользуемые биты

В MQTT все неиспользуемые или зарезервированные биты должны быть сброшены (иметь значение 0)

**ПРИЛОЖЕНИЕ №2 Синтаксис подстановок в Темах**

Подписка, осуществляемая Абонентом, может содержать спецсимволы, которые позволяют подписаться сразу на несколько Тем сразу. Это утверждение справедливо только для Абонентов. Издатели обязаны формировать Тему без подстановок.

Разделитель уровня подтем используется для представления структуры в Теме. Могут применяться многоуровневые и одноуровневые подстановки.

***Разделитель подтем – символ «/»***

Применяется для разделения каждого уровня в дереве Тем и обеспечивает иерархическую структуру адреса, по которому находится конкретная Темы.

***Многоуровневая подстановка – символ «#»***

Применение символа «#» указывает на все подтемы в пределах заданной Темы. Например, если указано

*finance/stock/IBM/#*

то Абоненту будет открыта подписка к подтемам

*finance/stock/IBM*

*finance/stock/IBM/closingprices*

*finance/stock/IBM/currentprice*

Мультиуровневая подстановка может быть на нулевом и более низких уровнях. Например, подписка на ***«#»*** будет означать, что Абоненту будет открыта подписка на все темы, опубликованные на Сервере.

Мультиуровневая подстановка должна быть последним знаком в строке, то есть фраза *finance/#/stock* недопустима

Также мултиуровневая подстановка может быть самостоятельным объектом подписки, Например, подписка на ***«#»*** будет означать, что Абоненту будет открыта подписка на все темы, опубликованные на Сервере, либо последним знаком в строке, то есть фразы *finance/#/stock* или *finance#* недопустимы.

***Одноуровневая подстановка – символ «+»***

Применение символа «+» указывает на темы конкретного уровня. Например, если указано

*finance/stock/+*

то Абоненту будет открыта подписка к подтемам

*finance/stock/IBM*

*finance/stock/Microsoft*

*finance/stock/* ConocoPhillips

но не на

*finance/stock/IBM/currentprice*

Одноуровневые подстановки могут применяться на любом уровне иерархии подтем, также своместно с многоуровневой подстановкой. Обязательно применение после разделителя подтем. Таким образом, «+» и «finance/+» допустимы, а «finance+» нет. Одноуровневая подстановка может как в конце, так и в середине иерархии, например, «finance/+» и «finance/+/IBM» допусимы.

Полезные ссылки

1. IBM Podcast: Piper, Diaz, Nipper – MQTT. [online]. Доступ через: [http://www.ibm.com/podcasts/software/](http://www.ibm.com/podcasts/software/" \t "_blank)websphere/connectivity/piper\_diaz\_nipper\_mq\_tt\_1118201 1.pdf.
2. OASIS Standard – MQTT Version 3.1.1.
3. ISO/IEC 20922:2016 Information technology – Message Queuing Telemetry Transport (MQTT) v3.1.1.
4. Web-хостинг GitHub, раздел MQTT – libraries. [online]. Доступ через: [https://github.com/mqtt/mqtt.github.io/wiki/libraries](https://github.com/mqtt/mqtt.github.io/" \t "_blank).
5. IBM WebSphere MQ. [online]. Доступ через: [http://www-03.ibm.com/software/products/ru/websphere-mq](http://www03.ibm.com/software/products/ru/websphere-mq" \t "_blank).
6. An Open Source MQTT v3.1/v3.1.1 Broker. [online]. Доступ через: [https://mosquitto.org.](https://mosquitto.org/" \t "_blank)
7. Everywhere Device Cloud [online]. Доступ через: [http://www.eurotech.com/en/products/software+services/everyware+device+cloud](http://www.eurotech.com/en/products/software+services/e" \t "_blank).
8. EMQ – The Massively Scalable MQTT Broker for IoT and Mobile Applications. [online]. Доступ через: [http://emqtt.io.](http://emqtt.io/" \t "_blank)
9. HiveMQ – Enterprise MQTT Broker. [online]. Доступ через: [http://www.hivemq.com.](http://new.groteck.ru/imag20/redir.php?s=9604&url=www.hivemq.com" \t "_blank)
10. Протокол MQTT. Особенности, варианты применения, основные процедуры MQTT . Доступ через: http://www.tssonline.ru/articles2/fix-corp/protokol-mqtt-osobennosti-varianty-primeneniya-osnovnye-protsedury-mqtt-protocol./