Тема 14. Протоколи Інтернету речей

Протоколи інфраструктури

Обсяг інформації, що формується одним сенсорним вузлом, порівняно невеликий, однак більшість сервісів Інтернету речей побудовано на принципі обробки інформації від безлічі вузлів, що принципово відрізняється від архітектур, прийнятих в класичних мережах, типу абонент - вузол зв'язку для телефонії, клієнт - сервер для передачі даних.

Таким чином, ми стикаємося з новою архітектурою: багато джерел - багато одержувачів, крім того, обсяг трафіку від сенсорного вузла може бути як дуже маленьким, так і дуже великим. Звичні прикладні протоколи для передачі повідомлень не розраховані на таке використання.

Рисунок 14.1 – Протоколи архітектури IoT

Протокол маршрутизації RPL

RPL означає протокол маршрутизації для мереж низької потужності та трат. Це протокол IPv6. Мережі з низьким рівнем втрат потужності включають в себе безпроводові локальні мережі (WPAN), мережі низьковольтних лінійних зв'язків (PLC) та мережі безпроводових датчиків (WSN).

Ці мережі мають деякі характеристики:

– Можливість оптимізувати та заощадити енергію

– Можливість підтримувати схеми трафіку, відмінних від одноадресного спілкування
– Можливість запускати протоколи маршрутизації через шари каналів з обмеженими розмірами кадрів

Рисунок 14.2 – Категорії протоколів IoT

RPL був розроблений, щоб підтримувати мінімальні потреби в маршрутизації шляхом створення високоточної топології над мережами з втратами. Цей протокол надає підтримку різноманітних типів моделей трафіку: багатоточкове, точка-багатоточка та точка-точка. Пристрої в мережі, що використовують цей протокол, підключаються один до одного таким чином, щоб у цьому з'єднання не було циклів. Для досягнення цього спочатку споруджується вузол, який називається цільовим орієнтованим ациклічним графіком (destination oriented directed acyclic graph, DODAG), який перенаправляється на одне призначення. Специфікації RPL звертаються до DODAG як до основи DODAG. Кожний вузол, який входить до складу DODAG, знає свій головний вузол, але не має інформації про його дочірні вузли. RPL підтримує щонайменше один шлях від кожного вузла до кореневого і до бажаного батьківського.

Це зроблено для підвищення продуктивності пошуку швидшого шляху. Топологія DODAG, що використовується в RPL, зображена на рис. 14.3.:

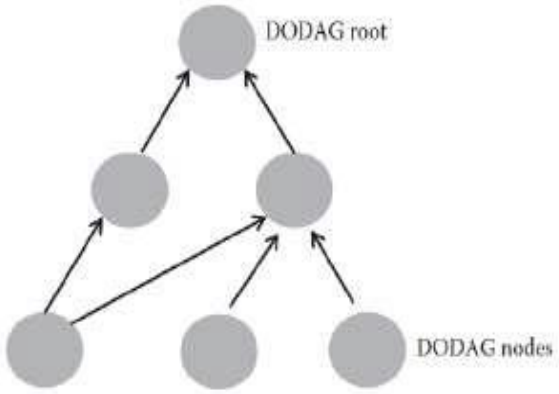

Рисунок 14.3 – Протоколи інфраструктури

Маршрутизатори RPL працюють в одному з двох режимів роботи (MOP): режим не зберігання або зберігання.

У режимі не зберігання повідомлення маршруту RPL рухаються у бік нижчих рівнів на основі маршрутизації джерела IP, тоді як у режимі зберігання маршрутизація вниз здійснюється на основі адресі призначення IPv6.

IEEE 802.15.4

Цей протокол був створений для того, щоб вказати підрівні для MAC та фізичного рівня, насамперед, для низькошвидкісних бездротових приватних мереж. Враховуючи різноманітні переваги, пропоновані цим протоколом, такі як низьке енергоспоживання, низька швидкість передачі даних, а також низька вартість та висока пропускна здатність повідомлень, вона дуже підходить для використання в системах IoT як протокол зв'язку. Цей протокол також забезпечує надійне з'єднання і може обробляти величезну кількість вузлів (приблизно близько 65K вузлів). Ідеально підходить для забезпечення зв'язку, оскільки забезпечує високий рівень безпеки, шифрування та служби автентифікації. Єдиною негативною стороною цього протоколу є те, що вона не забезпечує жодної з QoS(quality of service) гарантій.

Рисунок 14.4 – Архітектура IEEE 802.15.4

Цей протокол ґрунтується на ZigBee та інших протоколах, що використовуються в IoT-комунікації. IEEE 802.15.4 підтримує передачу у трьох частотних діапазонах, використовуючи метод DSSS(direct sequence spread spectrum).

На основі частотного каналу, передача даних відбувається в три рази швидкість передачі даних:

– 250 kbps at 2.4 GHz;
– 40 kbps at 915 MHz;
– 20 kbps at 868 MHz.

Цей протокол підтримує два типи вузлів мережі:

– Повнофункціональні пристрої (FFD);
– Знижено функціональні пристрої (RFD).

FFD можуть працювати як координатор персональної зони (PAN) або просто як звичайний вузол. Координатор має можливість створювати, керувати та підтримувати мережу. FFDs можуть зберігати таблицю маршрутизації у своїй пам'яті і можуть забезпечити MAC. Вони також можуть спілкуватися з іншими пристроями, використовуючи одну з наступних топологій:

– зірка;
– однорангова;
– кластерне дерево.

RFD - це дуже прості вузли, і вони мають обмежені ресурси. Вони можуть спілкуватися тільки з вузлом координатора, використовуючи тільки топологію зірки.

Топологія зірок: містить принаймні один FFD та кілька інших RFD. FFD, призначений для роботи в якості координатора PAN, повинен бути розташований у центрі мережі. Цей FFD несе відповідальність за управління та контроль усіх інших вузлів, які є частиною мережі (рис.).

Топологія одноранговой мережі: вона містить координатора PAN, а інші вузли зв'язуються між собою в тій самій мережі або через проміжні вузли до інших мереж.

Топологія кластерного дерева: це особливий тип одноранговых топології. Він складається з координатора PAN, кластерної голови та нормальних вузлів.

IPv6 over Low-Power Wireless Personal Area Networks(6LoWPAN)

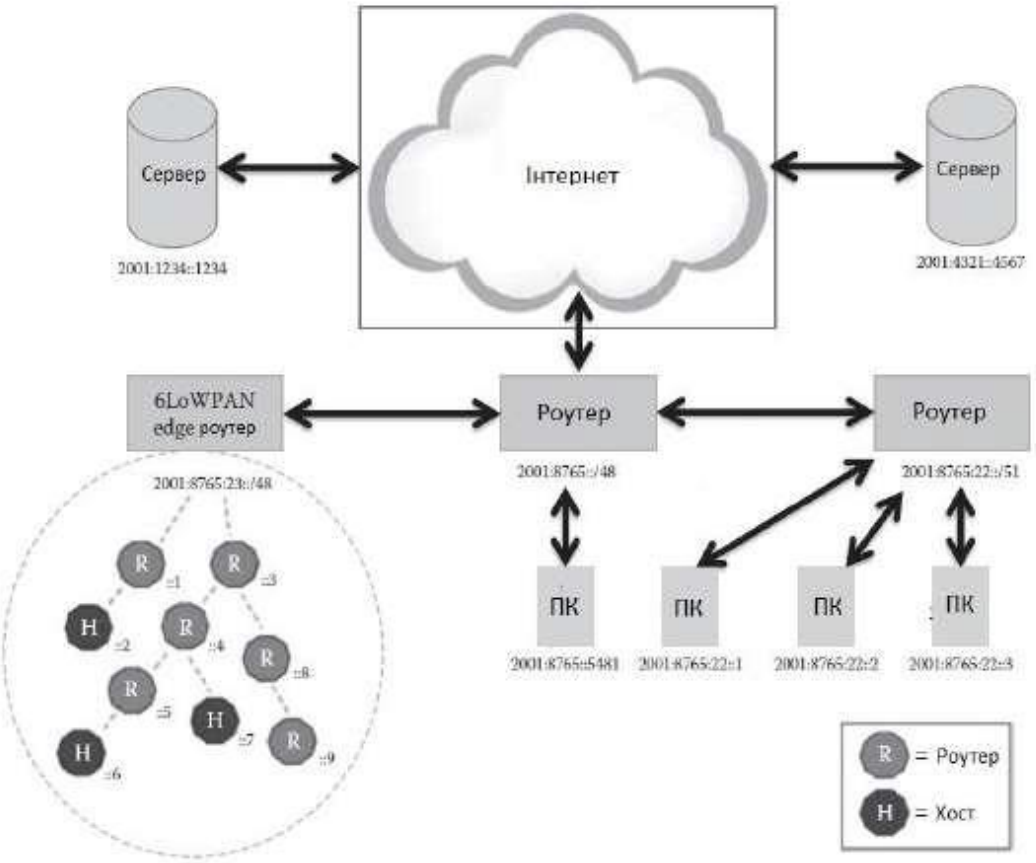

Рисунок 14.5 – Архітектура 6LoWPAN

Вихідна лінія до Інтернету забезпечується точкою доступу (AP, access point), яка в цьому випадку є маршрутизатором IPv6. Різні типи пристроїв, таких як ПК та сервери, можуть бути підключені до AP. Компоненти мережі 6LoWPAN підключаються до мережі IPv6 за допомогою маршрутизатора 6LoWPAN. Нижче наведено функції, які виконує «edge» маршрутизатор:

- Це дозволяє обмінюватися даними між пристроями 6LoWPAN та Інтернетом (або іншою IPv6 мережею).
- Це дозволяє обмінюватися даними між пристроями, що входять до мережі 6LoWPAN.
- Це допомагає генерувати та підтримувати мережу 6LoWPAN.

Оскільки мережі 6LoWPAN можуть спілкуватися з IP-мережами, вони підключаються до IP-мереж просто за допомогою IP-маршрутизаторів.

«Edge» маршрутизатори, які використовуються для підключення мереж 6LoWPAN до інших IP-мереж, передають IP-датаграми між різними носіями, що використовуються в IP- мережах. Медіа, що використовується в мережі IP, може бути Ethernet, Wi-Fi, 3G або 4G. Оскільки «edge» маршрутизатори, що використовуються в мережевих датаграмах мережі 6LoWPAN для інших IP-мереж із використанням мережевого рівня, вони не підтримують стан прикладного рівня. Це, в свою чергу, знижує робоче навантаження на «edge» маршрутизаторі з точки зору потужності обробки, що дозволяє використовувати дешеві вбудовані пристрої з простим програмним забезпеченням.

Bluetooth Low Energy

Bluetooth Low Energy (BLE) спочатку працював як частина базової специфікації Bluetooth 4.0.

BLE використовує радіоприймач малої дальності з мінімальною потужністю і працює довгий час. Його діапазон охоплення становить близько 100 метрів, що приблизно в 10 разів перевищує звичайний Bluetooth.

Затримка BLE в 15 разів менша за звичайну Bluetooth. BLE працює, використовуючи потужність від 0,01 мВт до 10 мВт. Ці характеристики роблять BLE ідеальним протоколом для використання пристроями IoT.

EPCglobal

RFID (радіочастотна ідентифікація) пристрої - безпроводові мікрочіпи, які використовуються для позначення об'єктів для автоматичної ідентифікації.

Електронний код продукту (EPC) - це унікальний ідентифікатор, що зберігається в тезі RFID, що допомагає ідентифікувати та відслідковувати елементи в сценарії керування ланцюжком постачання. EPCglobal – це організація, що розробила EPC, а EPCglobal також готує та підтримує стандарти, пов'язані з RFID та EPC. RFID може бути використана як ключова технологія для пристроїв IoT з наступних причин:

- Відкритість;
- Масштабованість;
- Надійність;
- Підтримка ідентифікаторів об'єктів та відкриття сервісів.

Тег RFID має дві основні компоненти: електронний мікросхеми для зберігання ідентичності об'єкта та антени, що дозволяє чіпу спілкуватися з системою читання тегів. Зв'язок між тегом і читачем тегів відбувається за допомогою радіохвиль. Два основних компоненти системи RFID:

- радіоприймач;
- читач тегів.

Z-Wave

Z-Wave - це протокол бездротового зв'язку з малою потужністю, який використовується переважно для домашніх мереж (HAN, home area networks).

Він має широке застосування в розробці програм дистанційного керування для розумних будинків, а також інших невеликих комерційних областей. Z-Wave була розроблена компанією ZenSys, а пізніше вдосконалена альянсом Z-Wave.

Z-Wave працює переважно в частотному діапазоні біля ГГц, що зазвичай становить близько 900 МГц.

Цей протокол використовує топологію мережевої сітки з малою потужністю. Кожен вузол або пристрій, що входить до складу мережі, має можливість надсилати та отримувати команди керування через стіни та поверхи будинку, і вони використовують проміжні вузли для маршрутизації даних навколо перешкод, які можуть бути присутніми в будинку. Складається мережа з контролерів та підпорядкованих пристроїв.

ZigBee

Протокол ZigBee був обрамлений альянсом ZigBee. Наступні особливості ZigBee

роблять його дуже придатним для застосування IoT:

- Низьке енергоспоживання
- низька вартість
- Підтримка великої кількості вузлів мережі (<= 65K вузлів)

Крім особливостей, перерахованих вище, ZigBee має децентралізовану топологію мережі, яка дуже подібна до Інтернету. Цей протокол має можливість, яка дозволяє вузлам знаходити нові маршрути, якщо один маршрут не працює в мережі. Ця функція робить його дуже надійним бездротовим протоколом .

Специфікація ZigBee використовує нижні шари стека протоколу IEEE 802.15.4 і визначає власні верхні шари від мережі до програми.

Протоколи виявлення сервісів Multicast Domain Name System (mDNS)

mDNS - це служба, яка може працювати як унікальний DNS-сервер. Цей підхід дуже гнучкий через те, що простір імен DNS можна використовувати локально без будь-якої додаткової конфігурації. mDNS - це вигідний вибір для вбудованих пристроїв на базі Інтернету з наступних причин:

- Для керування пристроями не потрібна ручна настройка або адміністрування.

– Можна запустити без будь-якої додаткової інфраструктури.
– Високий рівень відмовостійкості через здатність функціонувати, навіть якщо відбудеться несправність інфраструктури.

DNS Service Discovery

Цей протокол допомагає клієнтам знаходити набір необхідних послуг, які присутні в мережі за допомогою стандартних DNS-повідомлень. Цей протокол також допомагає підключати пристрої без зовнішнього адміністрування або конфігурації. Виявлення служби DNS (DNS-SD) зазвичай використовує mDNS для надсилання пакетів DNS до певних адрес мультимовлення за допомогою UDP. Робота сервісу - це двоетапний процес:

1. Пошук назв вузлів необхідних служб.
2. Об'єднання IP-адреси з іменами хостів, використовуючи mDNS.

Universal Plug and Play (UPnP) - це набір мережевих протоколів, який був розроблений форумом UPnP. Основні особливості UPnP, що робить його придатним для сервісного виявлення пристроїв IoT, є наступні:

– Можливість підключення пристрою UPnP до мережі динамічно (автоматично) та отримання IP-адрес інших пристроїв і одночасно передавати свої можливості на інші пристрої.
– Конфігурація та адміністрування з нуля.

Протоколи рівня додатків

Представлена топологія на рис. 14.6 відповідає шаблоном проектування передачі повідомлень, який має назву "видавець-підписник“ (Publisher-Subscriber, або pub/sub). У такій схемі вводиться поняття видавця – джерела інформації та передплатника – одержувача інформації. Термін підписка пов'язаний з певною операцією, виконаною учасниками, з метою отримання інформації передплатником від конкретного видавця, а також упорядкування збору інформації – параметрів періодичності отримання та аналогічних (в залежності від реалізації) показників.

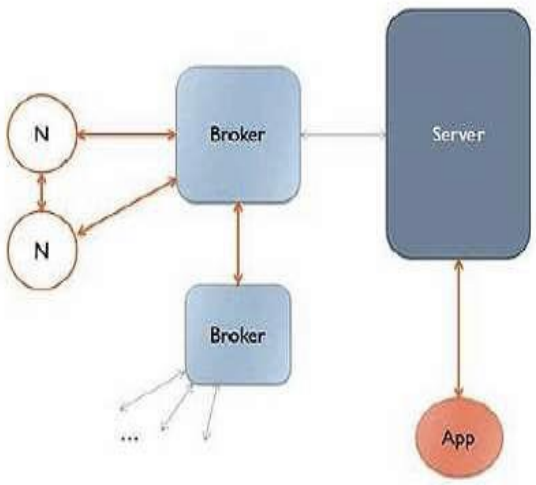

Рисунок 14.6 – Базова топологія, яка використовується для передачі повідомлень в IoT

В даному випадку розглядається ситуація, коли сенсорний вузол (Node) об'єднує інформацію від багатьох датчиків (наприклад, дані вологості повітря) і направляє її згідно з параметрами передплати або за запитом, або самостійно через певний інтервал часу. Зазвичай самі датчики досить примітивні, їх завдання зводяться до постійної передачі інформації про контрольовані параметри. Тому з'являється необхідність об'єднувати датчики в вузли, оснащені мікроконтролерами, які будуть відповідати за зчитування вимірюваних даних і відправку їх за заздалегідь визначеними алгоритмами далі на сервер. Також найчастіше для взаємодії клієнта з системою необхідна ще клієнтська програма (Application), встановлена на персональному пристрої, яка необхідна для наочного представлення одержуваної від датчиків, або вже обробленої сервером інформації та управління системою. Така топологія також розрахована на включення брокера (Broker).

Брокер – це сервер, який приймає інформацію від видавців і передає її відповідним передплатникам, в складних системах може виконувати, також різні операції, пов'язані з аналізом та обробкою даних, що надійшли на сервер. Брокер може встановлювати пріоритети сполученням і формувати черги для передачі повідомлень. Таким чином брокер організовує пересилання повідомлень, їх

зберігання та фільтрацію. Під чергою повідомлень розуміється контейнер, або блок, в якому зберігаються повідомлення в процесі їх пересилання. При недостатньому ресурсі каналу зв'язку, або якщо одержувач недоступний під час того, як надсилається повідомлення, черга зберігає повідомлення до тих пір, поки воно не буде доправлено до відправника.

Протокол DDS

DDS (Data Distribution Service) - протокол прикладного рівня M2M для систем реального часу. Базується на моделі "видавець-передплатник".

Основна функція протоколу полягає в тому, щоб здійснити з’єднання пристроїв з іншими пристроями за допомогою шини обміну повідомленнями (див. рис. 14.7). Протокол DDS може ефективно та синхронно доставляти мільйони повідомлень в секунду. Пристрої дають запит на дані інакше, ніж в IT мережах.

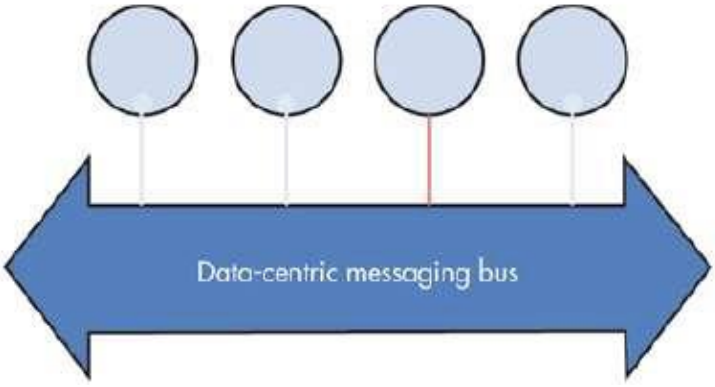

Рисунок 14.7 – Принцип з’єднання пристроїв за допомогою протоколу DDS в IoT

По-перше, пристрої працюють швидко. Масштаб реального часу часто вимірюється в долях мікросекунд. Пристроям потрібно здійснювати зв'язок з іншими пристроями, використовуючи складні шляхи, тому прості і надійні двоточкові TCP потоки даних обмежують можливості для такої передачі.

Натомість DDS забезпечує деталізований контроль якості обслуговування (QoS), багатоадресну передачу, переналаштовану надійність і всеосяжну надмірність. Крім того, сильною стороною DDS є розгалуження даних.

Протокол DDS забезпечує потужні способи фільтрації та відбору даних за адресами призначення, причому число синхронних одержувачів даних може обчислюватися тисячами. Деякі пристрої досить компактні, тому існують полегшені версії протоколу DDS, які працюють в умовах обмеженого обсягу.

Для використання даних від пристроїв зіркоподібна мережа зовсім не годиться. Замість цього DDS реалізує пряму шинний зв'язок між пристроями на базі реляційної моделі даних. ЇЇ називають шиною даних (DataBus), оскільки це мережевий аналог бази даних (database).

Високопродуктивні системи інтегрованих пристроїв використовують протокол DDS. Це єдина технологія, яка забезпечує гнучкість, надійність та швидкість, необхідні для побудови складних додатків реального часу. Ці додатки містять в собі військові системи, вітроелектростанції, інтегровані системи лікарень, системи діагностичної візуалізації, системи супроводження ресурсів і автомобільні системи випробувань і забезпечення безпеки.

Протокол XMPP

XMPP (eXtensible Messaging and Presence Protocol) - відкритий, заснований на XML, вільний для використання протокол для миттєвого обміну повідомленнями та інформацією про присутність в режимі, близькому до режиму реального часу.

Спочатку спроектований легко розширюваним, протокол, крім передачі текстових повідомлень, підтримує передачу голосу, відео та файлів через мережу.

В протоколі XMPP використовується текстовий формат XML в якості вбудованого типу, забезпечуючи природний зв'язок між людьми. Протокол працює по TCP, або за допомогою HTTP поверх TCP. Його перевагою є метод адресування за допомогою ідентифікаторів Jabber ID, який містить в собі

наступні, такі компоненти як вузол, домен та ресурс, причому два останні компоненти не є обов’язковими.

Адреса має такий вид username@gmail.com, який допомагає з'єднувати користувачів у величезному просторі Інтернету. XMPP підтримує різні комунікаційні моделі (запит- відповідь, публікація, підписка та інші)[6].

XMPP забезпечує простий спосіб адресуванні пристроїв. Це особливо зручно, коли дані передаються між віддаленими, найчастіше незалежними точками, як у випадку зв'язку між двома абонентами. Цей протокол не володіє високою швидкістю. Фактично, в більшості реалізацій цього протоколу використовується метод опитування або перевірки доповнень тільки на вимогу.

Сильними сторонами цього протоколу також є безпека, масштабованість, тому він ідеально підходить для невеликих мереж Інтернету речей.

Протокол CoAP

CoAP (Constrained Application Protocol) - це спеціалізований протокол передачі, розроблений робочою групою IETF-CORE, створений для мереж і пристроїв з обмеженими ресурсами, M2M додатків. CoAP можна розглядати як доповнення до HTTP, але на відміну від HTTP CoAP націлений на використання в пристроях з певними обмеженнями. CoAP використовує транспортний протокол UDP.

Повідомлень, використовуваних протоколом CoAP, не так багато, більшість з них це запити відповіді: GET(отримати інформацію з приводу ресурсу), PUT(задати нове завдання над ресурсом), POST(зміна дій над ресурсом), DELETE(видалити активні можливості ресурсу), CONNECT(з’єднання). Клієнти (додатка користувача) використовують повідомлення для управління і спостереження за ресурсом. За запитом встановлюється прапор спостереження, і сервер продовжує відповідати після того, як початкове повідомлення було передано. Це дозволяє серверам організовувати потокову передачу змін станів датчиків.

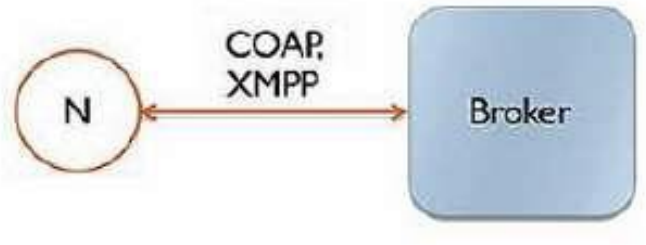

Рисунок 14.8 – Сегмент мережі де використовується протокол CoAP та XMPP

Таким чином, на ділянці мережі між сенсорним вузлом і брокером для забезпечення їх зв'язку, з метою реєстрації та конфігурації вузлів, а також для передачі інформації найчастіше застосовуються два протоколи - XMPP та CoAP. Фактичний протокол протоколу залежить від умов, які реалізується на мережі. Можна відзначити, що XMPP знайшов своє застосування в системах освітлення і клімату, а також використовується для адресування пристроїв в невеликих персональних мережах.

Очевидно, що CoAP призначений для пристроїв з обмеженими ресурсами й для мереж з низьким енергоспоживанням. Відомо застосування протоколу в системах датчиків температури та інших датчиків розумного будинку.

Протокол STOMP

STOMP - Simple (або Streaming) Text Oriented Message Protocol - простий протокол обміну повідомленнями, що передбачає широку взаємодію з багатьма мовами, платформами та брокерами. Даний протокол підходить під шаблон "видавець-передплатник" і за допомогою повідомлень SEND (відправити), SUBSCRIBE (підписатися), UNSUBSCRIBE (відписатися), BEGIN (почати), ABORT (переривати), ACK (підтвердити), NACK(не підтвердити), ISCONNECT(відключити) організовує зв'язок з брокером за методом "запит-відповідь".

Цей протокол використовується в тих випадках, коли необхідно застосування простого протоколу передачі повідомлень в мережах, що мають обладнання різних платформ.

Протокол схожий на HTTP, використовує транспорт TCP, є простим текстовим протоколом, що дозволяє клієнтам STOMP спілкуватися з будь-яким брокером повідомлень, котрі підтримують цей протокол.

Таким чином, цей спосіб взаємодії, розроблений для обміну повідомленнями між платформою, описаною одною мовою програмування, і клієнтом, програмне забезпечення якого розроблене іншою мовою. Підтримує велику кількість сумісних клієнтських бібліотек.

Треба відзначити, що для забезпечення роботи брокера в мережі Інтернету речей можливе використання обох протоколів: MQTT і STOMP. Тільки протокол STOMP орієнтований тільки на взаємодію брокера з сервером, а протокол MQTT забезпечує "наскрізний“ зв'язок, як від брокера до сенсорних вузлів, так і від брокера до сервера.

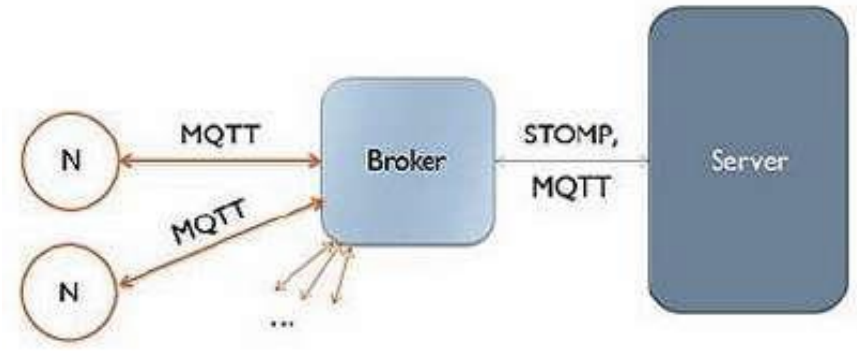

Рисунок 14.9 – Сегмент мережі де використовується протокол MQTT та STOMP

Протокол SOAP

SOAP (Simple Object Access Protocol) – протокол обміну сруктурованими та довільними повідомленнями формату XML в розподіленому обчислювальному середовищі. SOAP використовує базову модель з'єднання, що

забезпечує узгоджену передачу повідомлення від відправника до одержувача, потенційно допускає наявність посередників, які можуть обробляти частина повідомлення або додати до нього додаткові елементи.

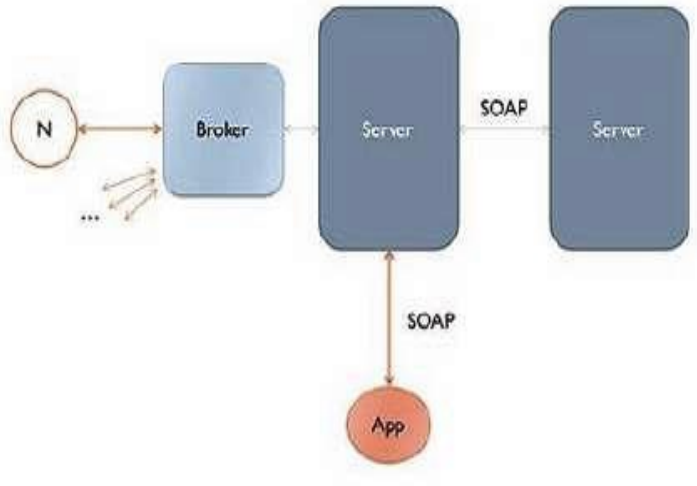

Рисунок 14.10 – Сегмент мережі де використовується протокол SOAP SOAP підтримує два механізми доступу - SOAP MESSAGE та SOAP RPC.

SOAP MESSAGE - це протокол для відправлення та обробка SOAP повідомлень, заснованиа на об'єкті Message. Може використовуватися для асинхронних комунікацій та має на увазі негайну, або відкладену відповідь на запит.

SOAP RPC являє собою простий протокол "запит-відповідь", який базується на об'єкті Call. Цей об'єкт використовується для синхронного віддаленого виклику процедур за допомогою XML.

Завдяки декільком повідомленням (GET, SOAP ACTION–RESPONSE, SOAP ACTION), який передбачає запит-відповідь, протокол може використовуватися з будь-яким протоколом прикладного рівня: FTP, HTTPS, SMTP.

Протокол MQTT

MQTT (Message Queue Telemetry Transport) - це легкий, компактний і відкритий протокол обміну даними створений для передачі даних на віддалених локаціях, де потрібний невеликий розмір коду і є обмеження до пропускної здатності каналу. Перераховані вище вимоги дозволяють застосовувати його в системах M2M (машина-машина).

MQTT був внутрішнім і пропрієтарним протоколом для IBM протягом багатьох років, поки не був випущений у версії 3.1 в 2010 р в якості безкоштовного продукту. У 2013 р MQTT був стандартизований і прийнятий в консорціум OASIS. У 2014 р OASIS опублікував його публічно як версію MQTT 3.1.1. MQTT також є стандартом ISO (ISO / IECPRF 20922). MQTT базується на стеці TCP/IP.

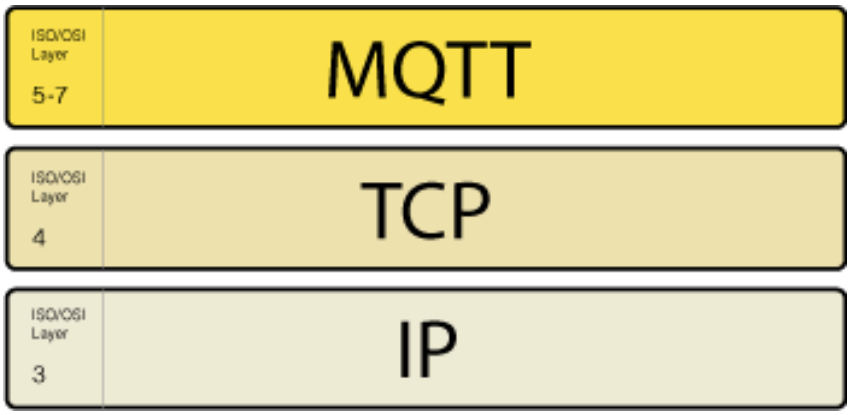

Рисунок 14.11 – Протокол MQTT

У той час як архітектури клієнт-сервер багато років є основою для сервісів центрів обробки, моделі публікація-підписка (publish-subscribe) представляють собою альтернативу, яка корисна для використання IoT. Публікація-підписка, також відома як pub/sub, є способом відокремити клієнта, що передає повідомлення, від іншого клієнта, який отримує повідомлення. На відміну від традиційної моделі клієнт-сервер, клієнти не обізнані про будь- які фізичні ідентифікатори інтерфейсів пристроїв/програм, на зразок IP-адреси або порту.

MQTT - це архітектура pub/sub, однак не є чергою повідомлень. Черги повідомлень по природі своїй зберігають повідомлення, а MQTT - ні. У MQTT, якщо ніхто не підписується (або не слухає) на тему (topic), повідомлення просто ігноруються і втрачаються. Черги повідомлень також підтримують топологію клієнт-сервер, де один споживач з'єднаний з одним виробником.

Клієнт, що передає повідомлення, називається видавцем (publisher); клієнт, який отримує повідомлення - абонентом (subscriber). У центрі знаходиться MQTT- брокер (MQTT broker), який несе відповідальність за обмін повідомленнями між клієнтами і фільтрацію даних. Такі фільтри забезпечують:

- фільтрацію за темами (Topic Filters) - за задумом, клієнти підписуються на теми (topic) і певні гілки тем і не отримують даних більше, ніж хочуть. Кожне опубліковане повідомлення повинно містити тему (topic), і брокер несе відповідальність за повторну передачу цього повідомлення абонентам або ігнорування його;
- фільтрація по вмісту - брокери мають можливість перевіряти і фільтрувати опубліковані дані. Таким чином, будь-які дані, які не зашифровані, можуть керуватися брокером до того, як зберегти їх або передати іншим клієнтам;
- фільтрація за типом - клієнт, що прослуховує потік даних, на які він підписаний, може також застосовувати свої власні фільтри. Вхідні дані можуть аналізуватися, і в залежності від цього потік даних обробляється далі або ігнорується.

Клієнти працюють на краю (Edge), публікують і/або підписуються на теми, керовані брокером MQTT. У прикладі розглянуті дві теми (topic): вологість і температура. Клієнт може підписатися на кілька тем. На рисунку представлені розумні датчики, що володіють достатніми ресурсами для управління власним клієнтом MQTT, а також прикордонні маршрутизатори (Edge Routers), які надають клієнтські послуги MQTT від імені датчиків або пристроїв, які не підтримують MQTT. Одне із особливостей моделі обчислень видавець/абонент

полягає в тому, що як видавець, так і абонент повинні знати тему гілки і формат даних перед початком передачі.

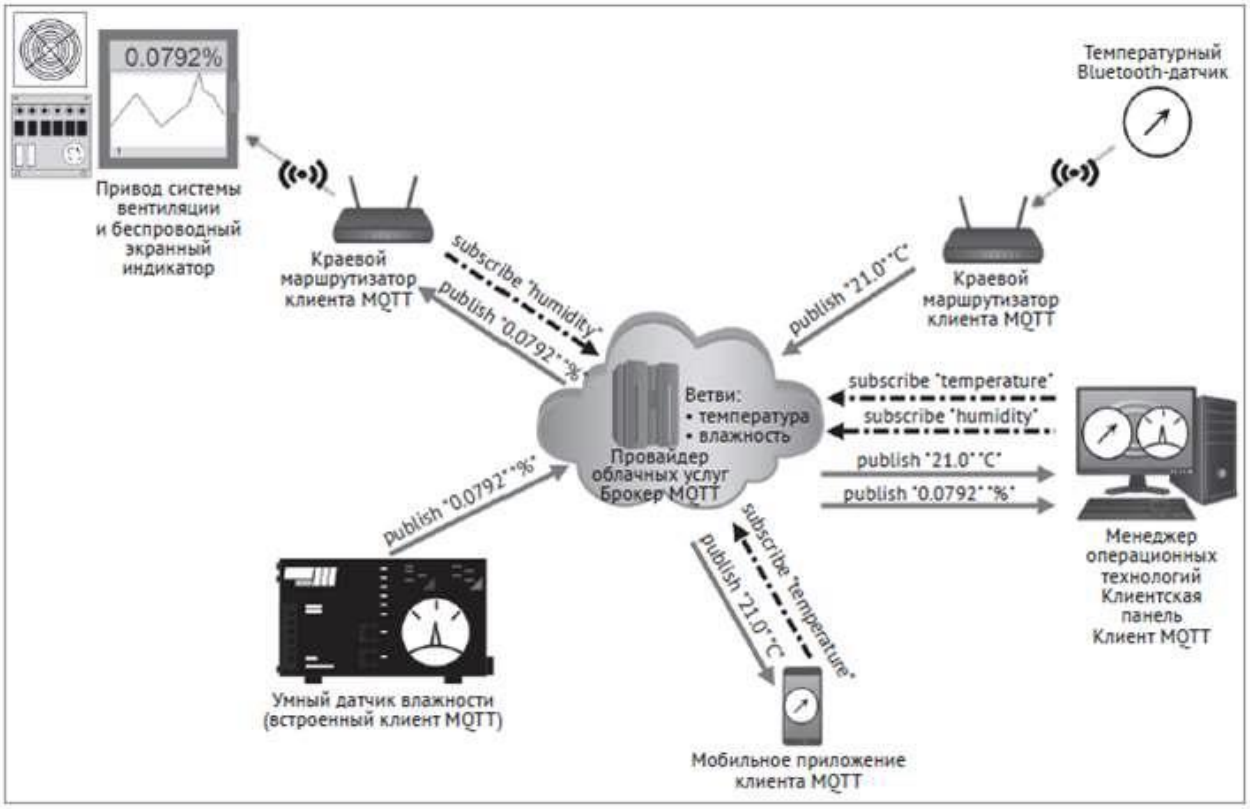

Рисунок 14.12 – Основні принципи взаємодії MQTT

MQTT успішно відокремлює видавців від абонентів. Оскільки брокер є керівним органом між видавцями і абонентами, немає необхідності безпосередньо ідентифікувати їх на основі фізичних даних (таких як IP-адреса). Це дуже корисно при розгортанні IoT, оскільки фізичний ідентифікатор може бути невідомим або загальним. MQTT і інші моделі pub/sub також є часово-незалежними. Це значить, що повідомлення, опубліковане одним клієнтом-видавцем, може бути прочитане абонентом в будь-який час. Абонент може перебувати в місці з дуже низьким енергоспоживанням/обмеженою пропускною здатністю і відповісти на повідомлення через кілька хвилин або годин. Через відсутність фізичних і часових залежностей моделі pub/sub дуже добре підходять для підвищення продуктивності.

MQTT не залежить від формату даних. Корисне навантаження (payload) може містити будь-який тип даних, тому і видавці, і абоненти повинні розуміти і погоджувати формат даних. У корисне навантаження можна надсилати текстові повідомлення, дані зображення, звукові дані, зашифровані дані, двійкові дані, об'єкти JSON або практично будь-яку іншу структуру. Однак текстові і двійкові дані JSON є найбільш поширеними типами даних корисного навантаження.

Максимально допустимий розмір пакета в MQTT становить 256 Мб, що дозволяє отримати надзвичайно велике корисне навантаження. Зверніть увагу, однак, що це також залежить від хмари і брокера. Наприклад, IBM Watson дозволяє обробляти дані розміром до

128 Кб, а Google підтримує 256 Кб. З іншого боку, опубліковане повідомлення може включати корисне навантаження нульової довжини. Поле корисного навантаження не є обов'язковим. Доцільно звірити відповідність розмірів корисного навантаження з вашим хмарним провайдером. Недотримання цієї вимоги призведе до помилок і відмови у доступі до хмарного брокера.

Деталі архітектури MQTT

MQTT може зберігати повідомлення в брокері необмежено довго. Цей режим роботи керується прапорцем. Збережене на брокері повідомлення відправляється будь-якому клієнту, який підписується на цю тематичну гілку MQTT. Повідомлення негайно відправляється цьому новому клієнту. Це дозволяє йому отримати статус або сигнал з теми, на яку він недавно підписався, без очікування. Як правило, клієнт, що підписується на тему, може очікувати годину або навіть дні, перш ніж інший клієнт-видавець опублікує нові дані.

MQTT означує додатковий об'єкт під назвою Остання воля і заповіт (LWT). LWT - це повідомлення, яке вказує клієнт під час етапу підключення. LWT містить тему «Остання воля», QoS і фактичне повідомлення. Якщо клієнт неправильно відключається від брокерського з'єднання (наприклад, тайм-аут keep-alive, помилка введення-виведення або клієнт закриває сеанс без

відключення), тоді брокер зобов'язаний транслювати повідомлення LWT всім іншим підписаним на цю тему клієнтам.

Незважаючи на те, що MQTT заснований на TCP, з'єднання все ще можуть обриватися, особливо в разі безпроводових датчиків. Пристрій може втратити живленя, зв'язок, або може бути просто польова поломка, і сеанс перейде в напіввідкритий стан (тобто з одного боку вважається що з’єднання є, а з іншого воно відсутнє). Тоді сервер буде вважати, що з'єднання як і раніше є надійним і очікувати дані. Щоб вийти з цього напіввідкритого стану, MQTT використовує систему keep-alive (утримування).

Використовуючи цю систему, як брокер MQTT, так і клієнт мають гарантію того, що з'єднання залишається працездатним, навіть якщо протягом деякого часу не було передачі. Після отримання чергового будь-якого пакету, таймери keep-alive скидаються на клієнті і сервері і починають відлік. Якщо протягом часу keep-alive клієнти не мають даних для відправки, вони повинні сформувати відправити пакет PINGREQ брокеру, який, в свою чергу, підтверджує повідомлення за допомогою PINGRESP. Якщо протягом півтора часу keep-alive пакет не буде отримано, брокер закриє з’єднання і відправить LWT-пакет всім клієнтам. Максимальний час keep-alive – 18 годин 12 хвилин 15 секунд.

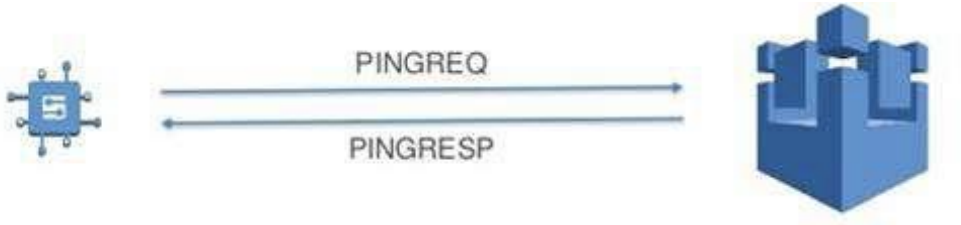

Рисунок 14.13 – Використання системи keep-alive

MQTT дозволяє також підтримувати постійні з’єднання. Постійні з’єднання зберігає на стороні брокера наступне:

– всі підписки клієнта
– всі повідомлення QoS, які не були підтверджені клієнтом

– всі нові повідомлення QoS, пропущені клієнтом

Параметр client_id посилається на цю інформацію для унікальної ідентифікації клієнтів. Клієнт може запитувати постійне з'єднання, проте брокер може відхилити запит і примусово перезапустити новий сеанс. При з'єднанні брокером використовується прапорець cleanSession для дозволу або заборони постійних з'єднань. Клієнт може визначити, чи збереглося постійне з'єднання за допомогою повідомлення CONNACK.

Постійні сеанси повинні використовуватися для клієнтів, які повинні отримувати всі повідомлення, навіть коли немає зв'язку. Вони не повинні використовуватися в ситуаціях, коли клієнт тільки публікує (записує) дані в теми.

Рівні якості обслуговування MQTT

У MQTT є три рівня якості обслуговування:

– QoS-0 (незавірена передача) - це мінімальний рівень QoS. Це аналогічно моделі «спалити і забути», докладно описаної в деяких бездротових протоколах. Це найефективніший процес доставки без підтвердження одержувачем повідомлення і без повторної передачі повідомлення відправником;
– QoS-1 (гарантована передача) - цей режим гарантує доставку повідомлення хоча б один раз одержувачу. Повідомлення може бути доставлено кілька разів, і одержувач відправить назад підтвердження з відповіддю PUBACK;
– QoS-2 (гарантований сервіс для додатків) - це найвищий рівень QoS, який переконується в доставці і інформує відправника і одержувача, що повідомлення було передано правильно. Цей режим генерує більше трафіку через багатокрокове рукостискання між відправником і отримувачем.

Якщо одержувач отримує повідомлення з рівнем обслуговування QoS-2, він відповість відправнику повідомленням PUBREC. Це підтверджує

повідомлення, і відправник відповість повідомленням PUBREL. PUBREL дозволяє одержувачеві безпечно відкинути будь-які повторні передачі цього повідомлення. Потім PUBREL підтверджується одержувачем за допомогою PUBCOMP. Поки повідомлення PUBCOMP передано не буде, приймач буде кешувати вихідне повідомлення для забезпечення безпеки.

QoS в MQTT визначається і контролюється відправником, і у кожного відправника може бути своя політика.

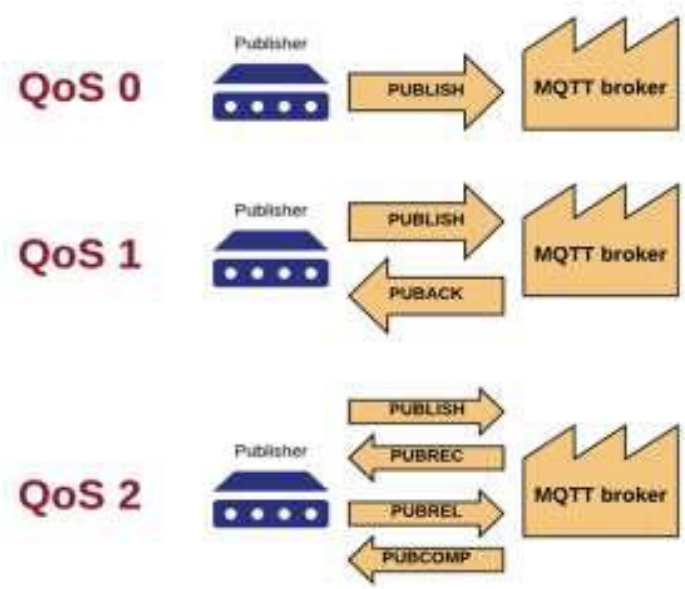

Рисунок 14.14 – Рівні якості обслуговування MQTT

Типові випадки використання:

QoS-0 слід використовувати, коли повідомлення не потрібно зберігати в черзі. QoS-0 найкраще підходить для провідного підключення або коли система сильно обмежена в пропускної здатності;

Qos-1 слід використовувати за замовчуванням. QoS1 набагато швидше, ніж QoS2, і значно знижує вартість передачі;

QoS-2 - для критично важливих застосунків. Крім того, для випадків, коли повторна передача дубльованого повідомлення може привести до помилок.

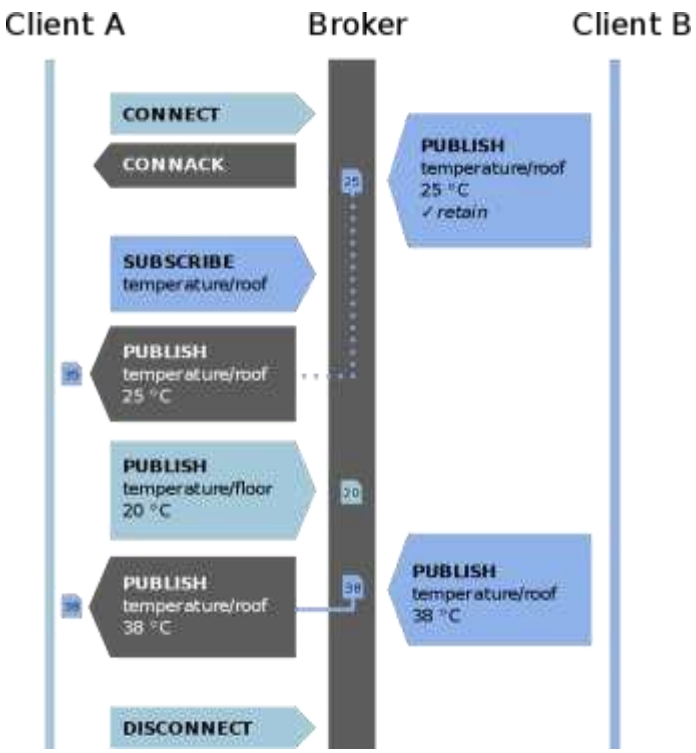

Рисунок 14.15 – Процедури та повідомлення, що використовуються в MQTT

Також існує версія протоколу MQTT-SN (MQTT для мереж датчиків), раніше відома як MQTT-S, яка призначена для вбудованих бездротових пристроїв без підтримки TCP/IP мереж.

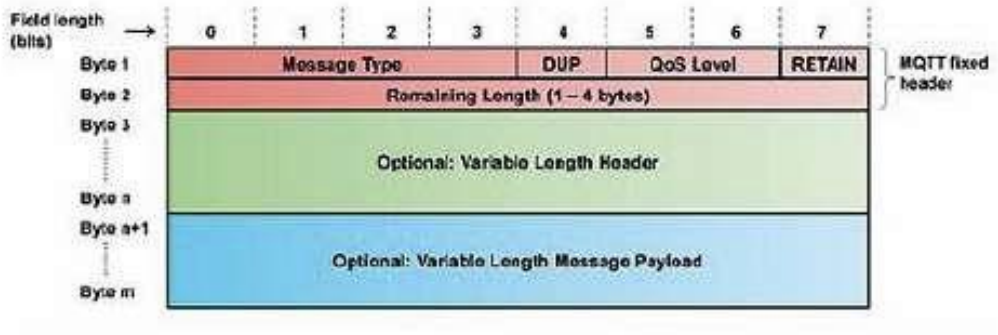

Рисунок 14.16 – Загальний формат повідомлення протоколу MQTT

На рисунку 14.16 представлений загальний формат повідомлень проколу MQTT. Повідомлення складається з двох заголовків:

– MQTT Fixed Header – заголовок фіксованої довжини;

– Variable Length Header – заголовок змінної довжини (в залежності від типу повідомлення);
– Variable Length Message Payload – поля корисного навантаження змінної довжини.

До заголовка фіксованої довжини входять такі поля:

– Message Type – тип повідомлення,
– DUP – прапор дублювання повідомлення,
– QoS Level – рівень якості обслуговування,
– Retain – спеціальний прапор збереження останнього прийнятого брокером повідомлення,
– Remaining Length – залишкова довжина,

Спрощений процес роботи протоколу MQTT:

Видавець передає повідомлення з певними даними (наприклад, інформація з датчиків вологості) на брокера, вказуючи при цьому тему (Topic), до якої ці дані відносяться (наприклад, "вологість") (див. рис. 14.17).

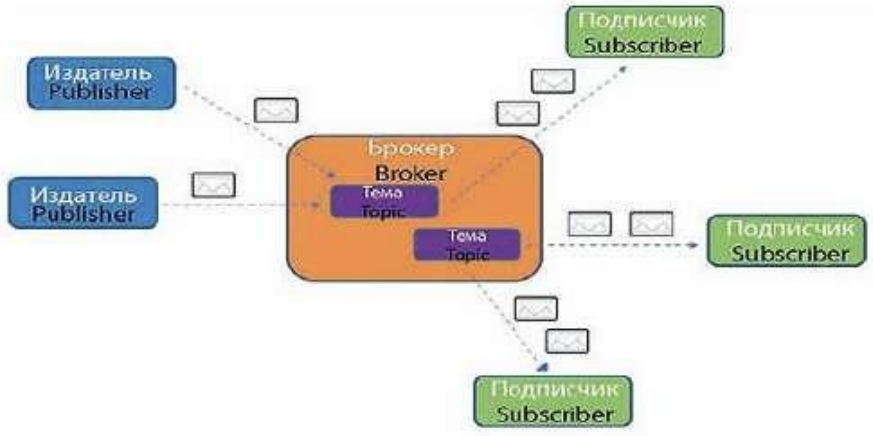

Рисунок 14.17 – Спрощений процес роботи протоколу MQTT

Брокер аналізує, які із передплатників мають підписку на певні теми, в даному випадку – на тему "вологість". Передплатникам, які підписані на тему "вологість", брокером буде відправлено повідомлення з інформацією від датчиків вологості. Таким чином, безліч передплатників можуть бути підписані

на різноманітні теми і в залежності від цих підписок отримувати необхідну їм інформацію, не спілкуючись з видавцем безпосередньо.

На рисунку 14.17. зображено схему передачі інформації за принципом "видавець- передплатник".

Таблиця 14.1 – Порівняльна характеристика протоколів передачі повідомлень

Протокол	Транспорт	Призначення	Особливості
DDS	UDP	Для мереж, що потребують розподіленого навантаження	Реалізує прямий шинний зв'язок між пристроями на базі реляційної моделі даних
XMPP	TCP	Для адресації в невеликій персональній мережі	Для ідентифікації користувачів використовуються JID, по формату схожі на адреса електронної пошти (username@gmail.com)
COAP	UDP	Для мереж з обмеженими ресурсами, низьким Електроспоживанням	Враховує різні питання середовища реалізації в обмежених мережах
MQTT	TCP	Для завантажених мереж з великою кількістю пристроїв та Брокером	Використання механізму черг повідомлень
STOMP	TCP	Для мереж, в яких є можливість використання декількох комбінацій різних протоколів, що потребують простий протокол передачі повідомлень через брокера	Взаємодія з більшістю мов, платформ та Брокерами
SOAP	TCP	Для розподіленої обчислювальної мережі	Підтримує два механізми доступу: SOAP RPC та SOAP Message