Математические структуры и моделирование
2023. № 2 (66). С. 91–102

УДК 004.054
DOI 10.24147/2222-8772.2023.2.91-102

СРАВНЕНИЕ ПРОТОКОЛОВ СВЯЗИ ДЛЯ ОРГАНИЗАЦИИ M2M-ВЗАИМОДЕЙСТВИЙ В SCADA-СИСТЕМАХ И СИСТЕМАХ ПРОМЫШЛЕННОГО ИНТЕРНЕТА ВЕЩЕЙ

Т.В. Костеннов
аспирант, e-mail: timofey.kostennov@gmail.com

Омский государственный университет им. Ф.М. Достоевского, Омск, Россия

Аннотация. В современной промышленности широко применяются информационные технологии и системы, которые помогают в отслеживании и управлении технологическими параметрами и процессами для повышения эффективности и безопасности производства. Исходя из беспрецедентных темпов роста данной отрасли в настоящее время и тенденции к расширению, ожидается, что большое количество подобных устройств и систем будет создано в будущем. Рассмотрены основные теоретические характеристики наиболее распространённых протоколов передачи данных в M2M-соединениях интернета вещей и результаты практических экспериментов, позволяющие оценить эти протоколы по таким параметрам, как количество сетевых ресурсов, размер одного сообщения и доли потерянных пакетов при работе в условиях нестабильного канала связи.

Ключевые слова: Интернет вещей, IoT, MQTT, AMQP, CoAP M2M.

Введение

Применение автоматизированной передачи данных позволяет, например, сократить время переналадки оборудования, быстрее реагировать на аварии и обеспечить плавный ход производства. Однако возникновение проблем при передаче информации, её искажение или потеря может привести к снижению эффективности и нарушениям технологического процесса. Поэтому задача обеспечения надёжности, безопасности и быстрой передачи данных является важной и требует комплексного подхода для её решения. Актуальность проблемы информационной инфраструктуры промышленности обусловлена как усложнением технологий производства, так и высокими темпами роста отрасли [1]. Из-за гетерогенности устройств внутри сетей промышленного интернета вещей выбор оптимального протокола для обеспечения M2M передачи данных может производиться на основе различных параметров. Кроме заявленных разработчиками протоколов параметров, следует использовать и те данные, которые можно получить в ходе экспериментальных оценок. Одним из возможных решений может быть использование протоколов интернета вещей для обеспечения M2M-соединений. Эти протоколы находят всё более широкое применение в сфере межмашинных взаимодействий. Однако, прежде чем переходить к

применению конкретного протокола, стоит рассмотреть существующие решения для M2M связи, их преимущества и недостатки, а также область их применения в SCADA-системах.

SCADA-системы представляют собой программные комплексы, являющиеся частью автоматизированных систем управления технологическим процессом. Термин является аббревиатурой и расшифровывается как "Supervisory Control And Data Acquisition" – «диспетчерское управление и сбор данных». Для обозначения комплексных систем, объединяющих в сети промышленные или производственные объекты со встроенными датчиками, программным обеспечением, для сбора и обмена данными и удалённого контроля используется термин «промышленный интернет вещей» или же IIoT – "Industrial Internet of Things". Для обеспечения межмашинных коммуникаций как в промышленном, так и в обычном интернете вещей используются такие протоколы, как Advanced Message Queuing Protocol (AMQP) [2], Message Queuing Telemetry Transport (MQTT) [3] и Constrained Application Protocol (CoAP) [4]. Предлагается рассмотреть теоретические характеристики данных протоколов и провести эксперименты для оценки возможностей протоколов и выбора наиболее оптимальных вариантов для задачи достоверной передачи данных и задачи передачи данных с наименьшей нагрузкой на канал связи.

1. Advanced Message Queuing Protocol (AMQP)

Протокол AMQP был разработан в 2003 году Джоном О'Хара (John O'Hara), работавшим в компании JPMorgan [5]. AMQP – это открытый стандартизованный протокол прикладного уровня, предназначенный для обеспечения взаимодействия между широким спектром различных приложений и систем независимо от их внутреннего устройства. Первоначально он разрабатывался для финансового сектора с идеей предложить непатентованное решение, способное управлять процессом обмена большим количеством сообщений, генерируемых в системе за короткий промежуток времени. Протокол AMQP позволяет различным платформам обмениваться сообщениями вне зависимости от внутренней реализации, что может быть особенно полезно в гетерогенных системах.

AMQP использует модель «издатель – подписчики» (рис. 1). Основной частью протокола является брокер (broker), который состоит из точек обмена (exchanges) и очередей сообщений (message queues) [6]. Издатель отправляет сообщения в точки обмена. Точка обмена маршрутизирует входящее сообщение в соответствующую очередь сообщений посредством сопоставления ключа привязки (binding key) очереди с ключом маршрутизации сообщения (routing key). Если сообщение должны получить более одного подписчика, сообщение дублируется в другие очереди сообщений.

Протокол поддерживает три уровня надёжности доставки (качества обслуживания) [6]:

- "at most once"(«не более одного раза») – без подтверждения доставки сообщения;
- "at least once"(«по крайней мере один раз») – с подтверждением доставки;
- "exactly once"(«ровно один раз») – с подтверждением доставки без дублирования.

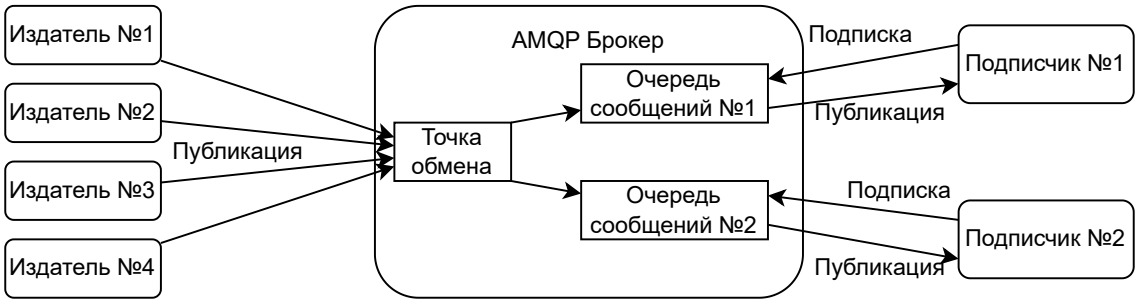

Рис. 1. Архитектура протокола AMQP

Пакет протокола состоит из пяти частей:

1. Заголовка протокола, использующегося при установлении связи брокера с клиентом.
2. Кадра метода, использующегося для удалённого вызова процедур.
3. Заголовка данных, описывающего содержимое тела данных.
4. Тела данных, содержащего в себе до 131 KB данных.
5. Кадра времени жизни, использующегося для определения состояния соединения.

2. Message Queuing Telemetry Transport (MQTT)

MQTT – это легковесный сетевой протокол обмена сообщениями по принципу «издатель – подписчик» (publisher/subscriber). Первая версия была представлена в 1999 году Энди Стэнфордом-Кларком (Andy Stanford-Clark) из IBM и Арленом Ниппером (Arlen Nipper) из Cirrus Link. MQTT разрабатывался как способ поддержания связи между устройствами в сетях с ограниченной пропускной способностью или непредсказуемой связью в рамках разработки. Впервые протокол MQTT был опубликован консорциумом OASIS (Organization for the Advancement of Structured Information Standards) в октябре 2014 года. Данный стандарт находится в открытом доступе [7]. MQTT – один из вариантов для разработчиков, которые создают приложения и устройства с надёжной функциональностью и широкой совместимостью с подключёнными к интернету устройствами и приложениями, включая браузеры, смартфоны и устройства IoT [8].

Протокол MQTT построен по шаблону «издатель – подписчик». В соответствии с данным шаблоном отправители, именуемые издателями, не связаны напрямую с получателями, именуемыми подписчиками, и, как правило, разделены. Разделение может быть организовано в различных плоскостях:

- Издатель и подписчик не общаются напрямую – разделение в пространстве.
- Издатель и подписчик могут быть включены в разное время – разделение во времени.
- Издатель и подписчик не приостанавливают выполнение операций в процессе публикации или получения информации – разделение в синхронизации.

Координацией и передачей сообщений управляет брокер (рис. 2). Упрощённо

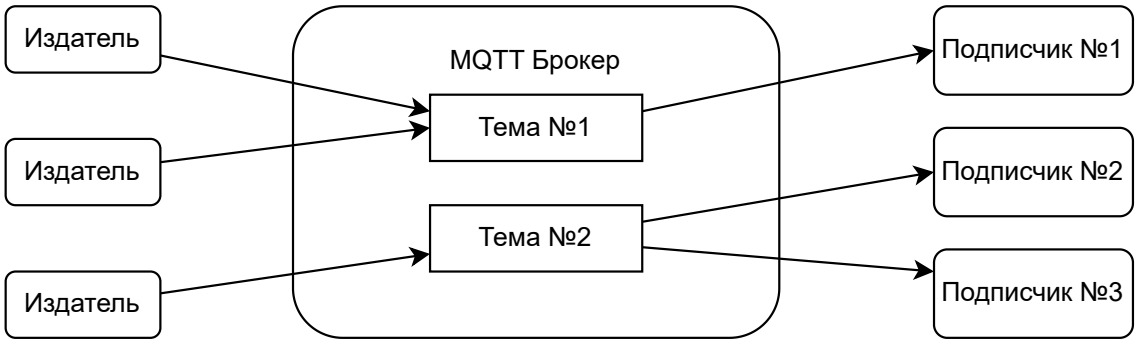

Рис. 2. Архитектура протокола MQTT

процесс обмена данными в такой системе можно описать следующим образом:

1. Издатель отправляет данные с указанием темы, к которой относятся эти данные, брокеру.
2. Брокер отправляет данные подписчикам, имеющим подписку на указанную тему.

Любое количество подписчиков может быть подписано на любое количество тем и получать с помощью этого механизма различные данные без необходимости контактировать напрямую с издателем. Тема представляет собой строку из символов с кодировкой UTF-8. Структура тем имеет формат дерева, что облегчает их организацию. Различные уровни дерева тем разделяются с помощью символа «/» [9].

Протокол поддерживает три уровня надёжности доставки (качества обслуживания):

- 0 – без подтверждения доставки сообщения;
- 1 – с подтверждением доставки;
- 2 – с подтверждением доставки без дублирования.

Пакет протокола состоит из трёх частей:

1. Фиксированного заголовка, содержащего в себе информацию о типе сообщения, необходимости дублирования, качестве обслуживания, длине сообщения.
2. Переменного заголовка, содержащего в себе идентификатор пакета, название и версию протокола, флаги соединения.
3. Кадра данных, содержащего в себе полезные данные.

3. Constrained Application Protocol (CoAP)

Протокол CoAP (описывается стандартом RFC 7252) – это протокол передачи данных, предназначенный для использования в устройствах с сильно ограниченными ресурсами (встраиваемые системы на основе микроконтроллеров), работающих в сетях интернета вещей. Протокол был разработан рабочей группой в 2014 году специально для работы с устройствами, которые имеют весьма ограниченный объём ресурсов, например, порядка 10 КБ оперативной памяти и 100 КБ постоянной [10].

CoAP предназначен для непосредственного соединения двух узлов между собой,

используя взаимодействие в рамках логики «сервер – клиент»: здесь, во время взаимодействия с клиентом, сервер делает доступным свои ресурсы по определённому адресу, а клиенты обращаются к тому адресу с помощью стандартных HTTP-методов. Однако в рамках этого протокола возможно общение не только между двумя устройствами с ограниченными аппаратными и сетевыми возможностями, но и между слабым клиентом и более мощным сервером (рис. 3). Обмен сообщениями между сервером и клиентом возможен в асинхронном режиме, что существенно ускоряет работу системы. Кроме того, спецификация протокола рассчитана на достаточно скромные сетевые возможности, и именно поэтому в его рамках не реализуются многоуровневые транспорты для передачи сообщений. Вместо этого используется протокол UDP поверх IP.

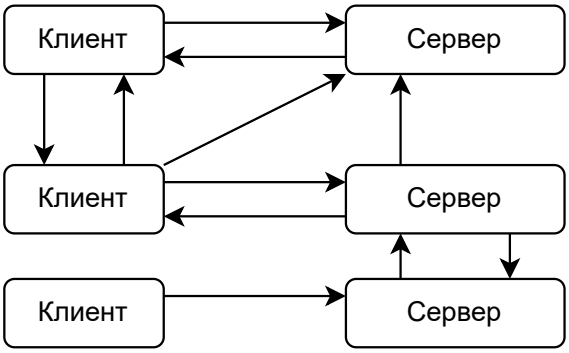

Рис. 3. Архитектура протокола CoAP

Протокол не имеет встроенной настройки для качества обслуживания, однако поддерживает различные типы ответа:

- несогласующийся ответ – без подтверждения от сервера;
- совмещённый ответ – подтверждение от сервера в ACK-пакете;
- отдельный ответ – в случае если подтверждение от сервера было потеряно, оно будет отправлено в другом ACK-пакете в ходе следующего запроса от клиента.

Пакет протокола состоит из четырёх частей:

1. Фиксированного заголовка, содержащего в себе информацию о версии протокола, типе сообщения, длине переменного заголовка, коде запроса/ответа, идентификационном номере сообщения.
2. Токена, использующегося для идентификации клиента.
3. Заголовка настроек, содержащего в себе служебные настройки протокола.
4. Кадра данных, содержащего в себе полезные данные, описанные в предыдущей части.

4. Дизайн эксперимента

Проведены два эксперимента, в которых рассмотрены протоколы AMQP, MQTT и CoAP. В первом эксперименте предлагалось оценить производительность каждого

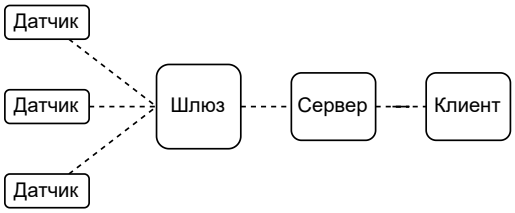

Рис. 4. Структурная схема сети первого эксперимента

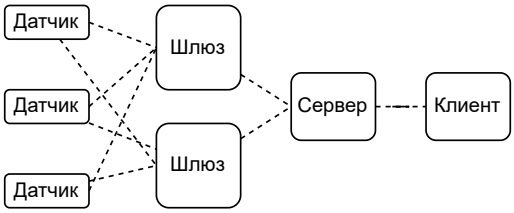

Рис. 5. Структурная схема сети второго эксперимента

протокола при штатной работе без сбоев связи. Во втором эксперименте учитываются случайные сетевые сбои при передаче данных.

Все эксперименты проведены не менее 30 раз для обеспечения выборки не менее чем 30 повторений для получения средних значений с приблизительно нормальным распределением. Это позволяет уменьшить влияние возможных шумов измерений. Используемые метрики – средняя используемая ширина канала связи, размер сообщения и потери пакетов.

Оба эксперимента предполагают использование датчиков, собирающих информацию о внешней среде, и пересылку данных на сервер, используя беспроводную 2.4 ГГц сеть. В первом эксперименте существует только один маршрут от датчиков к серверу (рис. 4). Сенсоры опрашиваются каждые 5 секунд. Спустя 300 секунд рассчитываются средние значения каждой метрики. Это позволяет отфильтровать возможные шумы измерений.

Во втором эксперименте существует два возможных маршрута (рис. 5). Всякий раз, когда приложение сталкивается с ошибкой передачи данных, происходит перенаправление на альтернативный маршрут. Сенсоры так же, как и в первом эксперименте, опрашиваются каждые 5 секунд. Из-за сбоев время до подсчётов средних значений каждой метрики увеличено до 600 секунд.

Для эмуляции отказа оборудования был предложен следующий алгоритм:

1. Сгенерировать случайное число на интервале от 0 до 1 с нормальным распределением.
2. Вычислить время отказа оборудования по формуле (1).
3. Приостановить любые действия на время отказа оборудования.
4. Сбросить соединение.

$$t_{TTF} = -t_{ATTF} \times \ln(1 - c), \tag{1}$$

где t_{TTF} – время отказа оборудования, t_{ATTF} – среднее время отказа оборудования (в рамках эксперимента принято равным 60 секундам), c – случайное число на интервале от 0 до 1, полученное с использованием равномерного распределения.

Схожий алгоритм используется и для восстановления оборудования после отказа в обслуживании:

1. Сгенерировать случайное число на интервале от 0 до 1 с нормальным распределением.
2. Вычислить время восстановления оборудования после отказа в обслуживании по формуле (2).

3. Приостановить любые действия на время восстановления оборудования после отказа в обслуживании.
4. Восстановить соединение.

$$t_{TTR} = -t_{ATTR} \times ln(1 - c), \tag{2}$$

где t_{TTR} – время восстановления после отказа в обслуживании, t_{ATTR} – среднее время восстановления оборудования после отказа в обслуживании (в рамках эксперимента принято равным 2 секундам), c – случайное число на интервале от 0 до 1, полученное с использованием равномерного распределения.

В зависимости от протокола физические объекты были по-разному описаны на логическом уровне. В случае протокола AMQP, использующего модель «публикация – подписка», датчики являются издателями, а сервер – подписчиком. Схожую модель использует и протокол MQTT. Для него использовано такое же сопоставление. Протокол CoAP использует модель «клиент – сервер», где сервер отвечает за получение данных от клиентов, в рамках эксперимента клиентами являются датчики.

5. Используемые инструменты

Датчики подключены к SoC-системам ESP8266. Такие системы используются для проектирования устройств интернета вещей [11]. Сервер использует платформу Raspberry Pi 4 и Raspbian Kernel 5.15.

Для реализации протоколов передачи данных использованы инструменты с открытым исходным кодом Mosquitto [12], RabbitMQ [13] и CoAPthon3 [14]. Существуют и другие реализации, однако используемые в данном эксперименте имеют открытый исходный код и подробную документацию. Для сбора данных о работе сети используется инструмент tcpdump.

В обоих экспериментах настройки протоколов установлены в такое состояние, чтобы сообщения от датчиков достигли пункта назначения как минимум один раз. Для MQTT и AMQP используется параметр QoS-1 (QoS-at least once для AMQP), для CoAP – режим работы с совмещённым ответом. Все протоколы были настроены на изменение опции поддержки активности (keep alive). Режим подключения скорректирован таким образом, чтобы приложение продолжало попытки подключения при сбое связи.

6. Результаты экспериментов

Как было описано ранее, было проведено два эксперимента для оценки производительности протоколов связи. В первом эксперименте производительность измерялась без сбоев связи, во втором учитывались возможные сбои, внедряемые искусственно. К результатам каждого эксперимента был применён однофакторный дисперсионный анализ с заданным уровнем значимости 0.05, а также апостериорный критерий Тьюки [15].

7. Первый эксперимент

Результаты дисперсионного анализа для используемой ширины канала связи и размера сообщения представлены в табл. 1.

Таблица 1. Анализ первого эксперимента

Ширина канала связи			Размер сообщения		
df	F	p	df	F	p
2	2310	<0.001	2	30936	<0.001

В данной таблице df – степень свободы, F – статистика теста, p – p-значение. С точки зрения пропускной способности (рис. 6), наибольшее среднее значение, а значит, наибольшее использование сетевых ресурсов имеет протокол AMQP со значением 97.24 Б/с. Протокол MQTT, также как и AMQP, основывается на TCP, но имеет меньшие требования к пропускной способности канала на уровне 57.62 Б/с. Протокол CoAP имеет наименьшее значение, равное 35.65 Б/с. Из этого можно сделать вывод, что протокол CoAP оказывает наименьшее влияние на пропускную способность сети, что может оказаться важным при использовании в сетях IoT с ограниченными сетевыми ресурсами.

Рассматривая такую метрику, как размер сообщения (рис. 7), можно отметить, что протокол AMQP имеет в среднем наибольшее значение в 145.62 байт/пакет. Наименьшее значение, равное 59.2 байт/пакет, имеет протокол CoAP. Протокол MQTT показывает результат в среднем 89.2 байт/пакет. Заметна высокая корреляция между количеством занимаемых ресурсов канала связи (коэффициент корреляции $r = 0.945$). Это означает, что сообщение с большим размером увеличивает объём передаваемых данных. Из этого следует вывод о том, что протокол CoAP передаёт тот же объём информации, используя меньше данных.

8. Второй эксперимент

В ходе этого эксперимента было произведено искусственное внедрение ошибок для оценки протоколов в условиях нестабильной связи.

В табл. 2 представлены результаты дисперсионного анализа для используемой ширины канала связи, размера сообщения и количества потери пакетов.

Таблица 2. Анализ второго эксперимента

Ширина канала связи			Размер сообщения			Потеря пакетов		
df	F	p	df	F	p	df	F	p
2	2737	<0.001	2	25901	<0.001	2	11924	<0.001

Исходя из полученных данных (рис. 8), AMQP имеет наибольшее использование сетевых ресурсов, равное 103.24 Б/с, что превышает значение из первого экспери-

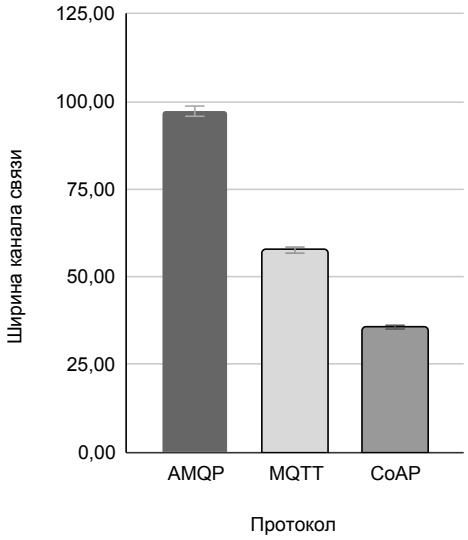

Рис. 6. Первый эксперимент. Средняя ширина канала связи

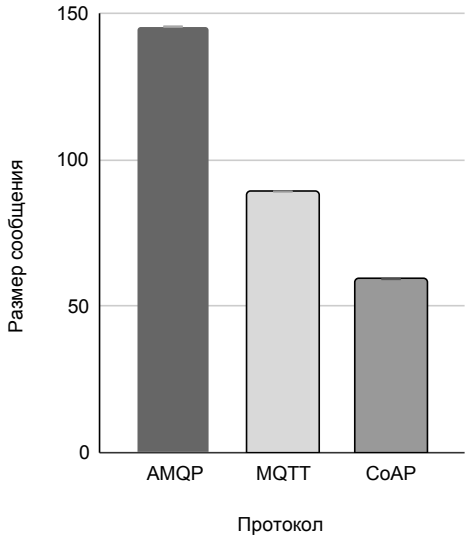

Рис. 7. Первый эксперимент. Средний размер сообщения

мента. Причиной этому послужили дополнительные повторные передачи, включающие дублирующие данные. Уже исходя из этого можно предположить, что протокол AMQP не следует использовать для систем, в которых используется нестабильный канал связи. Протокол CoAP имеет меньшую используемую ширину канала связи, значение которой равно 36.2 Б/с и почти не увеличивается по сравнению с первым экспериментом. Из этого следует вывод о том, что с помощью данного протокола получилось справиться с нестабильным каналом связи без значимого увеличения используемой ширины канала связи. Для протокола MQTT во втором эксперименте среднее использование канала связи составило 60.1 Б/с, также незначительно возрастая по сравнению с первым экспериментом.

Для среднего размера сообщения данные, полученные в ходе второго эксперимента, согласуются с данными первого, при этом сохраняется умеренная корреляция между размером сообщения и использованием канала связи (коэффициент корреляции $r = 0.65$) (рис. 9). Самые большие сообщения всё ещё имеет протокол AMQP с результатом 93.42 байт/пакет. Результаты для протоколов MQTT и CoAP почти совпадают с результатами первого эксперимента и равны 90.1 байт/пакет и 60.1 байт/пакет соответственно. Снижение размера пакета в случае протокола AMQP можно объяснить тем, что AMQP передаёт данные только в том случае, когда соединение с сервером полностью восстановлено.

Наиболее показательной метрикой в данном эксперименте следует считать долю потерянных пакетов (рис. 10) Протокол MQTT имеет наименьший показатель, равный 0.3 %. Протокол CoAP имеет 1.2 % потерянных пакетов. Самый большой показатель по данной метрике наблюдается у протокола AMQP и равен 51 % даже с учётом уменьшения сообщения для повторной передачи. Следует отметить корреляцию между долей потерянных пакетов и средней используемой шириной канала

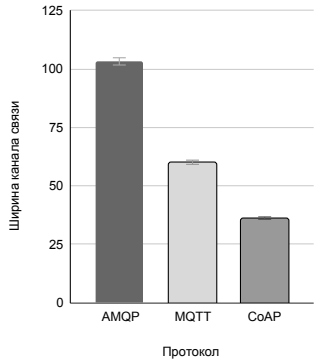

Рис. 8. Второй эксперимент. Средняя ширина канала связи

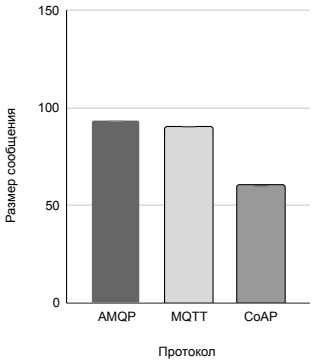

Рис. 9. Второй эксперимент. Средний размер сообщения

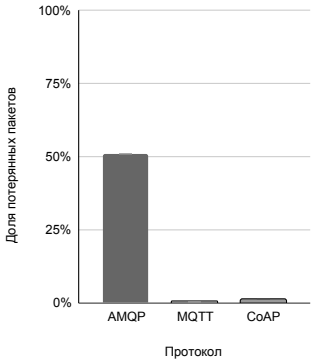

Рис. 10. Второй эксперимент. Доля потерянных пакетов

(коэффициент корреляции r = 0.85). Но между долей потерянных пакетов и размером сообщения сильной корреляции не наблюдается (коэффициент корреляции r = 0.25).

9. Выводы

Исходя из результатов обоих экспериментов, можно сказать, что протокол CoAP наиболее подходит для сетей промышленного интернета вещей с низкой пропускной способностью из-за низкой используемой ширины канала связи. Протокол MQTT показал самый высокий показатель доставленных пакетов в случаях нестабильного канала связи. На использование протокола AMQP налагают ограничения как высокие по сравнению с другими протоколами требования к ширине канала связи, так и большие потери в случае, если канал связи нестабилен. Кроме того, следует отметить, что реализация протокола AMQP для устройств на базе SoC-систем ESP8266 является гораздо более ресурсоёмкой, чем двух других протоколов, хоть это и выходит за рамки данного эксперимента.

Наиболее подходящий протокол для построения межмашинных взаимодействий в сетях промышленного интернета вещей и SCADA-системах следует выбирать исходя из предъявляемых требований. В тех случаях, когда наиболее важна сохранность данных, следует отдать предпочтение протоколу MQTT, который имеет наименьшую долю потери пакетов. В тех же случаях, когда требуется максимально снизить нагрузку на сетевую инфраструктуру, следует использовать протокол CoAP, который имеет наименьшее значение используемой ширины канала связи.

Данные эксперименты проводились без углублённой настройки каждого протокола под различные возможные задачи, сравнение проводилось со стандартными настройками, за исключением настройки качества обслуживания.

10. Заключение

Межмашинные взаимодействия в сетях промышленного интернета вещей являются одним из ключевых процессов. Многие исследования были проведены для создания, оценки и изучения применений протоколов связи для систем IIoT. В данной статье рассмотрены теоретические характеристики трёх наиболее популярных протоколов связи интернета вещей: AMQP, MQTT, CoAP. Были проведены эксперименты, в результате которых были получены практические данные о каждом протоколе, связанные с работой протоколов в различных условиях сети.

В дальнейшем планируется использование данных протоколов в построении автоматических систем управления.

Литература

1. 8.4 Billion Connected Things Will be in Use 2017 | Gartner. URL: `https://www.information-age.com/gartner-8-4-billion-iot-2017-4225` (дата обращения: 04.04.2023).
2. Advanced Message Queuing Protocol (AMQP) Version 1.0. // OASIS Standard. 29 October 2012.
3. Banks A., Briggs E., Borgendale K., Gupta R. MQTT Version 5.0. // OASIS Standard. 07 March 2019.
4. Shelby Z., Hartke K., Bormann C. The constrained application protocol (CoAP). // IETF RFC-7252. 2014.
5. O'Hara J. Toward a Commodity Enterprise Middleware // ACM Queue. May-June 2007. V. 5, Issue 4. P. 48–55.
6. Walcher F., Kastner W. KNX to MQTT/AMQP // Computer Science. 2019.
7. OASIS Standard – MQTT Version 3.1.1. URL: `http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html` (дата обращения 24.03.2023).
8. Ли П. Архитектура интернета вещей. М. : ДМК Пресс, 2019. 454 с.
9. MQTT Specification. URL: `https://mqtt.org/mqtt-specification/` (дата обращения: 11.03.2023)
10. The Constrained Application Protocol (CoAP). URL: `https://www.rfc-editor.org/rfc/rfc7252` (дата обращения: 11.03.2023).
11. Light R.A. Mosquitto: server and client implementation of the MQTT protocol // Journal of Open Source Software. 2017. V. 2, No. 13.
12. Корзухин С.В., Хайдарова Р.Р., Шматков В.Н. Конфигурируемые IoT-устройства на основе SOC-систем ESP8266 и протокола MQTT // Научно-технический вестник информационных технологий, механики и оптики. 2020.
13. Inc. Pivotal Software. Messaging that just works – RabbitMQ. 2014. URL: `https://www.rabbitmq.com/` (дата обращения: 12.03.2023).
14. Tanganelli G., Vallati C., Mingozzi E. Coapthon: Easy development of coap-based iot applications with python // Internet of Things (WF-IoT). 2015 IEEE 2nd World Forumon. IEEE. 2015. P. 63–68.
15. Montgomery D.C. Design and analysis of experiments. John wiley and sons, 2017.

COMPARISON OF COMMUNICATION PROTOCOLS FOR ORGANIZING M2M-INTERACTIONS IN SCADA SYSTEMS AND INDUSTRIAL IOT-SYSTEMS

T.V. Kostennov
Ph.D. Student, e-mail: timofey.kostennov@gmail.com

Dostoevsky Omsk State University, Omsk, Russia

Abstract. In modern industry, information technologies and systems are widely used, which help in tracking and managing technological parameters and processes in order to increase the efficiency and safety of production. Based on the unprecedented growth rate of the industry at present and the expansion trend, it is expected that a large number of such devices and systems will be created in the future. This paper discusses the main theoretical characteristics of the most common data transfer protocols in M2M connections of the Internet of Things and the results of practical experiments that allow us to evaluate these protocols in terms of such parameters as the number of network resources, the size of one message, and the percentage of lost packets when working in unstable conditions.

Keywords: Internent of Things, MQTT, AMQP, CoAP, M2M.

Дата поступления в редакцию: 22.04.2023