Одним из широко используемых протоколов для взаимодействия между IoT-устройствами (Internet of Things) или IoT-устройствами и внешней средой является протокол CoAP – Constrained Application Protocol [1, 2, 3].

Протокол CoAP создан рабочеи группои The Internet Engineering Task Force (IETF) Constrained RESTful Environments (CoRE) в июне 2014 г. Стандарт данного протокола описан в документе RFC 7252 [4].

Протокол CoAP предназначен для взаимодействия простых устройств, например датчиков малой мощности, выключателей, клапанов, которые управляются или контролируются удаленно через сеть Интернет. Такие устройства используются в области Интернета вещей, а порождаемый ими информационный обмен называется межмашинным взаимодействием (М2М). Часто подобные устройства называют устройствами с ограниченными ресурсами. Они обычно имеют ограниченный энергоресурс, небольшой объем памяти и невысокую мощность, поэтому в работе с ними важно обеспечивать низкие энергозатраты, использовать передачу сообщений малого объема. Протокол CoAP обеспечивает взаимодействие этих устройств, соблюдая все необходимые требования. Сеть, в которой работают такие устройства, называют сетью с ограниченными ресурсами. Существенная особенность протокола СoAP – это его совместимость с протоколом HTTP, что обеспечивает при его использовании взаимодействие совокупности устройств IoT, формирующих некую сеть, с всемирной паутиной Интернет.

**Использование протокола СоАР**

Рассмотрим такое устройство с ограниченными ресурсами, как датчик. У пользователя в помещении может быть установлено некоторое количество датчиков, управлять которыми он может либо через сеть Интернет, либо непосредственно через сеть с ограниченными ресурсами в том случае, если на его устройстве установлено приложение, работающее по протоколу СоАР (рис. 1).



На рис.1 область "А" показывает пример взаимодействия пользовательского устройства (HTTP-клиента) с СоАР-датчиком через сеть Интернет. HTTP-клиент генерирует HTTP-запросы и отправляет их на сервер, который при отсутствии необходимой информации на полученный запрос обращается к CoAP-прокси. Здесь CoAP-прокси – это устройство, соединяющее сеть Интернет на основе протокола HTTР и сеть с ограниченными ресурсами, поддерживающую протокол CoAP. Прокси преобразует сообщения одной сети (протокол HTTP) в сообщения, понятные для другой (протокол CoAP).

Если запрос от пользовательского устройства сразу попадает в сеть с протоколом СоАР, то он поступает на СоАР-датчик (на рисунке – область "Б"). При таком взаимодействии сервер получает все запросы и далее индивидуально однонаправленно посылает соответствующему устройству относящийся к нему запрос и ожидает от него ответ.

**Особенности CoAP**

Структура протокола CoAP была разработана в соответствии с REST-архитектурой. REST (Representational State Transfer) – это вид архитектуры, позволяющий строить приложения, в основе которых будет лежать следующая группа методов:

 GET – выполняет поиск ресурсов и предоставляет информацию, которая соответствует ресурсу, указанному в ссылке URI (Uniform Resource Identifier), где ресурс – это источник данных, которым, например, может являться датчик;

 PUT – задает новое действие над ресурсом;

 POST – отвечает за изменения действия над ресурсом;

 DELETE – служит для удаления активированных возможностей ресурса.

**Протокол CoAP создан рабочеи группои The Internet Engineering Task Force (IETF) Constrained RESTful Environments (CoRE) в июне 2014 г. Стандарт данного протокола описан в документе RFC 7252 [4]**

В апреле 2017 г. вышел документ RFC, расширяющий протокол CoAP новыми методами [12]. В данном стандарте описаны два метода:

 FETCH – выполняет предоставление частичной информации о ресурсе по параметрам в запросе;

 PATCH – отвечает за частичное изменение действия над ресурсом.

Протокол был дополнен этими методами, потому что возникла необходимость некоторым приложениям получать доступ или изменять в ресурс не полностью, а взаимодействовать только с определенной его составляющей.

Доступ к ресурсам, как упоминалось выше, выполняется по URI-последовательности символов, идентифицирующей физический или абстрактный ресурс. URI идентифицирует CoAP-ресурсы и служит средством для определения их местоположения. Пользовательское устройство запрашивает данные с датчика через URI-ссылку, но чтобы данный запрос был понятен датчику, он будет преобразован на пользовательском устройстве в один из методов REST и занесен в СоАР-сообщение. Например, при вводе URI-ссылки приложение на пользовательском устройстве создает запрос GET на датчик, где в опциях сообщения протокола CoAP будет содержаться URI. Более подробно про перевод URI в соответствующие опции рассмотрено ниже в разделе "Формат сообщений".

При взаимодействии с датчиком через сеть Интернет приложение на пользовательском устройстве будет работать по протоколу HTTP и использовать методы REST для отправки запросов. Благодаря использованию одинаковых методов в протоколах HTTP и CoAP возможен обмен сообщениями между пользовательским устройством, на котором находится приложение, и датчиком, использующим разные протоколы, ведь сообщения будут легко транслироваться друг в друга на прокси.

Протокол CoAP использует UDP (User Datagram Protocol), в качестве транспортного протокола по умолчанию, что позволяет уменьшить размер служебных данных и увеличить эффективность работы. В редких случаях могут также использоваться TCP или SCTP. Так как CoAP – это протокол прикладного уровня, то в случаях, когда нужно обеспечить безопасное соединение, используется шифрование сообщений по протоколу DTLS (Datagram Transport Layer Security) [5]. DTLS работает поверх протокола UDP и служит для защиты данных на транспортном уровне (работает между уровнем приложений и транспортным уровнем).

**Формат сообщений протокола CoAP**

У протокола СоАР определено всего четыре вида сообщений: Confirmable, Non-confirmable, Acknowledgement, Reset. Все сообщения кодируются в бинарном виде. Формат сообщения изображен на рис. 2. Сообщение начинается с заголовка, с фиксированным размером в 4 байта, за ним следует маркер, опции и полезная нагрузка, которая занимает всю оставшуюся часть дейтаграммы.



Поля в сообщении определяются следующим образом.

Версия (Version) (Ver) – двухбитное целое число, указывающее на номер версии CoAP. На данный момент существует только одна версия протокола.

Тип (Type) (Т) – двухбитное целое число. Указывает на тип сообщения Confirmable (0), Non-confirmable (1), Acknowledgement (2) или Reset (3).

Длина маркера (Token Length) (TKL) – четырехбитное целое число. Указывает длину поля маркера (Token) переменной длины (0–8 байт). Длины от 9 до 15 зарезервированы.

Код (Code) – восьмибитное целое число, делится на трехбитный и пятибитный класс (рис. 3).



Поле "код" записывается, как "c.dd", где "с" – это цифра от 0 до 7 для трехбитного подполя и "dd" – две цифры в диапазоне от 00 до 31 для пятибитного подполя.

 Код 0.00 указывает на пустое сообщение.

 Код 0.01–0.31 указывает на запрос.

 Код 1.00–1.31 зарезервирован.

 Код 2.00–2.31 указывает на успешный ответ.

 Код 3.00–3.31 зарезервирован.

 Код 4.00–4.31 указывает на ошибку клиента.

 Код 5.00–5.31 указывает на ошибку сервера.

 Код 6.00–7.31 зарезервирован.

Значения "dd" – это фиксированные табличные величины, которые указывают в случае, если сообщение – запрос на метод запроса (например, 0.01 – это запрос GET, 0.02 – это запрос POST); в случае, если сообщение – ответ, указывают на код ответа (например, 2.05 – это ответ, содержащий запрашиваемую информацию (Content), 4.04 – это ошибка, возникающая, когда запрашиваемое устройство отсутствует или временно недоступно (Not Found)). Всю таблицу значений поля Code можно посмотреть в RFC7252, раздел 12.1.2. [4].

Идентификатор сообщения (Message ID) – шестнадцатибитное целое число, являющееся уникальным идентификатором для сообщения. Message ID идентифицирует сообщение, чтобы определить, на какой запрос пришла информация или ошибка. CoAP-устройство, отправляя сообщение Confirmable или Non-confirmable, случайным образом генерирует значение идентификатора для этого сообщения. В ответных сообщениях Acknowledgement- или Reset-идентификаторы будут те же, что и в сообщении, на которое они отвечают.

За заголовком следует значение маркера (Token), которое может составлять от 0 до 8 байт, как указано в поле TKL. Значение маркера случайным образом генерируется устройством CoAP и используется в пределах одной сессии для установления соответствия запроса с ответом, т.е. происходит группировка сообщений по цепочке "запрос-ответ". Нулевое значение маркера используется, когда никакие другие маркеры не используются в устройстве, на которое отправляется запрос, или если запросы делаются последовательно и в малом количестве.

Далее следуют опции (Options), в которых описываются различные параметры сообщений. Например, есть Max-Age Option, которая устанавливает максимальное время хранения информации во временной памяти, по умолчанию этот параметр равен 60 сек. Другая опция, Content-Format Option, задается в виде числа, которое устанавливает формат представления полезной нагрузки: значение 0 указывает на то, что полезная нагрузка будет текстовой, а значение 41 указывает на то, что полезная нагрузка будет в формате xml. Для адресации по URI в протоколе CoAP предусмотрены опции Uri-Host (определяет адрес интернет-хоста запрашиваемого ресурса), Uri-Port (определяет номер порта транспортного уровня запрашиваемого ресурса), Uri-Path (определяет часть пути до ресурса) и Uri-Query (определяет параметр, который запрашивает ресурс).

 Например:

 Uri-Host = "example.net"

 Uri-Port = 5683

 Uri-Path = ".well-known"

 Uri-Path = "core"

 Uri-Query = "login?"

 coap://example.net:5683/.well-known/core/ login?

**У протокола СоАР определено всего четыре вида сообщений: Confirmable, Non-confirmable, Acknowledgement, Reset. Все сообщения кодируются в бинарном виде**

Список всех остальных опций протокола можно посмотреть в RFC7252, раздел 5.4. [4].

Поле, завершающее формат сообщения, – это полезная нагрузка (Payload), в которой содержится запрашиваемая информация.

В COAP используется четыре типа сообщений:

 Confirmable(CON) – сообщение, содержащее запрос или ответ, требующее подтверждения и считающееся надежным. Каждое сообщение Confirmable вызывает одно ответное сообщение подтверждения или сброса;

 Non-confirmable(NON) – сообщение, содержащее запрос или ответ, не требующее подтверждения и надежной передачи, так как передается регулярно (например, показания от датчика);

 Acknowledgement(ACK) – сообщение, подтверждающее, что пришло сообщение Confirmable. При этом само сообщение Acknowledgement не означает успех или неудачу любого из запросов, содержащегося в сообщении Confirmable;

 Reset(R) – сообщение сброса, указывающее на то, что конкретное сообщение (Confirmable или Non-confirmable) было получено, но некоторая часть текста отсутствует и невозможно правильно его обработать. Такая ситуация обычно возникает, когда принимающий узел перегружен. Вызов этого сообщения (например, путем отправления пустого сообщения Confirmable) также полезен в качестве проверки доступности узла ("CoAP ping").

**Примеры сценариев взаимодействия**

Рассмотрим несколько примеров того, как отправляются запросы и какие сообщения (их содержание) будут при этом отправлены. На рис. 4 показан пример взаимодействия между HTTP-клиентом и СоАР-датчиком и трансляцией их сообщений через CoAP-прокси. Пусть пользователь запрашивает показание освещенности кухни, которое определяется датчиком света.



**Все датчики, которые можно использовать для многоадресной рассылки, будут внесены в RD и формирование групп будет происходить по сообщенной датчиками информации**

HTTP-клиент посылает сообщение GET, в котором содержится запрос на получение уровня освещенности. Сообщение приходит на прокси, который преобразует HTTP-запрос в запрос по протоколу CoAP. Далее информация о требовании параметра освещенности передается в сообщении CON, и ему присваивается идентификатор сообщения; такой же идентификатор содержится в ответном сообщении ACK, которое несет в себе требуемую информацию.

Рассмотрим детально формат ответного сообщения ACK на запрос (рис. 5). В поле "тип" указывается, что передается сообщение ACK (2). Далее стоит 1 байт, который говорит о наличии маркера, за ним следует код, указывающий на успешный ответ (2.05).



Значение идентификатора сообщения одинаковое у запроса и ответа, что помогает их сопоставить. Значение маркера принимает значение 0х20 и не изменяется в пределах сессии. Опции в сообщении отсутствуют. Завершается сообщение передачей в поле полезной нагрузки запрашиваемого значения освещенности (150 Люкс).



На рис. 6 показан пример работы CoAP-клиента и СоАР-датчика. В данном примере рассмотрен случай, когда ответное сообщение от CoAP-клиента теряется, тогда будет произведена повторная отправка запроса после завершения таймера ожидания ответа; значение таймера равно 1 сек, согласно RFC 6298 [6].

**Многоадресная рассылка**

Протокол CoAP поддерживает возможность послать запрос сразу группе устройств, реализуя таким образом многоадресную рассылку (multicast) [7].

СoAP-устройства с ограниченными ресурсами могут объединятся в группы либо одинаковой функцией устройств (снятие показаний света или температуры), либо по местоположению (снятие показаний в определенной комнате, этаже здания). Есть несколько способов создания группы:

 устройство может быть предварительно запрограммированно под определенную группу;

 устройство может быть определено в группу через каталог ресурсов;

 устройство может определяться в группу с пользовательского устройства.

В первом случае задание группы происходит по групповому IP-адресу или по полному имени хоста, т.е. FQDN (Fully Qualified Domain Name), которому в соответствии с системой DNS сопоставляется групповой IP-адрес.

Во втором случае задание группы происходит через каталог ресурсов (RD – Resource Directory) [9]. RD хранит URI, по которым предоставляется доступ к датчикам. При помощи методов REST датчики регистрируются в каталоге и периодически обновляют информацию о себе.

По умолчанию все узлы принадлежат к группе "All CoAP Nodes". Членам этой группы назначается широковещательный адрес: для IPv4 это адрес 224.0.1.187 [10], а для IPv6 – это адреса FF02 :: FD и FF05 :: FD [11], UDP-порт, по умолчанию, 5683.

Для обнаружения адреса RD датчик посылает по широковещательному IP-адресу запрос с указанием типа ресурса "rt=core.rd", при этом сервер RD ответит на этот запрос, а все остальные устройства в сети его проигнорируют. Все запросы создаются с помощью формата связи CoRE (Constrained RESTful Environments) [8]. CoRE задает ссылку, по которой можно обратиться для внесения информации, поиска размещенных ресурсов и их атрибутов, связей между ресурсами в RD. По умолчанию точкой входа в каталог ресурсов является URI/"Well-Known/Core". Датчик отправляет методом REST запрос POST к RD, в который внесена информация о том, на какие запросы датчик сможет предоставить информацию, но внесение информации не обязательно, тогда датчик будет принимать любые запросы. Таким образом все датчики, которые можно использовать для многоадресной рассылки, будут внесены в RD и формирование групп будет происходить по сообщенной датчиками информации. Например, CoAP-клиент посылает запрос в RD на возможность формирования группы из датчиков. Такой запрос будет отправлен по CoAP-протоколу методом POST с URI в общем виде, который выглядит как:

URI: /{+rd-group}{?gp, d, con},

где rd-group – функция, указывающая, что необходимо сформировать группу;
gp – имя группы;
d – домен, к которому принадлежит группа;
con – контекст, используемый для установки собственного IP-адреса многоадресной рассылки для группы, в форме: // multicastaddress:port, указывать этот параметр необязательно.

**В статье рассмотрены такие важные особенности протокола, как формат сообщения, различные сценарии взаимодействия, способы группировки устройств с ограниченными ресурсами**

В третьем случае датчик определяется в группу пользовательским устройством, с которого отправляется запрос при помощи формата обмена данными Java Script Object Notation (JSON). Запрос будет послан сообщением CoAP при помощи методов REST с опцией Content-Format, равной 50, что устанавливает формат представления полезной нагрузки JSON. Полезная нагрузка будет состоять из одной или более пар ключей/значений, например, пара будет выглядеть так: "n":

"All-Devices.floor1.west.bldg6.test.com",
"a": "[ff15::4200:f7fe:ed37:abcd]:4567",

где ключ n – name является именем хоста, а значение a – address является значением адреса члена, обозначающим групповой IP-адрес.

CоAP-группа будет работать в режиме ASM (Any Source Multicast). При таком режиме работы адрес для группы уникален. Когда принадлежность датчиков к группе определена, пользовательское устройство посылает запрос по групповому IP-адресу широковещательно. Если датчик настроен на работу с таким групповым адресом, то он примет такой запрос (если не настроен, то просто отбросит).

На рис. 7 показан сценарий отправления запроса с использованием многоадресной рассылки. Многоадресная рассылка всегда осуществляется с помощью сообщения Non-confirmable.



Пользовательское устройство отправляет запрос на широковещательный IP-адрес, назначенный для группы, такой запрос достигает всех датчиков, которые настроены на такой адрес. В сценарии CoAP датчики 1, 3 и 5 настроены на IP-адрес группы, поэтому они принимают запрос и отвечают: CoAP-датчик 1 посылает успешный ответ (2.05), и он достигает сервера; CoAP-датчик 3 посылает успешный ответ, но при передаче он теряется; CoAP-датчик 5 не может дать ответ, так как временно недоступен (4.04); оставшиеся CoAP-датчики 2 и 4 не настроены на получение данного запроса и просто его отбрасывают.

**Заключение**

В статье рассмотрены такие важные особенности протокола, как формат сообщения, различные сценарии взаимодействия, способы группировки устройств с ограниченными ресурсами.

Рассмотренный материал позволяет сделать вывод о существенных достоинствах протокола CoAP, выделяющих его среди других протоколов Интернета вещей, например, в отличие от MQTT [13], у CoAP небольшой объем сообщений, простые алгоритмы их обработки, а так же поддержка многоадресной рассылки. Протокол СоАР ориентирован на М2М-приложения и обеспечивает взаимодействие сетей IoT с сетью Интернет через CoAP-прокси.

Протокол СоАР целесообразно использовать для устройств и сетей с ограниченными ресурсами.