УДК 004.7
DOI https://doi.org/10.32838/2663-5941/2020.2-1/15

Іванчук О.В.
Херсонський національний технічний університет

Завгородній В.В.
Херсонський національний технічний університет

Козел В.М.
Херсонський національний технічний університет

Дроздова Є.А.
Херсонський національний технічний університет

АНАЛІЗ ПРОТОКОЛІВ ОБМІНУ ДАНИМИ ДЛЯ КЕРУВАННЯ СИСТЕМАМИ ІНТЕРНЕТУ РЕЧЕЙ

У статті проведено аналіз наявних протоколів для обміну даними під час керування системами Інтернету речей. Розглянуто технологію M2M, що використовується для формування обміну даними за шаблоном «видавець» – «підписник». У такій системі «видавець» відправляє на сервер обробки пакети з даними, а «підписник» має повідомити, які дані бажає отримувати, й під час їхнього надходження обробляти їх. Під час розробки системи Інтернету речей необхідно обрати протокол обміну даними між пристроями й сервером керування. Основними протоколами керування є HTTP, SOAP, XMPP, STOMP, CoAP, MQTT, MQTT-SN. Для кожного протоколу розглянуто структуру пакету даних й основні типи пакетів. Для порівняльного аналізу протоколів як критерій було прийнято розмір пакету даних із повідомленням. Обрано однакові дані для кожного протоколу. На основі даних, які необхідно передати, сформовано пакети даних, що мають мінімальний набір необхідних даних у структурі пакету для обміну повідомленнями в системі. Для отриманих пакетів було пораховано кількість символів, що передаються. Результати свідчать, що протоколи XMPP та SOAP з XML-структурою пакету мають надмірність інформації через подвійний запис ідентифікатору параметру. Протоколи MQTT та MQTT-SN мають найменший розмір пакету через використання двійкового кодування параметрів, що дозволяє зменшити розмір пакету шляхом меншого розміру заголовків пакету й двійкового кодування числових значень. Оптимальним протоколом для систем Інтернету речей є протокол CoAP через посилання сервером даних до «підписників» відразу після отримання від «видавця». Протоколи MQTT і MQTT-SN через особливості реалізації не дозволяють миттєво оновлювати дані. Протоколи вимагають виконання запитів від «підписників» до сервера для перевірки наявності нових даних. Оскільки проміжки часу для виконання запитів можуть складати від декількох хвилин до декількох годин, то цей протокол не підходить для швидкого реагування на події. Протокол CoAP займає наступне місце після MQTT і MQTT-SN за розміром пакету.

Ключові слова: *віддалене керування, Інтернет речей, сервер керування, пакет, протокол.*

Постановка проблеми. Стрімкий розвиток Інтернету речей призвів до появи безлічі прикладних протоколів, необхідних для його реалізації. Архітектура Інтернету речей передбачає наявність таких функціональних рівнів: мережа датчиків, шлюз, управління, додаток [1]. Оскільки нижній рівень складається з датчиків і сенсорів, то відразу ж виникає необхідність в «особливих» протоколах для забезпечення взаємодії цих пристроїв із сервером. Стандартні прикладні протоколи не підходять через їхню непристосованість до умов мережі Інтернету речей. Датчик – зазвичай мініатюрний, з невеликою пам'яттю – вимірює фізичні параметри в режимі реального часу, найчастіше в умовах низького енергозабезпечення. Результати вимірювань обробляються вузлом і передаються на сервер. Обсяг інформації, що формується одним вузлом, порівняно невеликий, проте більшість сервісів Інтернету речей побудовано на принципі обробки інформації від безлічі вузлів, що принципово відрізняється від архітектур, прийнятих у класичних мережах типу абонент – вузол зв'язку для телефонії, клієнт-сервер для передачі даних.

Таким чином, з'являється нова архітектура: багато джерел – один одержувач. Крім того, обсяг трафіку від вузла може бути як дуже маленьким, так і дуже великим.

Аналіз останніх досліджень і публікацій. У XXI столітті ідея виконувати смарткерування будинком й інтернет речами набирає все більшої популярності. Збільшується попит на технології, які дозволяють керувати все більшою кількістю пристроїв [2]. Актуальним стає бажання користувача зробити свій будинок місцем відпочинку без необхідності виконувати прибирання, керувати безпекою, перевіряти наявність їжі і так далі.

Науковці Т.А. Москаленко, Р.В. Кірічек й А.Є Кучерявий розглядають протоколи, що використовуються під час розробки систем Інтернету речей [3]. Дослідники В.Ю. Гойхман й А.А. Савельєва проводять аналітичне порівняння протоколів Інтернету речей і визначають рекомендації по використанню кожного з них у різних частинах системи [1].

Інтернет речей (далі – IoT) розвивається швидкими темпами й незабаром увійде в усі сфери життєдіяльності людини, тому проблема вибору оптимального протоколу актуальна.

Постановка завдання. У зв'язку з тим, що нині існує безліч протоколів обміну даними між пристроями й сервером, поставлене завдання визначити протокол у системі Інтернету речей, який матиме мінімальний розмір пакету для забезпечення максимальної енергоефективності й мінімального часу обробки пакету IoT-пристроєм.

Виклад основного матеріалу дослідження. Для реалізації «Розумного будинку» використовують Інтернет речей (Internet of Things – IoT) – це пристрої, які увійшли в мережу й взаємодіють між собою без участі людини.

В основі усіх протоколів Інтернету речей лежить технологія M2M (Machine-to-Machine). Згідно із цією технологією, обмін даними виконується між машинами у двох- або односторонньому порядку.

У мережах Інтернету речей використовується шаблон обміну даними «видавець-підписник». У цьому шаблоні сервер обробки даних зберігає інформацію про наявних видавців і підписників. Коли видавець передає інформацію про подію, сервер визначає підписників, які мають отримати інформацію про подію, після чого виконує передачу даних.

Такий спосіб обміну даними дозволяє виконувати просте масштабування системи. Це виконується шляхом відсутності необхідності в зберіганні «видавцем» інформації про всіх «підписників», що дозволяє додавати або вилучати вузли із системи в реальному часі.

Для обміну даними в системах Інтернету речей використовуються такі протоколи: HTTP, SOAP, XMPP, STOMP, CoAP, MQTT, MQTT-SN [4; 5].

У пакеті протоколу HTTP тип запиту визначає, що клієнт бажає виконати з даними. Основними діями є:

- GET – отримати дані;
- POST – відправити дані на сервер;
- PUT – замінити дані на сервері;
- DELETE – видалити дані із сервера [5].

Ресурс у пакеті визначає ідентифікатор об'єкту, з яким виконуватиметься взаємодія.

Версія протоколу допомагає серверу визначити, чи зможе він обробити пакет. У нових версіях протоколу були додані нові обов'язкові поля в параметри обміну, які старі версії серверу не зможуть правильно обробити. Вказана версія дозволить одразу відхилити пакет.

У параметрах обміну вказується інформація про розмір даних; адреса, з якої відбувається підключення до сервера; необхідність підтримки відкритого каналу й інші.

Тіло повідомлення містить у собі дані, що необхідно обробити. Оскільки протокол передбачає передачу текстових даних, то будь-які дані форматуються в текст.

У протоколі SOAP заголовок містить у собі параметри обміну даними. Тіло повідомлення містить дані для обробки сервером [6]. Оскільки в протоколі відсутнє обов'язкове зазначення ресурсу для взаємодії в заголовку, то його розміщують у тілі повідомлення.

Усі дані форматуються в текст для передачі за протоколом SOAP.

Протокол XMPP вимагає вказувати в заголовку відправника й отримувача через відсутність постійного з'єднання між клієнтом і сервером [7]. У протоколі можуть використовуватися шлюзи для об'єднання мереж із різними протоколами, що не дозволяє постійно підтримувати відкритий канал зв'язку. Шлюз дозволяє виконати переадресацію згідно з даними, вказаними в заголовку пакету.

Тіло повідомлення в протоколі використовує текстове форматування, так само як HTTP і SOAP.

Основними типами повідомлень у протоколі STOMP є:

- SUBSCRIBE – клієнт повідомляє, з якого ресурсу бажає отримувати дані при їхньому оновлені;
- SEND – клієнт передає нові дані;
- MESSAGE – сервер відправляє клієнту оновлені дані з ресурсу, який було вказано через повідомлення типу SUBSCRIBE;

– ERROR – це повідомлення відправляє сервер при неможливості обробити дані [8].

У параметри додаються обов'язкові поля, що необхідно відправити в залежності від типу повідомлення. Також можна додати власні поля.

Тіло повідомлення містить дані для обробки сервером.

У пакеті CoAP версія протоколу необхідна для швидкої відмови в обробці даних через неможливість обробки нових типів повідомлення під час отримання пакету сервером, в якого версія протоколу старіша за версію клієнта [9].

Тип повідомлення визначає дію, що необхідно виконати над даними. Основними діями є:

– GET – отримання даних із вказаного ресурсу;

– PUT – оновлення даних у вказаному ресурсі;

– POST – перевірка наявності ресурсу на сервері й створення ресурсу в разі його відсутності;

– DELETE – видалення ресурсу з сервера [9].

Для кожного пакету задається унікальний номер, що зазначається в ідентифікаторі пакету. Це необхідно, оскільки протокол використовується над протоколом UDP, який не має алгоритмів підтвердження передачі даних. Наявність унікального ідентифікатора дозволяє відправити повідомлення у відповідь із підтвердженням вдалої передачі, вказуючи ідентифікатор пакету, який було отримано. Якщо сервер отримає пакет із тим же ідентифікатором, то це означає, що відповідь сервера не досягла клієнта.

Довжина ідентифікатора пакету визначає кількість байт, що займатиме ідентифікатор пакету.

Ідентифікатор повідомлення зберігає унікальний номер, що відповідає даним у тілі повідомлення. Це необхідно для перевірки наявності даних у системі. Якщо вони вже присутні або застарілі, то вони ігноруються.

Згідно з протоколом, ресурс, до якого йде звернення, задається в URL адресі сервера.

Основними типами повідомлення в пакетах протоколів MQTT і MQTT-SN є:

– CONNECT – запит клієнта на підключення до сервера;

– SUBSCRIBE – передача клієнтом до сервера інформації про ресурс, з якого клієнт бажає отримувати дані;

– PUBLISH – оновлення інформації в ресурсі [11].

Старші 4 біти першого байту заголовку є наступними флагами: для визначення повідомлень, що відправляються повторно; для визначення необхідності підтвердження передачі пакету; для визначення необхідності збереження даних після успішної їх передачі клієнтам.

Параметри повідомлення зберігають інформацію про розмір тіла повідомлення, ідентифікатор повідомлення, назву протоколу, його версію та параметри, що необхідні під час ініціалізації підключення.

Ідентифікатор ресурсу вказується в тілі повідомлення.

У MQTT-SN для обміну даними використовується протокол UDP замість TCP для зменшення розміру пакету. Також це дозволяє передати дані, не очікуючи підтвердження. Для пристроїв, дані яких не мають бути обов'язково передані, він дає можливість зменшити енерговитрати шляхом зменшення передач, що підтверджують успішну передачу пакету.

Для порівняння розміру пакетів були обрані такі дані для передачі:

– Число 15 з ідентифікатором int1;

– Число 11 з ідентифікатором int2;

– Число 2 004 з ідентифікатором int3;

– Число 1 959 з ідентифікатором int4;

– Текст Khersonal National Technical University з ідентифікатором text;

– Ідентифікатор ресурсу /send/info;

– Ідентифікатор серверу iot.server;

Рис. 1. Гістограма розміру тіла повідомлення

– Ідентифікатор клієнту iot.client.

Кожен рядок закінчується послідовністю ASCII символів '\r\n'. Послідовність займає 2 байти. Для кодування даних у двійковій формі використовується система кодування ASCII, що визначає розмір 1 байт для кожного символу.

Через використання мови розмітки XML у тілі пакету XMPP і SOAP, для зазначення будь-якого параметру необхідно двічі зазначати ідентифікатор параметру. Це викликає надмірність інформації для передачі. У заданому прикладі кожен параметр потребує додаткову передачу 7 байтів.

На рисунку 1 зображено гістограму розмірів тіла пакету для протоколів із текстовим форматуванням і XML форматуванням.

Для кожного з протоколів було складено мінімальний пакет. Кожен пакет використовується для оновлення інформації на серверах.

Протоколи CoAP, MQTT і MQTT-SN використовують бінарне кодування для заголовків. Протоколи MQTT і MQTT-SN використовують бінарне кодування для даних. Бінарні дані представлені за допомогою шістнадцятирічних чисел.

На рисунку 2 зображено гістограму розмірів пакетів у байтах за протоколами.

Згідно з отриманими результатами мінімальний розмір пакету виявився в протоколі MQTT. Але цей протокол потребує додатково відправляти пакет з'єднання до передачі пакету з новими даними, що збільшує об'єми даних.

На рисунку 3 зображена гістограма розміру пакетів даних у протоколах з урахуванням пакету з'єднання для протоколу MQTT.

Після додавання пакету з'єднання кількість даних, що передаються за протоколом MQTT, залишається найменшою. Це досягається шляхом двійкового кодування даних, що дозволило зменшити розміри заголовку пакету до 2 байтів.

Використання двійкового кодування даних дозволяє зменшити розміри пакету під час передачі числових значень. Можна використовувати числові ідентифікатори параметрів, що зменшить їхній розмір. Можна отримати 256 ідентифікаторів, використовуючи лише один байт даних. Також можливо закодувати числа від 0 до 255 одним байтом. Під час передачі тризначних чисел це дозволить зменшити розміри пакету на 2 байти з кожного числа.

Під час передачі 256 тризначних чисел двійкове кодування дозволить зменшити розміри пакету на $256 \times 2 = 512$ байтів.

Оскільки більшість даних із пристроїв мають числові значення, то двійкове кодування оптимальне в протоколах систем Інтернету речей.

Заголовок пакету в протоколі CoAP також кодується у двійковій формі, але він має більше параметрів. Також протокол вимагає розміщення ідентифікатора ресурсу в окремий параметр одразу після заголовку. Опис параметру вимагає вказувати ідентифікатор параметру, що не кодується двійковим кодом, а використовує текстовий опис. Опис параметру потребує 9 байтів даних. У протоколі MQTT використовується додатково 2 байти перед описом ідентифікатору ресурсу для зазначення довжини ідентифікатору.

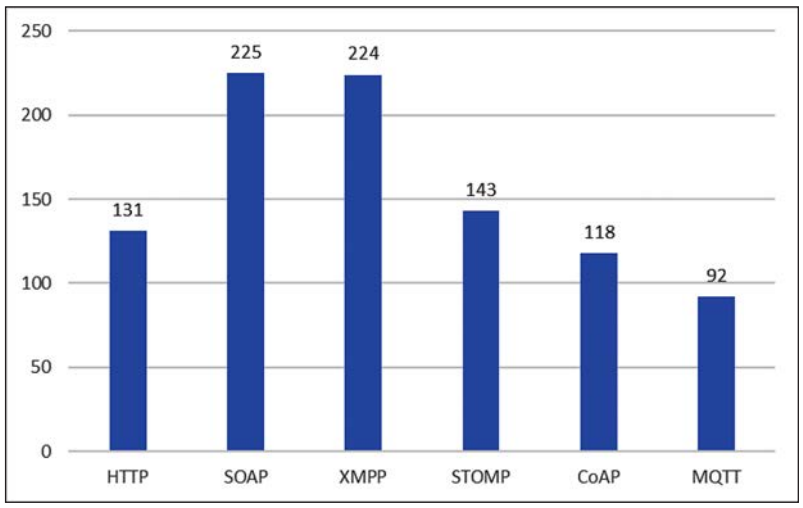

Рис. 2. Розмір пакету даних у протоколах

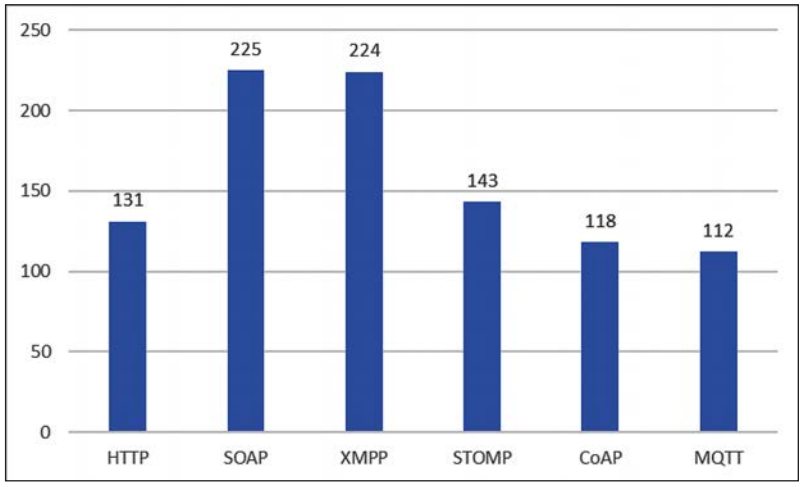

Рис. 3. Розмір пакетів у протоколах із урахуванням пакету з'єднання MQTT

Перевагою протоколу CoAP є схожість із протоколом HTTP, що дозволяє використовувати архітектуру REST при побудові серверів.

Також перевагою цього протоколу відносно протоколу MQTT є миттєве оновлення даних. У протоколі MQTT для оновлення даних на пристрої, для якого ці дані потребуються, пристрій має виконати підключення до сервера й отримати нові дані. Проміжки часу, через які виконується оновлення даних, залежать від конфігурації пристрою, через що час отримання команди для виконання може коливатися від декількох секунд до декількох годин, що не дозволить використовувати протокол MQTT у системах швидкого реагування. Протокол CoAP вимагає постійного очікування пристроєм підключення до сервера для оновлення даних.

Висновки. Було розглянуто пакети даних наявних протоколів для обміну даними в системах Інтернету речей. Отримані такі результати:

− протоколи, що використовують мову розмітки XML, для опису тіла повідомлення мають надмірність інформації в повідомленні;

− протоколи з двійковим кодуванням даних дозволяють зменшити розміри повідомлення під час передачі чисельних значень;

− протоколи MQTT і MQTT-SN мають мінімальний розмір пакету шляхом двійкового кодування даних і мінімального розміру заголовку. Мінімальний розмір пакету зменшує час передачі даних. Це зменшує енерговитрати в автономних системах;

− протокол CoAP шляхом двійкового кодування заголовку й схожості з HTTP протоколом дозволяє розробляти системи з REST архітектурою;

− оптимальним протоколом для керування системами Інтернету речей є CoAP через найменший розмір пакету даних серед протоколів, що одразу виконують оновлення даних на пристроях.

Список літератури:

1. Гойхман В.Ю., Савельєва А.А. Аналитический обзор протоколов Интернета вещей. *Технологии и средства связи*. 2016. № 4. С. 32–37. URL: http://lib.tssonline.ru/articles2/reviews/analiticheskiy-obzor-protokolov-interneta-veschey (дата звернення: 26.02.2020).

2. Knud Lasse Lueth. IoT 2019 in Review: The 10 Most Relevant IoT Developments of the Year. *Iot-analytics* : web-site. URL: https://iot-analytics.com/iot-2019-in-review/ (дата звернення: 26.02.2020).

3. Москаленко Т.А., Киричек Р.В., Кучерявий А.Е. Обзор протоколов Интернета вещей. *Информационные технологии и телекоммуникации*. 2017. № 2. С. 1–12. URL: http://www.sut.ru/doci/nauka/review/20172/1-12.pdf (дата звернення: 28.02.2020).

4. Протоколы передачи данных. *Википедия: свободная энциклопедия.* URL: https://iot.ru/wiki/protokoly-peredachi-dannykh-iot (дата звернення: 26.02.2020).

5. Fielding R., Reschke J. Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing. URL: https://httpwg.org/specs/rfc7230.html#http.message (дата звернення: 26.02.2020).

6. XML Soap. *W3schools* : web-site. URL: https://www.w3schools.com/xml/xml_soap.asp (дата звернення: 26.02.2020).

7. XML Stanzas. Extensible Messaging and Presence Protocol (XMPP): Core. URL: https://xmpp.org/rfcs/rfc6120.html#stanzas (дата звернення: 26.02.2020).

8. STOMP Protocol Specification, Version 1.2. URL: https://stomp.github.io/stomp-specification-1.2.html (дата звернення: 26.02.2020).

9. RFC 7252. The Constrained Application Protocol: Method Definitions. URL: https://tools.ietf.org/html/rfc7252#section-5.8 (дата звернення: 26.02.2020).

10. Azzola F. CoAP Protocol: Step-by-Step Guide. URL: https://dzone.com/articles/coap-protocol-step-by-step-guide (дата звернення: 26.02.2020).

11. MQTT Packet Structure. URL: http://www.bytesofgigabytes.com/mqtt/mqtt-protocol-packet-structure/ (дата звернення: 26.02.2020).

Ivanchuk O.V., Zavhorodnii V.V., Kozel V. M., Drozdova Ye.A. **ANALYSIS OF DATA SHEET PROTOCOLS FOR CONTROL OF THE INTERNET OF THINGS**

The article analyzes existing protocols for data exchange while managing Internet of Things systems. The M2M technology used to form data exchange on the system of "publisher" – "subscriber" is considered. In such a system, the "publisher" sends packets of data to the processing server, and the "subscriber" has to inform what data he wants to receive and when they receive them to process them. When designing the Internet of Things system, you need to choose the communication protocol between the devices and the management server. The main management protocols are HTTP, SOAP, XMPP, STOMP, CoAP, MQTT, MQTT-SN.

For each protocol, the structure of the data packet and the basic types of packets are considered. The size of the message packet was selected for comparative analysis of the protocols. For comparison, the same data for each protocol was selected. Based on the data to be transmitted, the generated packets have the minimum set of required data in the packet structure for messaging on the system. The number of transmitted characters was calculated for the received packets. The results show that XMPP and SOAP protocols that have XML packet structure have redundant information through dual parameter identifier entry. The MQTT and MQTT-SN protocols have the smallest packet size due to the use of binary parameter encoding, which reduces the packet size due to the smaller packet headers and binary encoding of numeric values. The best protocol for Internet of Things systems is the CoAP protocol, by sending data to "subscribers" by the server as soon as it is received from the "publisher". The MQTT and MQTT-SN protocols do not allow the data to be updated instantly due to implementation features. Protocols require requests from "subscribers" to the server to check for new data. Since queries can take between minutes and hours, this protocol is not suitable for quick response to events. CoAP ranks second after MQTT and MQTT-SN in packet size.

Key words: *remote control, Internet of Things, management server, packet, protocol.*