Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційних радіотехнологій і технічного захисту інформації
(повна назва)
Кафедра Радіотехнологій інформаційно-комунікаційних систем
(повна назва)

# КВАЛІФІКАЦІЙНА РОБОТА
# Пояснювальна записка

рівень вищої освіти **перший (бакалаврський)**

## АНАЛІЗ ТА ОПТИМІЗАЦІЯ ПРОТОКОЛІВ ПЕРЕДАВАННЯ ДАНИХ ДЛЯ ІНТЕРНЕТУ РЕЧЕЙ В УМОВАХ ОБМЕЖЕНОЇ ПРОПУСКНОЇ ЗДАТНОСТІ ТА ЕНЕРГОСПОЖИВАННЯ
(тема)

Виконав:
студент **IV** курсу, групи **ІТІР-20-1**
**Денькович О. М.**
(прізвище, ініціали)
Спеціальність **126 Інформаційні системи та технології**
(код і повна назва спеціальності)
Тип програми **освітньо-професійна**

Освітня програма **Інформаційні технології інтернету речей**
( повна назва освітньої програми)
Керівник **Доктор філософії Мерзлікін А. О.**
(посада, прізвище, ініціали)

Допускається до захисту
В.о зав. кафедри РТІКС _____ **Зарудний О.А.**

(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційних радіотехнологій і технічного захисту інформації
Кафедра Радіотехнологій інформаційно-комунікаційних систем
Рівень вищої освіти перший (бакалаврський)
Спеціальність 126 Інформаційні системи та технології
(код і повна назва)
Тип програми Освітньо-професійна

Освітня програма Інформаційні технології інтернету речей
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____»_____ 2024 р.

# ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові **ДЕНЬКОВИЧУ ОЛЕКСАНДРУ МИКОЛАЙОВИЧУ**

(прізвище, ім'я, по батькові)

1.Тема роботи **АНАЛІЗ ТА ОПТИМІЗАЦІЯ ПРОТОКОЛІВ ПЕРЕДАВАННЯ ДАНИХ ДЛЯ ІНТЕРНЕТУ РЕЧЕЙ В УМОВАХ ОБМЕЖЕНОЇ ПРОПУСКНОЇ ЗДАТНОСТІ ТА ЕНЕРГОСПОЖИВАННЯ**

затверджена наказом по університету від **27 травня 2024 р**. **№ 500 Ст**

2. Термін подання студентом роботи до екзаменаційної комісії **10 червня 2024 р.**

3. Вихідні дані до роботи _____
Літературні джерела та електронні ресурси за темою кваліфікаційної роботи

4. Перелік питань, що потрібно опрацювати в роботі _____
Вступ. 1 ТЕОРЕТИЧНІ ЗАСАДИ ДОСЛІДЖЕННЯ. 2 АНАЛІЗ ПРОТОКОЛІВ ПЕРЕДАЧІ ДАНИХ ДЛЯ ІОТ. 3 ОПТИМІЗАЦІЯ ПРОТОКОЛІВ ПЕРЕДАЧІ ДАНИХ ДЛЯ ІОТ. Висновки. Перелік джерел посилання. Додатки.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) (п.5 включається до завдання за рішенням випускової кафедри)_____
Слайди комп'ютерної презентації_____
_____
_____
_____

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1 )

| Найменування розділу | Консультант (посада, прізвище, ім'я, по батькові) | Позначка консультанта про виконання розділу | |
|---|---|---|---|
| | | підпис | дата |
| Основна частина | PhD Мерзлікін А.О. | | |
| | | | |

## КАЛЕНДАРНИЙ ПЛАН

| № | Назва етапів роботи | Терміни виконання етапів роботи | Примітка |
|---|---|---|---|
| 1 | Вступ | 06.05-13.05.2024 | виконано |
| 2 | Огляд літератури | 14.05-20.05.2024 | виконано |
| 3 | Огляд протоколів передавання даних для пристроїв інтернету речей | 21.05-27.05.2024 | виконано |
| 4 | Аналіз протоколів передавання даних для пристроїв інтернету речей | 28.05-02.06.2024 | виконано |
| 5 | Вибір методів оптимізації | 03.06-7.06.2024 | виконано |
| 6 | Висновки | 7.06.2024 | виконано |
| 7 | Оформлення пояснювальної записки | 8.06.2024 | виконано |
| 8 | Оформлення ілюстрацій | 9.06.2024 | виконано |
| 9 | Представлення роботи на кафедру | 10.06.2024 | виконано |

Дата видачі завдання **6 травня 2024 р.**

Студент _____ О.М. Денькович
(підпис)

Керівник роботи _____ PhD А.О. Мерзлікін
(підпис)

# РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи бакалавра містить: 60 сторінок, 20 рисунків, 1 таблицю, 20 джерел, 2 додатки.

IoT. ПРОТОКОЛИ. ОПТИМІЗАЦІЯ. ДАНІ. АНАЛІЗ. СТИСНЕННЯ. АДАПТИВНІСТЬ. ШВИДКІСТЬ. ЕФЕКТИВНІСТЬ. ПАРАЛЕЛЬНІСТЬ. ПЕРЕДАЧА. БАЛАНСУВАННЯ. КЕШУВАННЯ.

Об’єктом розробки є оптимізація протоколів передачі даних для Інтернету речей (IoT) в умовах обмеженої пропускної здатності та енергоспоживання.

Метод дослідження – описово-аналітичний.

Метою даної дипломної роботи є розробка та впровадження методів оптимізації протоколів передачі даних для IoT, спрямованих на зниження навантаження на мережу та енергоспоживання.

В даній роботі проведено огляд і аналіз існуючих протоколів передачі даних для IoT, таких як MQTT, CoAP та LoRaWAN, їх порівняння та визначення переваг і недоліків. Розглянуто основні проблеми та виклики, пов'язані з обмеженою пропускною здатністю та енергоспоживанням в системах IoT. Запропоновано методи оптимізації, включаючи стиснення даних, адаптивну швидкість передачі даних, паралельну передачу, балансування навантаження та кешування. Проведено експериментальне дослідження для оцінки ефективності запропонованих методів і розроблено рекомендації щодо їх впровадження у практичних системах IoT.

# ABSTRACT

The explanatory note to the bachelor's qualification work contains: 60 pages, 20 figures, 1 table, 20 sources, 2 applications.

IoT. PROTOCOLS. OPTIMIZATION. DATA. ANALYSIS. COMPRESSION. ADAPTABILITY. SPEED. EFFICIENCY. PARALLELNESS. TRANSFER. BALANCING. CACHING.

The object of development is the optimization of data transmission protocols for the Internet of Things (IoT) in conditions of limited bandwidth and energy consumption.

The research method is descriptive and analytical.

The purpose of this thesis is to develop and implement methods for optimizing data transmission protocols for IoT, aimed at reducing network load and energy consumption.

This paper reviews and analyzes existing data transmission protocols for IoT, such as MQTT, CoAP, and LoRaWAN, compares them, and identifies their advantages and disadvantages. The main problems and challenges associated with limited bandwidth and energy consumption in IoT systems are considered. Optimization techniques are proposed, including data compression, adaptive data rate, parallel transmission, load balancing, and caching. An experimental study was conducted to evaluate the effectiveness of the proposed methods and recommendations were developed for their implementation in practical IoT systems.

ЗМІСТ

# ПЕРЕЛІК УМОВНИХ ПОЗНАЧОК, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

IoT - Інтернет речей (Internet of Things)

MQTT - Протокол телеметрії чергування повідомлень (Message Queuing Telemetry Transport)

CoAP - Протокол прикладних обмежень (Constrained Application Protocol)

JSON - Легкий формат обміну даними (JavaScript Object Notation)

CBOR - Компактний двійковий об'єктний запис (Concise Binary Object Representation)

LPWAN - Мережа широкого охоплення з низьким енергоспоживанням (Low-Power Wide-Area Network)

LoRa - Далекобійна радіотехнологія (Long Range)

Sigfox - Назва компанії та технології для низькошвидкісного інтернету речей

Zigbee - Специфікація для високорівневих комунікаційних протоколів бездротових особистих мереж

UDP - Протокол користувацьких дейтаграм (User Datagram Protocol)

TCP - Протокол керування передаванням (Transmission Control Protocol)

HTTP - Протокол передачі гіпертексту (HyperText Transfer Protocol)

HTTPS - Захищений протокол передачі гіпертексту (HyperText Transfer Protocol Secure)

SSL - Протокол захищених сокетів (Secure Sockets Layer)

TLS - Протокол транспортного рівня захисту (Transport Layer Security)

RFID - Радіочастотна ідентифікація (Radio-Frequency Identification)

# ВСТУП

Інтернет речей (IoT) – це одна з найшвидше зростаючих галузей сучасної технології. Згідно з прогнозами, кількість підключених пристроїв буде продовжувати зростати в геометричній прогресії, що створює підвищене навантаження на існуючу інфраструктуру. Оптимізація протоколів передавання даних стає критичною для забезпечення ефективної взаємодії між пристроями та для уникнення перевантаження мереж [1].

Багато пристроїв IoT функціонують в умовах обмежених ресурсів, таких як низька пропускна здатність мережі та обмежене енергоспоживання. Це особливо актуально для пристроїв, що працюють на батарейках або мають обмежений доступ до електропостачання. У таких випадках важливо оптимізувати використання ресурсів, щоб забезпечити тривалий час роботи пристроїв та ефективне передавання даних.

IoT застосовується у багатьох галузях, включаючи сільське господарство, охорону здоров'я, промисловість, розумні міста тощо. Кожна з цих галузей має унікальні вимоги до пристроїв та мережі, що вимагає спеціальних підходів до оптимізації протоколів. Необхідно забезпечити сумісність, масштабованість та адаптивність протоколів до різних умов.

З ростом кількості пристроїв і обсягу даних, що передаються, потреба в масштабованості стає очевидною. Протоколи передавання даних повинні бути спроможними витримувати збільшення навантаження, зберігаючи при цьому надійність та продуктивність. Це вимагає інноваційних рішень, що забезпечують баланс між ефективністю та стійкістю [2].

Збільшення кількості пристроїв IoT і підвищення обсягів переданих даних також призводять до зростання ризику кібератак та витоку інформації. Протоколи передавання даних повинні забезпечувати належний рівень безпеки та конфіденційності, при цьому не перевантажуючи пристрої надмірними обчислювальними задачами.

Для багатьох застосувань IoT, особливо в промисловості та охороні здоров'я, важливо забезпечити стійкість і надійність систем. Це вимагає

протоколів, що можуть працювати в умовах несприятливих зовнішніх факторів, таких як втрати з'єднання, перешкоди у сигналі та інші види технічних проблем.

Мета даного дослідження полягає в розробці та оптимізації протоколів передавання даних для Інтернету речей (IoT), які будуть ефективно функціонувати в умовах обмеженої пропускної здатності та енергоспоживання. Основна ціль — забезпечити стабільну, надійну та безпечну комунікацію між пристроями IoT, зберігаючи ефективність та довговічність їхньої роботи. Для досягнення поставленої мети визначено такі завдання:

– провести комплексний аналіз існуючих протоколів передавання даних в контексті Інтернету речей, оцінюючи їх ефективність, пропускну здатність, енергоспоживання та безпеку. Визначити переваги та недоліки кожного протоколу, а також області їх застосування;

– встановити ключові вимоги до протоколів передавання даних для IoT в умовах обмеженої пропускної здатності та енергоспоживання. Ці вимоги повинні враховувати специфіку різних галузей, де використовуються пристрої IoT;

– на основі проведеного аналізу розробити нові або оптимізувати існуючі протоколи передавання даних, враховуючи вимоги до енергоспоживання, пропускної здатності та безпеки. Застосувати різні методи, такі як компресія даних, ефективне планування передавання, сплячі режими тощо;

– реалізувати розроблені або оптимізовані протоколи та провести їх тестування в реальних умовах. Перевірити їх ефективність, надійність та безпеку в різних сценаріях застосування;

– провести аналіз результатів тестування, порівнявши оптимізовані протоколи з існуючими стандартами. Визначити ключові рекомендації щодо впровадження протоколів передавання даних для IoT в умовах обмеженої пропускної здатності та енергоспоживання;

– вивчити вплив оптимізованих протоколів на практичне

застосування IoT у різних галузях. Окреслити напрямки для майбутніх досліджень, щоб продовжити вдосконалення та адаптацію протоколів до нових викликів і технологій.

Наукова новизна цього дослідження полягає у розробці та оптимізації протоколів передавання даних для Інтернету речей (IoT), з урахуванням обмеженої пропускної здатності та енергоспоживання. Новаторство полягає в розробці інноваційних підходів, що включають нові або суттєво оптимізовані протоколи, які дозволяють пристроям IoT працювати більш ефективно в умовах обмежених ресурсів. Ці оптимізації можуть включати компресію даних, використання сплячих режимів та адаптивну обробку інформації.

Особливу увагу приділено забезпеченню безпеки та надійності, що має велике значення для пристроїв IoT. Запропоновано легкі методи шифрування, аутентифікації та захисту від помилок, які дозволяють підтримувати безпеку даних без надмірного збільшення енергоспоживання чи зниження пропускної здатності. Дослідження також розглядає можливості застосування цих протоколів у різних галузях, таких як сільське господарство, охорона здоров'я, промисловість, розумні міста тощо, що забезпечує універсальність та гнучкість розроблених рішень.

Практична значимість цього дослідження проявляється в підвищенні ефективності роботи систем IoT, оскільки оптимізовані протоколи дозволяють збільшити тривалість роботи пристроїв на батарейках, зменшити навантаження на мережу та підвищити швидкість передавання даних. Важливий аспект практичної значимості — підвищення стійкості та надійності систем IoT, що особливо актуально для критичних галузей, як-от охорона здоров'я чи промисловість.

З точки зору економіки, оптимізовані протоколи можуть знизити витрати на експлуатацію систем IoT, оскільки зменшують необхідність частих замін батарейок та загалом знижують витрати на підтримку мережі. У цьому дослідженні також розглядаються способи підвищення рівня безпеки, що важливо для запобігання кібератакам та захисту конфіденційної інформації.

Нарешті, дослідження сприяє розширенню сфери застосування IoT, дозволяючи інтегрувати більше пристроїв та систем у різних галузях, відкриваючи нові можливості для розвитку технологій.

Таким чином, наукова новизна та практична значимість цього дослідження роблять вагомий внесок у розвиток Інтернету речей, пропонуючи інноваційні рішення для ефективного передавання даних та їх застосування в умовах обмежених ресурсів.

# 1 ТЕОРЕТИЧНІ ЗАСАДИ ДОСЛІДЖЕННЯ

## 1.1 Інтернет речей (IoT): поняття, структура, сучасний стан

Інтернет речей (IoT) – це концепція, що описує екосистему фізичних об'єктів, які можуть підключатися до Інтернету, взаємодіяти між собою та з іншими системами, а також збирати, обмінюватися і аналізувати дані. Сюди належать пристрої різних типів і розмірів, від невеликих носимих пристроїв до великих промислових систем. Вони оснащені вбудованими сенсорами, датчиками, мікроконтролерами та можуть передавати дані через різноманітні протоколи зв'язку, такі як Wi-Fi, Bluetooth, Zigbee, LoRaWAN або 5G[3].

На даний момент у світі налічується понад 15 мільярдів підключених пристроїв IoT по всьому світу [4]. Очікується, що число активних IoT пристроїв подвоїться до 2030 року (рисунок 1.1).

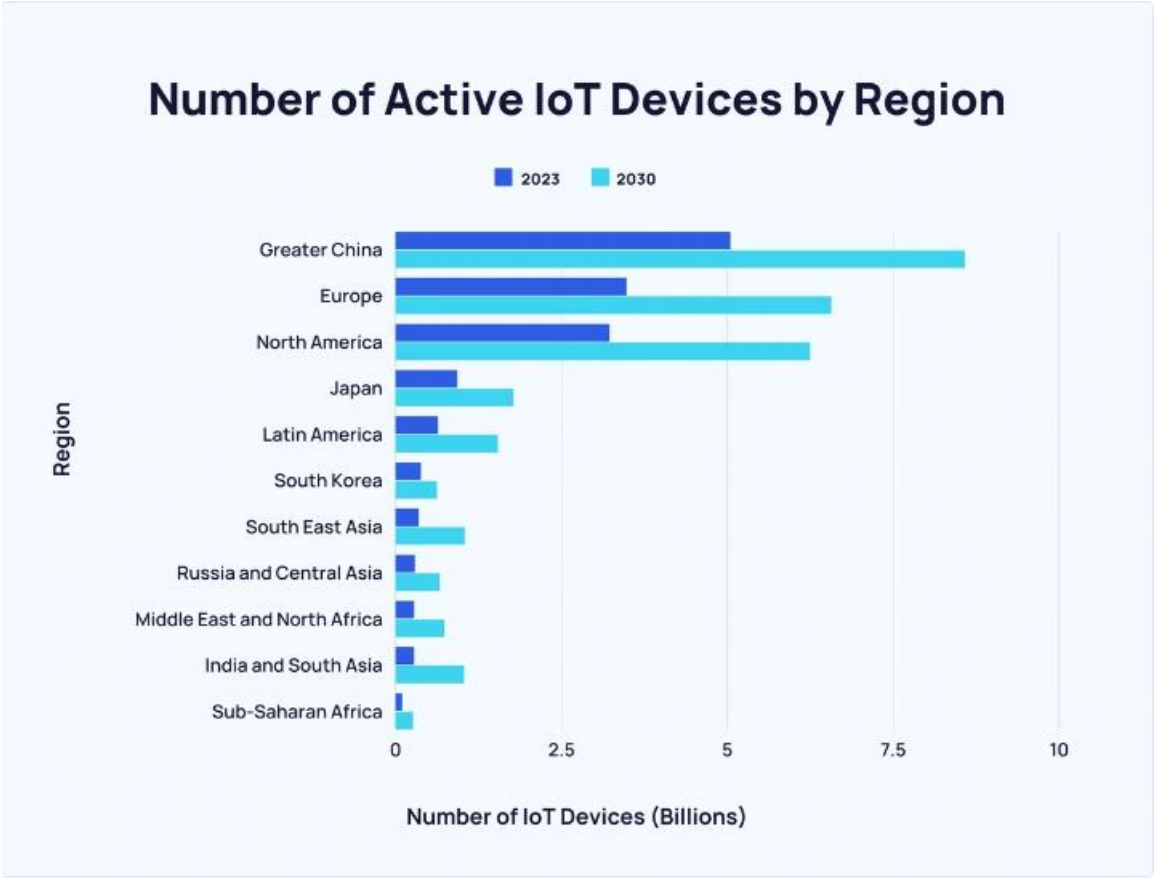



Рисунок 1.1 – Число активних пристроїв IoT за регіонами та прогноз їх збільшення до 2030 року

Віддалений моніторинг активів є найпопулярнішим напрямком використання пристроїв IoT (рисунок 1.2). Цей напрямок використовує спеціальні протоколи які можуть переправляти дані на великі відстані.

| Use Case | IoT Adoption Rate |
|---|---|
| Remote asset monitoring (read-only) | 34% |
| Process automation | 33% |
| Remote asset monitoring and control (read/wire) | 32% |
| Vehicle fleet management (track/trace) | 31% |
| Location tracking (eg. GPS) | 31% |
| Asset/plant performance optimization | 31% |
| Quality control and management | 30% |
| Goods condition monitoring in transit | 29% |
| Predictive maintenance | 29% |
| On-site track and trace | 29% |

Рисунок 1.2 – Віддалений моніторинг (Remote asset monitoring) є найпопулярнішим напрямком застосування IoT

Ефективне функціонування IoT, окрім самих пристроїв, потребує відповідної інфраструктури, що включає хмарні обчислення для зберігання та обробки даних, а також інструменти аналітики для витягування корисної інформації з масиву даних. Ці компоненти забезпечують гнучкість і масштабованість систем IoT, дозволяючи розширювати їх відповідно до потреб і підтримувати взаємодію між великою кількістю пристроїв.

Однією з ключових характеристик IoT є здатність пристроїв бути взаємопов'язаними, що створює великі можливості для інтерактивності та автоматизації. Системи IoT можуть бути масштабованими, додаючи нові

пристрої та розширюючи функціональність, що робить їх привабливими для різних галузей – від сільського господарства до розумних міст.

Багато пристроїв IoT працюють автономно, приймаючи рішення на основі зібраних даних, а також мають низьке енергоспоживання, що дозволяє їм функціонувати у віддалених або мобільних умовах. Враховуючи те, що ці пристрої взаємодіють із мережами та передають дані, важливо забезпечити високий рівень безпеки та приватності, використовуючи шифрування, аутентифікацію та інші заходи для захисту від кібератак і несанкціонованого доступу.

Отже, Інтернет речей – це складна, але надзвичайно перспективна екосистема, яка об'єднує фізичні пристрої та мережі для створення нових можливостей і автоматизації процесів у різних галузях, забезпечуючи при цьому гнучкість, масштабованість і безпеку.

## 1.2 Протоколи передачі даних для IoT: огляд та класифікація

Протоколи передачі даних для Інтернету речей (IoT) є ключовим елементом, який забезпечує зв'язок між різними пристроями та системами в межах екосистеми IoT. Ці протоколи необхідні для організації надійної передачі даних з урахуванням різних обмежень, таких як пропускна здатність, енергоспоживання та відстань зв'язку. Класифікація протоколів IoT відбувається за різними параметрами, включаючи рівень OSI, топологію мережі, відстань передачі та енергоспоживання.

Протокол – це обумовлені наперед правила передачі даних між двома пристроями[5]. До основних параметрів, які описує протокол, відносяться:

– тип перевірки помилок, що використовується;
– метод компресії (стискання) інформації (якщо такий є);
– спосіб визначення передаючим пристроєм завершення передачі.

Різні протоколи відрізняються своїми характеристиками: одні – більшою

надійністю, другі – швидкістю передачі даних, треті – простотою.

Що стосується рівня OSI, протоколи IoT охоплюють широкий спектр, від фізичного рівня до рівня застосунків (рисунок 1.3). На фізичному рівні визначаються типи передачі сигналу, частотні діапазони та інші фізичні характеристики, тоді як на канальному рівні регулюються питання контролю помилок та керування доступом до мережі. Мережевий рівень відповідає за маршрутизацію даних, а транспортний рівень за забезпечення надійності передачі. Нарешті, на рівні застосунків протоколи визначають способи взаємодії між програмами.

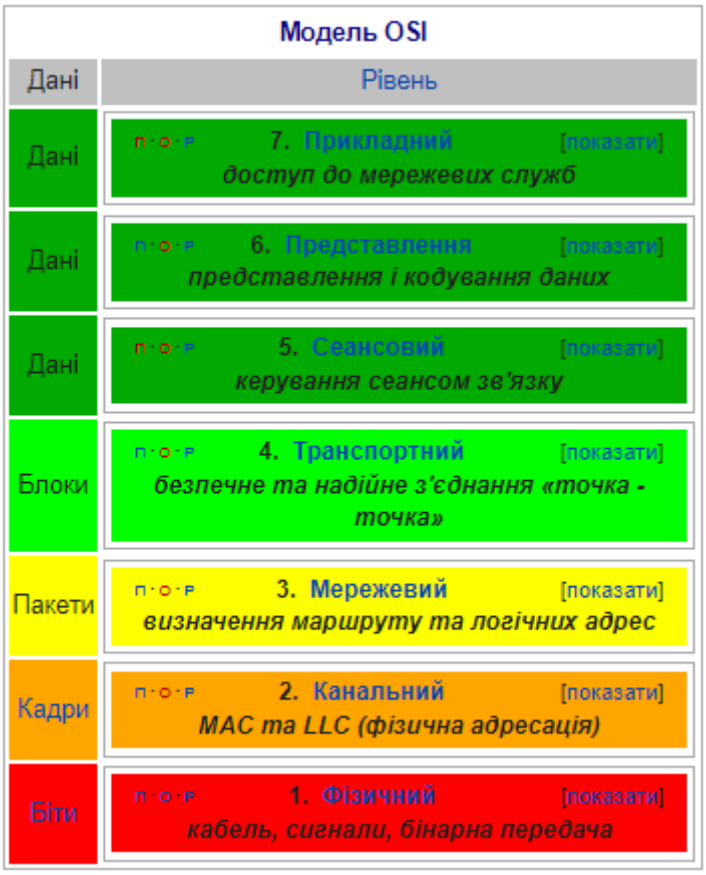



Рисунок 1.3 – Модель OSI

Топологія мережі також є важливим критерієм класифікації протоколів IoT. Деякі протоколи використовують зіркоподібну або лінійну топологію, інші—сітчасту, яка забезпечує додаткову стійкість, оскільки дозволяє пристроям передавати дані через інші вузли. Відстань передачі теж варіюється: деякі протоколи, як-от Bluetooth, оптимізовані для коротких відстаней, тоді як

інші, як-от LoRaWAN, призначені для передачі даних на великі відстані. Енергоспоживання також є важливим фактором, оскільки деякі протоколи орієнтовані на низьке енергоспоживання, а інші можуть забезпечувати більшу пропускну здатність за рахунок вищого енергоспоживання. Розглянемо більш детально класифікацію протоколів IoT, які охоплюють різні рівні моделі OSI, від фізичного рівня до рівня застосунків:

Фізичний рівень: Цей рівень стосується передачі бітів через фізичні носії. Протоколи на цьому рівні визначають типи передачі сигналу, частотні діапазони та інші фізичні характеристики. Приклади включають Bluetooth, Zigbee, Z-Wave, LoRaWAN та NB-IoT.

Канальний рівень: Тут визначаються механізми для передачі даних між пристроями в одній мережі. Протоколи на цьому рівні також відповідають за контроль помилок і керування доступом до мережі. Zigbee, Z-Wave, Bluetooth Low Energy (BLE), Wi-Fi та інші працюють на цьому рівні.

Мережевий рівень: Цей рівень забезпечує маршрутизацію даних через мережу. Протоколи, такі як 6LoWPAN (IPv6 over Low-Power Wireless Personal Area Networks), використовуються для забезпечення маршрутизації в мережах з низьким енергоспоживанням.

Транспортний рівень: Цей рівень відповідає за надійність передачі даних. Протоколи на цьому рівні, такі як TCP і UDP, визначають способи передачі даних з різним ступенем надійності.

Рівень застосунків: На цьому рівні протоколи визначають способи взаємодії між програмами. Основні протоколи для IoT на цьому рівні включають MQTT, CoAP, AMQP, HTTP та інші.

Протоколи можуть використовувати різні топології, включаючи зіркоподібну, лінійну, сітчасту (mesh) та інші. Сітчасті топології (mesh) дозволяють пристроям з'єднуватися між собою і передавати дані через інші вузли, підвищуючи стійкість мережі.

Протоколи розрізняються за дальністю передачі:

Ближній діапазон: Протоколи, такі як Bluetooth, Zigbee та Z-Wave,

призначені для коротких відстаней, зазвичай до 100 метрів.

Середній діапазон: Протоколи, такі як Wi-Fi, мають середню дальність передачі, до декількох сотень метрів.

Далекий діапазон: Протоколи для зв'язку на великі відстані, такі як LoRaWAN та NB-IoT, можуть забезпечувати зв'язок на кілька кілометрів або більше.

Протоколи також розрізняються за рівнем енергоспоживання. Протоколи з низьким енергоспоживанням, такі як BLE та LoRaWAN, розроблені для пристроїв з обмеженими ресурсами енергії. Протоколи, як Wi-Fi, мають вищий рівень енергоспоживання, але пропонують більшу пропускну здатність.

Ось короткий огляд основних протоколів передачі даних для IoT:

– MQTT (Message Queuing Telemetry Transport): Легкий протокол на основі TCP, який використовує модель "публікація-підписка". Популярний у застосунках, де потрібна низька затримка та висока надійність;

– CoAP (Constrained Application Protocol): Протокол на основі UDP, який призначений для пристроїв з обмеженими ресурсами. Він використовує REST-підхід і підходить для сценаріїв, де потрібна легкість і простота;

– AMQP (Advanced Message Queuing Protocol): Цей протокол забезпечує високий рівень надійності і зазвичай використовується в промислових та підприємницьких застосунках;

– HTTP/HTTPS: Стандартні протоколи веб-застосунків, що забезпечують передачу даних через Інтернет. HTTPS використовується для безпечної передачі;

– Bluetooth: Протокол для ближнього зв'язку, який підтримує передачу даних між пристроями на коротких відстанях;

– Zigbee: Протокол, призначений для сітчастих мереж, який використовується для зв'язку в розумних будинках та промислових застосунках.

– LoRaWAN: Протокол для передачі даних на далекі відстані з низьким енергоспоживанням. Популярний для сільського господарства та

інших галузей, де потрібен зв'язок на великих відстанях;

– Wi-Fi: Стандартний протокол для бездротового зв'язку, що забезпечує високу пропускну здатність, але з вищим енергоспоживанням.

– NB-IoT (Narrowband IoT): Протокол для мереж широкого охоплення, призначений для підключення великої кількості пристроїв з низьким енергоспоживанням.

## 1.3 Обмежена пропускна здатність та енергоспоживання в IoT: проблеми та виклики

Обмежена пропускна здатність та енергоспоживання — два ключові виклики в Інтернеті речей (IoT), які впливають на ефективність, надійність і довговічність пристроїв і мереж. Ці виклики мають комплексний характер, оскільки різні пристрої IoT можуть працювати в різних умовах з різними потребами.

Пропускна здатність в IoT стосується обсягу даних, який можна передати за одиницю часу. Багато пристроїв IoT мають обмежену пропускну здатність, особливо якщо використовують протоколи з низьким енергоспоживанням або працюють у віддалених районах. Це може призвести до затримок у передачі даних, втрати інформації або погіршення якості зв'язку, особливо при підвищеному навантаженні на мережу. Внаслідок цього виникають додаткові накладні витрати, пов'язані з контролем помилок та повторною передачею даних, що знижує загальну ефективність мережі.

Енергоспоживання є ще однією важливою проблемою в IoT, особливо для пристроїв, що працюють від батарейок або інших обмежених джерел енергії. Прагнення знизити енергоспоживання може вплинути на функціональність пристроїв і призвести до збільшення затримок у передачі даних або обмеження можливостей підключення до мережі. Така ситуація може ускладнити використання протоколів з високою пропускною здатністю, оскільки вони потребують більше енергії.

Крім того, обмежені пропускна здатність і енергоспоживання створюють проблеми сумісності та безпеки. Пристрої з низьким енергоспоживанням часто мають обмежені можливості для забезпечення безпеки, що підвищує ризик кібератак. Також виникають складнощі зі сумісністю між різними протоколами та пристроями, що може ускладнити інтеграцію в більш широку екосистему IoT.

Подолання цих викликів вимагає комплексного підходу, який включає використання енергоефективних протоколів, оптимізацію передачі даних, компресію, сплячі режими та адаптивні алгоритми передачі даних. Важливо також забезпечити належний рівень безпеки, застосовуючи легкі методи шифрування та аутентифікації.

Загалом, проблеми, пов'язані з обмеженою пропускною здатністю та енергоспоживанням, вимагають ретельного планування і розробки рішень, які забезпечують ефективну роботу пристроїв IoT за різних умов, зберігаючи водночас належний рівень безпеки та надійності.

# 2 АНАЛІЗ ПРОТОКОЛІВ ПЕРЕДАЧІ ДАНИХ ДЛЯ IOT

## 2.1 MQTT: опис протоколу, характеристики, переваги та недоліки

MQTT - це стандартизований протокол, або набір правил, обміну повідомленнями, який використовується для взаємодії між комп'ютерами. Інтелектуальні датчики, носимі пристрої та інші пристрої Інтернету речей (IoT) зазвичай передають та отримують дані через мережі з обмеженими ресурсами та пропускною здатністю. Ці IoT-пристрої використовують MQTT для передачі даних, тому що він простий у реалізації та може ефективно передавати дані IoT.

Протокол MQTT винайдено в 1999 році для використання в нафтогазовій промисловості. Інженерам знадобився протокол із мінімальною пропускною спроможністю та мінімальною розрядкою акумулятора для відстеження нафтопроводу із супутника. Спочатку протокол називався телеметричним транспортом черги повідомлень під назвою продукту IBM MQ Series, який підтримував початкову фазу [6]. У 2010 році IBM випустила MQTT 3.1 як безкоштовний та відкритий протокол, який може реалізувати будь-який клієнт. У 2013 р. його відправили до підрозділу зі специфікацій Організації з удосконалення стандартів структурної інформації (OASIS) на доопрацювання. У 2019 р. OASIS випустила покращену версію MQTT 5. MQTT більше не є абревіатурою, але вважається офіційною назвою протоколу.

Протокол MQTT працює за моделлю «публікація-підписка». При традиційній взаємодії мережі клієнти і сервери пов'язуються між собою напряму. Клієнти запитують у сервера ресурси або дані, сервер обробляє запит і повертає відповідь. Але MQTT використовує шаблон «публікація-підписка», щоб відокремити відправника повідомлення (публіциста) від одержувача (підписника). Взаємодією між публіцистами та підписниками керує третій компонент – брокер повідомлень. Завдання брокера – відфільтрувати всі вхідні повідомлення від публіцистів та надіслати їх відповідним підписникам. Брокер відокремлює видавців від передплатників так.

Публіцист і підписник не знають про розташування один одного в мережі

та не обмінюються такою інформацією, як IP-адреса та номер порту. Публіцист та підписник не діють і не підключаються до мережі одночасно. Публіцисти та підписники можуть надсилати та отримувати повідомлення, не перериваючи роботу один одного. Наприклад, від передплатника не потрібно очікувати, коли видавець надішле повідомлення зображено на рисунку 2.1.

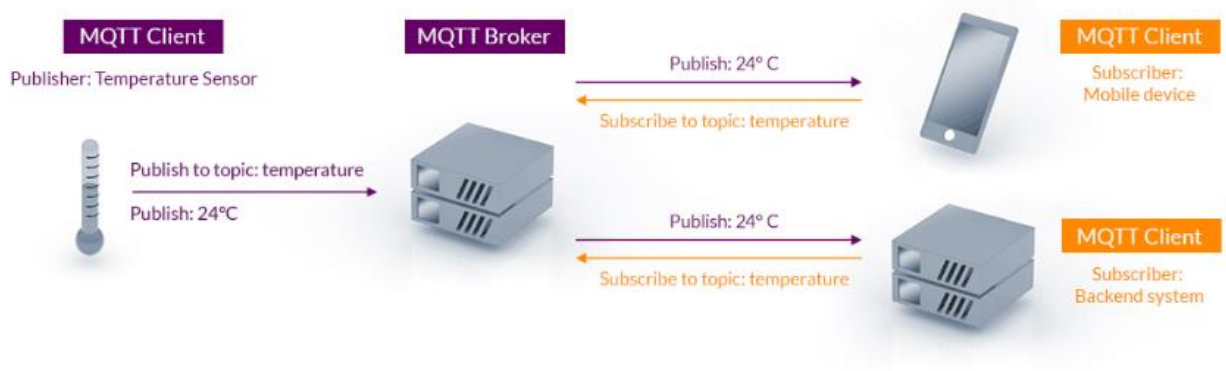



Рисунок 2.1 – MQTT Архітектура моделі «публіцист-підписник»

Протокол MQTT став стандартом для передачі даних IoT завдяки наведеним легкості та ефективності, масштабованості, надійності та безпеці.

Реалізація MQTT на пристрої IoT потребує мінімальних ресурсів і тому може використовуватися на невеликих мікроконтролерах. Наприклад, мінімальне керуюче повідомлення MQTT може складатися лише з двох байтів даних. Заголовки повідомлень MQTT також малі, тому можна оптимізувати пропускну здатність мережі [7].

Для реалізації MQTT потрібна мінімальна кількість коду, для роботи якого потрібно зовсім небагато енергії. Також у протоколі вбудовані функції для забезпечення взаємодії з великою кількістю пристроїв IoT. Тому можна реалізувати протокол MQTT для підключення мільйонів таких пристроїв.

Багато пристроїв IoT підключаються по ненадійних стільникових мереж з низькою пропускною здатністю та високою затримкою. У MQTT вбудовані функції, які скорочують час, який потрібен IoT для відновлення підключення до хмари. Також він визначає три різні рівні якості обслуговування, щоб забезпечити надійність прикладів використання IoT: максимум один раз (0), мінімум один раз (1) і рівно один раз (2).

MQTT спрощує для розробників завдання шифрування повідомлень та аутентифікації пристроїв та користувачів за допомогою сучасних протоколів аутентифікації, таких як OAuth, TLS1.3, керованих клієнтами сертифікатів та інших протоколів.

Деякі мови, наприклад Python, забезпечують хорошу підтримку протоколу MQTT. Тому розробники можуть швидко реалізувати його з мінімальною кількістю коду в будь-якому додатку.

Однією з основних характеристик MQTT є його легкість, що робить протокол придатним для пристроїв із низьким енергоспоживанням і обмеженими обчислювальними можливостями. Протокол використовує TCP, забезпечуючи надійність передачі, і підтримує різні рівні якості обслуговування (QoS), що дозволяє визначати ступінь надійності та гарантії доставки даних.

Переваги MQTT включають низькі накладні витрати, гнучкість у налаштуванні мережі, простоту використання та сумісність із різними мовами програмування. Завдяки моделі "публікація-підписка", протокол масштабований, що дозволяє розширювати мережу шляхом додавання нових клієнтів та тем без значних зусиль.

Однак у MQTT є і недоліки. Найбільш помітним є залежність від центрального брокера: якщо брокер виходить з ладу, вся комунікація припиняється, що створює єдину точку відмови. Безпека також може бути проблемою, оскільки протокол не має вбудованих механізмів для забезпечення шифрування або контролю доступу. Хоча можливе використання TLS для захисту, це додає накладних витрат та ускладнює впровадження. Ще однією проблемою є нестача стандартизації в назвах тем, що може призвести до плутанини в великих системах.

Загалом, MQTT є потужним інструментом для комунікації в IoT, особливо в сценаріях з низьким енергоспоживанням та обмеженою пропускною здатністю. Однак розробникам слід враховувати його недоліки, особливо в контексті безпеки та залежності від центрального брокера.

## 2.2 CoAP: опис протоколу, характеристики, переваги та недоліки

CoAP (Constrained Application Protocol) – це полегшений протокол, призначений для використання з пристроями з низьким енергоспоживанням та обмеженими ресурсами, такими як ті, що використовуються в Інтернеті речей (IoT). Він використовує модель "запит-відповідь", яка нагадує архітектуру REST, але оптимізована для роботи в умовах низької пропускної здатності та обмеженого енергоспоживання.

Він задумувався як простіша та ефективніша альтернатива HTTP для пристроїв з обмеженою обчислювальною потужністю, пам'яттю та часом автономної роботи. CoAP побудований поверх UDP (User Datagram Protocol) та надає набір методів для виявлення ресурсів, маніпулювання ними та спостереження за ними, а також підтримує асинхронну зв'язку та кешування.

Завдяки низьким накладним витратам та простоті CoAP стає все більш популярним у IoT-застосунках і, як очікується, відіграватиме важливу роль у майбутньому Інтернету речей.

CoAP працює поверх UDP (User Datagram Protocol) і зазвичай використовує порт 5683 для незахищеного зв'язку та порт 5684 для захищеного зв'язку з використанням DTLS (Datagram Transport Layer Security).

Він був розроблений целевою групою з розробки Інтернету (IETF) в робочій групі CoRE (Restricted RESTful Environments) з метою забезпечення ефективної зв'язку між IoT-пристроями через мережі з обмеженими можливостями, такі як низькоенергетичні бездротові мережі та інші середовища з обмеженими ресурсами [8].

Історія CoAP сягає 2010 року, коли в рамках IETF була сформована Основна робоча група. Робочій групі було дано завдання розробити протокол, який міг би забезпечити просте, ефективне та масштабоване рішення для зв'язку між IoT-пристроями в обмежених середовищах, де обмежені такі ресурси, як пропускна здатність, обчислювальна потужність та енергоспоживання.

У 2011 році перша версія CoAP, відома як CoAP version 13 (CoAP-v13), була опублікована як інформаційний документ RFC (Запит на коментарі), RFC 7252. Ця первісна версія CoAP ґрунтувалася на архітектурі REST (Representational State Transfer), яка є поширеним архітектурним стилем, що використовується в веб-застосунках, і запозичила багато концепцій з протоколу передачі гіпертексту (HTTP), який є стандартним протоколом, що використовується для зв'язку через Всесвітню павутину.

CoAP-v13 надав простий і полегшений протокол для IoT-пристроїв для виконання операцій CRUD (Створення, Витяг, Оновлення, Видалення) над ресурсами, ідентифікованими за допомогою уніфікованих ідентифікаторів ресурсів (URI), аналогічно операціям HTTP над ресурсами, ідентифікованими за URL.

Він використовував протокол користувацьких дейтаграм (UDP) як транспортний протокол, який є протоколом без встановлення з'єднання, ненадійним та легковесним, що підходить для обмежених середовищ.

CoAP-v13 також підтримував ряд типів повідомлень, включаючи підтверджувані (CON), непідтверджувані (NON), підтверджуючі (ACK) та скидні (RST) повідомлення, для забезпечення надійності та контролю перевантаження в ненадійних мережах.

Версія 18 CoAP (CoAP-v18), опублікована в 2014 році як RFC 7641, представила покращення в області безпеки, включаючи підтримку Datagram Transport Layer Security (DTLS) для забезпечення безпечного зв'язку між кінцевими точками CoAP.

Вона також додала нові функції, такі як:

– поблокова передача для ефективної обробки великих ресурсів;

– опція спостереження для отримання повідомлень, ініційованих сервером;

– групове спілкування для багатоадресного розсилання.

Версія 21 CoAP (CoAP-v21), опублікована в 2017 році як RFC 8323, додатково вдосконалила протокол, додавши функції та оптимізації:

– концепція CoAP поверх TCP дозволяє передавати повідомлення CoAP по протоколу TCP для надійної доставки;

– підтримка опції Proxy-Uri для проксирування повідомлень CoAP до інших кінцевих точок CoAP або інших протоколів, таких як HTTP;

– удосконалення для виявлення ресурсів та оновлення опції спостереження.

Ключові особливості протоколу CoAP перелічені нижче:

– легкий та ефективний: CoAP розроблений таким чином, щоб бути простим та легким, що дозволяє йому ефективно працювати на пристроях з обмеженими ресурсами, такими як пам'ять, обчислювальна потужність та час автономної роботи. Він використовує UDP як базовий транспортний протокол для мінімізації накладних витрат та зменшення затримки;

– спокійний: CoAP ґрунтується на архітектурі RESTful, яка широко використовується у веб-додатках. Він використовує URI для ідентифікації ресурсів та надає операції CRUD (Створення, вилучення, оновлення, видалення) для взаємодії з цими ресурсами;

– заснований на повідомленнях: CoAP використовує комунікаційну модель, засновану на повідомленнях, де кожне повідомлення є автономним блоком, що містить всю необхідну інформацію для її обробки одержувачем. Це робить його придатним для ненадійних мереж, де повідомлення можуть бути втрачені чи затримані;

– вбудована надійність: CoAP забезпечує надійність завдяки підтвердженням повідомлень, повторним передачам та контролю навантаження. Він використовує простий та ефективний механізм забезпечення надійності, що дозволяє уникнути непотрібних накладних витрат;

– безпека: CoAP підтримує безпеку транспортного рівня дейтаграм (DTLS) для забезпечення безпечного зв'язку протоколом UDP. Він також підтримує полегшені механізми безпеки, такі як попередній загальний ключ (PSK) та Необроблений відкритий ключ (RPK) для автентифікації кінцевих точок та безпечного зв'язку;

– виявлення ресурсів: CoAP надає простий та ефективний механізм для виявлення ресурсів у мережі. Він використовує формат базового посилання для опису ресурсів та їх властивостей;

– спостерігаючі ресурси: CoAP дозволяє клієнтам спостерігати за ресурсами в мережі та отримувати сповіщення при зміні ресурсів. Це корисно для програм реального часу, яким необхідно відстежувати зміни в середовищі та реагувати на них;

– підтримка багатоадресної розсилки: CoAP підтримує багатоадресний та груповий зв'язок, дозволяючи декільком пристроям отримувати те саме повідомлення при одній передачі. Це скорочує мережевий трафік та економить ресурси.

CoAP надає легкий та ефективний протокол для пристроїв та мереж з обмеженими ресурсами, підтримуючи при цьому такі важливі функції, як надійність, безпека, виявлення ресурсів та спостереження.

Цей протокол дозволяє пристроям взаємодіяти з іншими пристроями або веб-сервісами, використовуючи звичні HTTP-запити, такі як "GET", "POST", "PUT", "DELETE", але з меншим обсягом даних і зниженими вимогами до пропускної здатності. Завдяки цій гнучкості CoAP добре підходить для сітчастих мереж, де пристрої можуть комунікувати через інші пристрої, що особливо актуально для розумних будинків та великих систем з великою кількістю пристроїв.

Перевага CoAP полягає в його легкості та мінімальних накладних витратах, що робить його привабливим для пристроїв з низьким енергоспоживанням. Протокол також легко інтегрується з веб-технологіями, оскільки використовує модель REST, що спрощує розробку та впровадження. Проте CoAP має деякі недоліки. Через використання UDP виникає ризик втрати пакетів або збоїв у мережі, що може вимагати додаткових механізмів для забезпечення надійності. Крім того, хоча протокол може використовувати шифрування DTLS, це збільшує складність і накладні витрати, що може бути проблемою для пристроїв з обмеженими ресурсами.

Загалом, CoAP є корисним інструментом для передачі даних в контексті IoT, оскільки він поєднує в собі гнучкість і ефективність. Однак його використання може вимагати ретельного планування, особливо в аспектах надійності та безпеки.

## 2.3 LoRaWAN: опис протоколу, характеристики, переваги та недоліки

LoRaWAN – мережевий протокол, який характеризується низьким енергоспоживанням, використовуючи на фізичному рівні широкосмугову модуляцію LoRa. Безпосередньо модуляція LoRa забезпечує дальність передачі даних (до 15 км на відкритій місцевості, до 5 км у міській забудові), високу проникаючу здібність у міській забудові (можливе використання в глибоких підвалах без додаткових витрат), забезпечує захист сигналу від перешкод.

Протокол LoRaWAN – це програмний рівень, який визначає процес використання обладнання LoRa при передачі чи прийомі повідомлень (Рисунок 2.3). Він розроблений для того, щоб дозволити малопотужним пристроям обмінюватись даними з тими додатками, які підключені до Інтернету бездротовою мережею на великій відстані [9].

Популярність технології LoRaWAN в промисловості, «розумних містах», сільському господарстві зростає завдяки доступності протоколу бездротового двостороннього зв’язку, який покриває великий радіус при низькому енергоспоживанні (деякі пристрої працюють від батареї до 10 років).

Таким чином LoRaWAN вирішує найбільш гострі проблеми Інтернету речей для бізнесу:

– збір даних з чималої кількості пристроїв на великій території;

– збільшений строк служби кінцевих пристроїв (за рахунок низького енергоспоживання можлива робота до 10 років);

– економія засобів та часу (мережа розгортається швидко, легко масштабується, можливе дистанційне обслуговування).

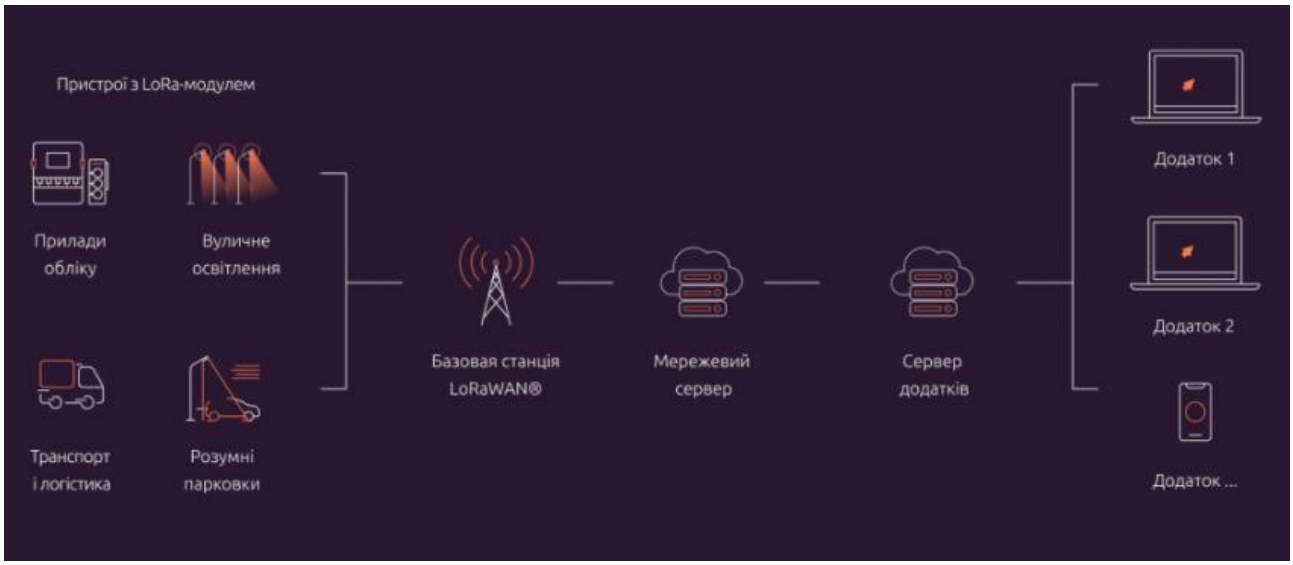



Рисунок 2.2 – Архітектура роботи мережі LoRaWAN

Кінцевий IoT-пристрій в мережі LoRaWAN (датчик, який виконує механізм або і те, і інше) підключається до бездротової мережі LoRaWAN через шлюзи, які працюють по модуляції LoRa. При цьому передача даних йде одразу на усі доступні в локації базові станції, а не на одну. Відсутність прив’язки до конкретного шлюзу дозволяє гарантувати передачу інформації й, при необхідності, контролювати датчик, який знаходиться у русі. Уся інформація, яка передається з кінцевого IoT-пристрою, захищена наскрізним шифруванням за допомогою 2 рівнів криптографічного захисту: 128-бітного мережевого ключа та 128-бітного ключа сесії додатку.

Кожний шлюз (базова станція) зареєстрований у мережі LoRaWAN і напряму пересилає отримані пакети даних на мережевий сервер (Network Server), використовуючи найбільш відповідний варіант підключення до мережі (3G, 4G, Wi-Fi, Ethernet, по оптоволокну або радіолінійному зв’язку).

Мережевий сервер (Network Server) керує всією мережею. Він отримує дані від шлюзів, видаляє дублікати повідомлень, пересилає отримані дані на відповідний сервер додатків (Application server), контролює швидкість передачі даних.

Сервер додатків (Application server) робить обробку отриманих даних і спрямовує їх на підключені кінцеві пристрої. Отримані дані можна

інтерпретувати та використовувати для вирішення бізнес-завдань.

На додаток до стрибкоподібної перебудови частоти пакети зв'язку між кінцевими пристроями і шлюзами також включають в себе параметр «Швидкість передачі даних» (Data Rate). Вибір DR дозволяє встановити динамічний компроміс між діапазоном зв'язку і тривалістю повідомлення. Крім того, завдяки технології з розширеним спектром, зв'язок з різними DR не заважає один одному і створює набір віртуальних «кодових» каналів, що збільшують пропускну здатність шлюзу. Щоб максимально збільшити час автономної роботи кінцевих пристроїв і загальну пропускну здатність мережі, мережевий сервер LoRaWAN управляє налаштуванням DR і вихідною потужністю RF для кожного кінцевого пристрою індивідуально за допомогою схеми адаптивної швидкості передачі даних (ADR).

Швидкість передачі даних в режимі LoRaWAN коливається від 0,3 кбіт / с до 50 Кбіт / с (в режимі FSK).

Існує 3 класи кінцевих пристроїв, які використовуються в мережі LoRaWAN: A, B та C. Вони відрізняються один від одного режимами прийому і передачі даних, енергоефективністю.

– клас A базовий клас, який повинен підтримуватись всіма пристроями. IoT-пристрої цього класу виходять у мережу по запланованому розкладу та мають 2 додаткових вікна для прослуховування мережі, одразу після відправлення даних. Мають найтриваліший термін служби, але задля направлення на пристрій повідомлення треба дочекатись найближчого запланованого часу виходу в мережу;

– клас B на відміну від класу A мають додаткові заплановані періоди прослуховування мережі, які відкриваються за розкладом. Пристрій синхронізується з базовою станцією через спеціальні маячки, дозволяючи знати точний час виходу пристрою на зв’язок;

– клас C кінцеві IoT-прилади цього класу прослуховують мережу увесь час, за винятком періоду передачі даних. Застосовуються у випадках, коли немає потреби економити енергію, але потрібно опитувати пристрій у незапланований

час.

Як і будь-яка інша технологія, LoRaWAN володіє своїми плюсами та мінусами, які визначають можливість використання для конкретного завдання. До сильних сторін технології LoRaWAN відносяться:

– висока дальність передачі даних, у порівнянні з іншими бездротовими технологіями;

– висока проникаюча спроможність у міській забудові;

– швидкість і простота розгортання мережі. Топологія «зірка» дозволяє швидко покривати великий радіус однією базовою станцією (шлюзом), не використовуючи додаткового обладнання;

– тривалий строк служби батареї. В залежності від використовуваного класу пристрою та заданих параметрів виходу у мережу, строк служби елемента живлення може досягати 10 років;

– простота масштабування;

– невисока вартість базових станцій і кінцевих пристроїв, у порівнянні зі слабкострумовими системами.

Як у будь-якій іншій системі, у LoRaWAN є і недоліки:

– затримка передачі даних від кінцевих пристроїв до додатка: від декількох секунд до декілька десятків секунд;

– невисока пропускна спроможність: від декількох сотень біт/с до декількох десятків кбіт/с.

Таким чином, LoRaWAN — це ефективний протокол для мереж на великих відстанях з низьким енергоспоживанням, який добре підходить для розподілених систем та застосувань у віддалених місцевостях. Незважаючи на обмеження, пов'язані з пропускною здатністю та затримками, він залишається популярним вибором для багатьох застосувань у сфері Інтернету речей.

## 2.4 Порівняльний аналіз протоколів MQTT, CoAP та LoRaWAN

MQTT, CoAP та LoRaWAN – це три популярні протоколи передачі даних

для Інтернету речей (IoT). Кожен з них має свої переваги та недоліки, які роблять їх придатними для різних випадків використання (таблиця 2.1).

MQTT – це легкий протокол обміну повідомленнями, який добре підходить для IoT-пристроїв з обмеженою пропускною здатністю. Він використовує модель публікатор-підписчик, в якій пристрої можуть публікувати дані на теми, а інші пристрої можуть підписуватися на ці теми, щоб отримувати дані.

Переваги MQTT:

– легкий та простий у використанні;
– ефективний у використанні пропускної здатності;;
– підтримує модель публікатор-підписчик
– підтримує високий рівень масштабованості.

Недоліки MQTT:

– не має вбудованих функцій безпеки;
– не підходить для передачі даних в режимі реального часу.

CoAP – це адаптований протокол HTTP, який розроблений для IoT-пристроїв. Він використовує формат JSON для передачі даних, що робить його зручним для використання з веб-сервісами.

Переваги CoAP:

– підтримує формат JSON;
– підходить для передачі даних в режимі реального часу;
– має вбудовані функції безпеки.

Недоліки CoAP:

– більш складний у використанні, ніж MQTT;
– менш ефективний у використанні пропускної здатності.

LoRaWAN – це протокол мережевого рівня, який розроблений для IoT-пристроїв, які працюють у широкомасштабних мережах. Він використовує розширення спектру з низькою пропускною здатністю, що дозволяє пристроям передавати дані на великі відстані.

Переваги LoRaWAN:

– великий радіус дії;

– низька вартість;

– високий рівень безпеки.

Недоліки LoRaWAN:

– низька пропускна здатність;

– не підходить для передачі даних в режимі реального часу.

Таблиця 2.1 – Порівняльна таблиця популярних протоколів MQTT, CoAP, LoRaWAN

| Протокол | Переваги | Недоліки |
|---|---|---|
| MQTT | Легкий, простий, економний, публікатор-підписчик, масштабований | Немає безпеки, не підходить для реального часу |
| CoAP | JSON, реальний час, безпека | Складний, менш економний |
| LoRaWAN | Великий радіус дії, низька вартість, безпека | Низька пропускна здатність, не підходить для реального часу |

# 3 ОПТИМІЗАЦІЯ ПРОТОКОЛІВ ПЕРЕДАЧІ ДАНИХ ДЛЯ ІОТ

## 3.1 Оптимізація розміру повідомлення

Зменшення розміру повідомлення є ключовою стратегією оптимізації протоколів передачі даних для Інтернету речей (IoT). Це особливо актуально в контексті пристроїв з обмеженими ресурсами та мереж із низькою пропускною здатністю. Основна ідея полягає в тому, щоб зменшити обсяг даних, які передаються між пристроями та мережами, з метою зниження накладних витрат, покращення пропускної здатності, зменшення затримок та економії енергії. Давайте розглянемо кілька підходів до зменшення розміру повідомлення в протоколах IoT [10].

Компресія даних – один із найпоширеніших способів зменшення розміру повідомлення. Вона дозволяє скоротити кількість бітів, необхідних для передачі інформації, шляхом усунення надмірності або використання ефективних алгоритмів кодування. Для IoT компресія може бути особливо корисною, оскільки більшість переданих даних мають прості структури або містять повторювані елементи. Компресія може бути застосована на рівні заголовків або на рівні корисного навантаження (payload).

Кодування даних – це ще один спосіб оптимізації, який дозволяє зменшити розмір повідомлень шляхом застосування ефективних схем кодування. Наприклад, протоколи можуть використовувати бінарне кодування замість текстового, щоб зменшити кількість бітів, необхідних для передачі. Кодування також може включати усунення непотрібних полів або заміну довгих ключових слів короткими кодами. Відповідний вибір кодування може значно скоротити розмір повідомлень.

Заголовки повідомлень можуть становити значну частину переданих даних, особливо в протоколах з великими накладними витратами. Мінімізація заголовків може допомогти зменшити розмір повідомлення. Це може бути досягнуто шляхом видалення надмірних полів, використання коротких

ідентифікаторів, а також оптимізації формату заголовків. Деякі протоколи, такі як CoAP, спеціально розроблені з мінімальними заголовками, щоб зменшити накладні витрати.

Усунення надмірності в даних — ще один спосіб зменшити розмір повідомлення. Це можна зробити шляхом видалення дублюючих або непотрібних елементів, використання більш ефективних структур даних, або об'єднання інформації. В деяких випадках можливо передавати лише змінені дані замість повних повідомлень, що може значно зменшити обсяг передачі.

Для протоколів, таких як MQTT, де використовується модель "публікація-підписка", використання коротких топіків та ідентифікаторів може зменшити розмір повідомлення. Це також стосується унікальних ідентифікаторів пристроїв або транзакцій, які можуть бути скорочені для зменшення заголовків.

Інші підходи до зменшення розміру повідомлень включають використання компактних форматів даних, таких як MessagePack або Protocol Buffers, які дозволяють зменшити накладні витрати в порівнянні з традиційними текстовими форматами, як-от JSON або XML. Компактні формати дозволяють передавати більше інформації за менший обсяг даних, що є критичним для пристроїв з обмеженими ресурсами та мереж з низькою пропускною здатністю.

Таким чином виділимо основні способи зменшити розмір повідомлення:

– видалення непотрібних даних, тому що багато IoT-пристроїв збирають більше даних, ніж їм дійсно потрібно, важливо ретельно аналізувати дані, які збираються, щоб видалити будь-які непотрібні або дублюючіся дані;

– стиснення даних за допомогою алгоритмів стиснення даних, які можна використовувати для зменшення розміру повідомлення. Деякі популярні алгоритми стиснення даних для IoT включають gzip, Zlib та Brotli;

– використання форматів даних з низьким рівнем бітів: Деякі формати даних, такі як JSON та XML, можуть бути дуже неефективними з точки зору використання бітів. Використання форматів даних з низьким рівнем

бітів, таких як Protobuf або MessagePack, може значно зменшити розмір повідомлення;

– кодування даних може допомогти зменшити розмір повідомлення за рахунок перетворення даних у більш компактний формат. Деякі популярні методи кодування даних для IoT включають Base64 та URL-кодування.

Важливо зазначити, що зменшення розміру повідомлення може призвести до деяких компромісів. Наприклад, стиснення даних може призвести до незначної втрати даних. Використання форматів даних з низьким рівнем бітів може зробити дані складнішими для обробки.

## 3.2 Стиснення даних

Завдяки стисненню можна значно зменшити обсяг даних, що передаються між пристроями та мережею, що має ряд переваг, зокрема зниження навантаження на мережу, швидшу передачу, зменшення енергоспоживання та покращення ефективності сховищ. Це особливо актуально для пристроїв з обмеженими ресурсами та мереж із низькою пропускною здатністю.

Існує кілька методів стиснення даних, що застосовуються в IoT. Безвтратне стиснення дозволяє зменшити обсяг даних без втрати інформації, використовуючи алгоритми, як-от Huffman coding або Deflate (рисунок 3.1). Стиснення з втратами, яке часто використовується для зображень чи аудіо, дає більшу ступінь стиснення, але з можливими втратами якості або точності [11].

Стиснення даних може застосовуватися в різних протоколах IoT. Для MQTT, який не має вбудованого стиснення, компресія на рівні корисного навантаження може значно зменшити розмір повідомлень. У CoAP, легкому протоколі, стиснення корисного навантаження допомагає знизити накладні витрати [12]. Для LoRaWAN, що працює в умовах низької пропускної здатності, стиснення даних може підвищити ефективність передачі на великі відстані.

Однак стиснення даних має певні виклики та обмеження. Серед них —

накладні витрати на декодування або розпакування, які можуть збільшити час обробки даних, а також ускладнення протоколів, що може вплинути на стабільність та надійність системи. Крім того, використання стиснення з втратами може призвести до втрати важливої інформації.

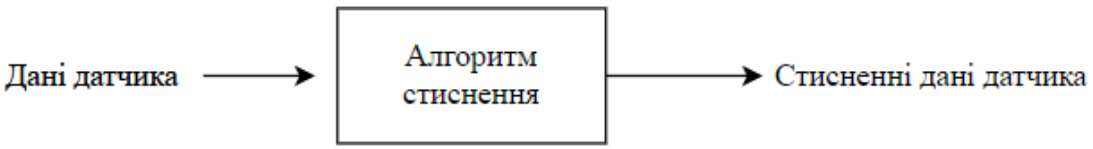



Рисунок 3.1 – Безвтратний метод сттиснення даних

Деякі популярні алгоритми стиснення даних для IoT включають:

– GZIP - це загальнодоступний алгоритм стиснення даних, який добре підходить для текстових даних.

– Zlib - ще один загальнодоступний алгоритм стиснення даних, який можна використовувати для різних типів даних, включаючи текстові, двійкові та зображення.

– Brotli - новий алгоритм стиснення даних, який може забезпечити кращий коефіцієнт стиснення, ніж GZIP, особливо для текстових даних.

– LZW - це алгоритм стиснення даних без втрат, який добре підходить для даних з повторюваними шаблонами.

– RLE - це алгоритм стиснення даних без втрат, який добре підходить для даних з довгими послідовностями однакових значень.

Вибір алгоритму стиснення даних для IoT залежить від декількох факторів:

Тип даних: Різні алгоритми стиснення даних краще підходять для різних типів даних.

Коефіцієнт стиснення: Деякі алгоритми стиснення даних можуть забезпечити кращий коефіцієнт стиснення, ніж інші, але вони можуть бути більш обчислювально витратними.

Швидкість стиснення та розпакування: Деякі алгоритми стиснення даних можуть бути швидшими за інші, але вони можуть забезпечити гірший коефіцієнт стиснення.

## 3.3 Адаптивна швидкість передачі даних

Адаптивна швидкість передачі даних передбачає можливість IoT-пристроям автоматично та динамічно змінювати свою швидкість передачі даних (так званий "data rate") на основі різних факторів, таких як якість сигналу, рівень перешкод, пропускна здатність мережі та стан пристроїв. Це дозволяє оптимізувати передачу даних, забезпечуючи максимальну ефективність при мінімальних витратах енергії та мережевих ресурсів.

Адаптивна швидкість передачі даних дає низку переваг. Вона підвищує ефективність, дозволяючи використовувати ресурси мережі більш раціонально, зменшуючи затримки і забезпечуючи оптимальну пропускну здатність. Завдяки можливості знижувати швидкість передачі, якщо це можливо, зменшується енергоспоживання, що критично важливо для пристроїв, що працюють від батарейок. Крім того, цей підхід підвищує надійність передачі даних, оскільки при погіршенні якості сигналу або збільшенні перешкод можна знизити швидкість, щоб уникнути втрати даних.

Існує декілька способів реалізувати адаптивну швидкість передачі даних для IoT:

– IoT-пристрій може використовувати алгоритм для визначення оптимальної швидкості передачі даних залежно від умов мережі та типу даних, які передаються;

– IoT-пристрій може отримувати зворотний зв'язок від мережі про те, яка швидкість передачі даних є оптимальною;

– IoT-пристрій може бути налаштований на використання певної політики адаптивної швидкості передачі даних.

Цей підхід можна застосовувати в різних протоколах IoT. LoRaWAN, наприклад, має вбудовану підтримку адаптивної швидкості передачі, що дозволяє пристроям змінювати швидкість відповідно до умов мережі. У Wi-Fi адаптивна швидкість допомагає зберігати стабільність з'єднання при

коливаннях якості сигналу. Bluetooth Low Energy (BLE) також використовує цей підхід, що дозволяє зменшити енергоспоживання.

Однак, використання адаптивної швидкості передачі даних може мати деякі виклики. Серед них — складність реалізації, оскільки необхідно розробляти алгоритми, що аналізують мережеві умови та вирішують, коли змінити швидкість. Зміна швидкості може викликати затримки, що може бути проблематичним для систем з високими вимогами до часу реакції. Також неправильне налаштування або часті зміни швидкості можуть негативно вплинути на стабільність мережі.

## 3.4 Балансування навантаження

Балансування навантаження в контексті оптимізації протоколів передачі даних для Інтернету речей (IoT) – це підхід, що дозволяє розподіляти трафік даних рівномірно по мережі для підвищення ефективності та надійності. У системах IoT балансування навантаження допомагає уникнути "вузьких місць", підвищити пропускну здатність, зменшити затримки та забезпечити стабільну роботу мережі навіть при високому навантаженні. Розглянемо концепцію балансування навантаження, його переваги, виклики та способи застосування в IoT.

Балансування навантаження може бути реалізовано на різних рівнях мережі. На рівні пристроїв це може означати розподіл обробки та передачі даних між кількома пристроями, щоб уникнути перевантаження одного з них. На рівні мережевих пристроїв, таких як шлюзи або брокери, балансування навантаження може включати розподіл вхідних запитів або трафіку між кількома вузлами, щоб забезпечити рівномірний розподіл навантаження.

Переваги балансування навантаження очевидні. По-перше, рівномірний розподіл трафіку знижує ризик перевантаження окремих вузлів, що підвищує стабільність та надійність мережі. Це також зменшує затримки у передачі даних, оскільки немає необхідності чекати на обробку на перевантажених

вузлах. Крім того, балансування навантаження дозволяє мережі масштабуватися, оскільки додавання нових пристроїв або вузлів не призводить до диспропорційного збільшення навантаження на окремі компоненти системи.

Способи застосування балансування навантаження можуть бути різними в залежності від конкретного протоколу та архітектури мережі. Наприклад, у протоколі MQTT балансування навантаження може бути реалізовано шляхом використання кількох брокерів, між якими розподіляється трафік. Це забезпечує відмовостійкість та покращує пропускну здатність [13]. У LoRaWAN балансування навантаження може включати розподіл пристроїв між кількома шлюзами, щоб уникнути перевантаження одного з них. Такий підхід важливий для великих мереж IoT, де велика кількість пристроїв може створити значне навантаження на окремі вузли.

Попри переваги, балансування навантаження має свої виклики. Складність реалізації може бути значною, особливо якщо необхідно координувати роботу кількох пристроїв або вузлів. Це вимагає розробки складних алгоритмів та може підвищити накладні витрати. Крім того, балансування навантаження може потребувати додаткових ресурсів, що може бути проблематичним у контексті пристроїв з обмеженими ресурсами.

В цілому, балансування навантаження є важливим підходом для оптимізації протоколів передачі даних в IoT. Воно дозволяє забезпечити рівномірний розподіл трафіку, підвищуючи ефективність та надійність мережі. Однак його реалізація потребує ретельного планування, врахування потенційних викликів та використання відповідних методів для забезпечення стабільності мережі навіть при високому навантаженні.

## 3.5 Паралельна передача – використання декількох потоків

Паралельна передача даних – це підхід, що дозволяє розподіляти трафік на кілька незалежних потоків, які можуть передаватися одночасно. Такий підхід особливо корисний для оптимізації протоколів передачі даних в Інтернеті речей

(IoT), оскільки він підвищує пропускну здатність, зменшує затримки та підвищує загальну ефективність мережі. Використання кількох потоків дає змогу передавати більше даних одночасно, що значно зменшує загальний час передачі. Це корисно в системах IoT, де багато пристроїв взаємодіють між собою або де існує високий рівень навантаження на мережу [14].

В основі паралельної передачі даних лежить принцип поділу даних на кілька потоків, які потім передаються одночасно різними каналами або шляхами. Це може бути реалізовано кількома способами: використанням кількох мережних інтерфейсів, наприклад якщо пристрій IoT має кілька мережних інтерфейсів, наприклад, Ethernet та Wi-Fi, дані можна розділити на потоки та надсилати по кожному інтерфейсу одночасно; поділом частот: в бездротових мережах можна використовувати кілька каналів для передачі різних потоків даних; та мультиплексування (рисунок 3.2): технології мультиплексування, такі як тимчасовий поділ каналів (TDMA) або частотний поділ каналів (FDM) передбачає розміщення в межах смуги пропускання вихідного каналу зв'язку декількох каналів зв'язку з меншою шириною, дозволяють кільком потокам даних ділити один канал зв'язку за часом або частотою [15].

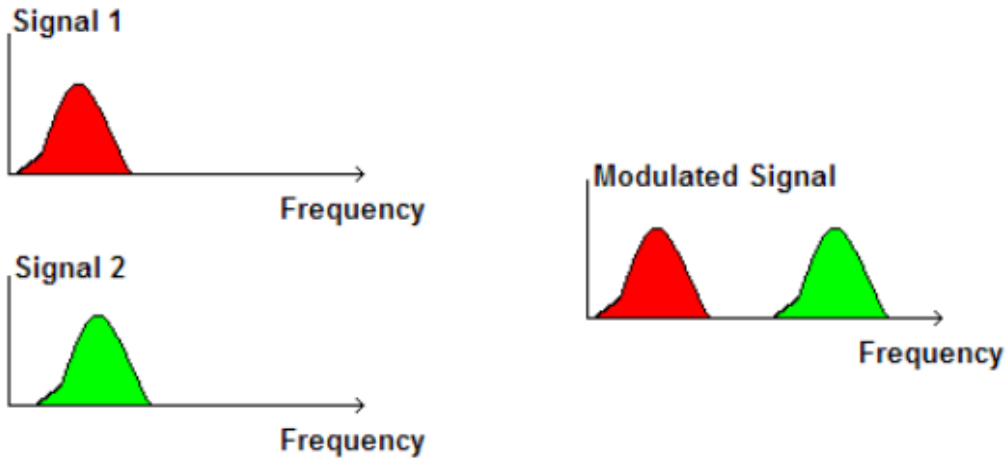



Рисунок 3.2 – Мультиплексування двох каналів з поділом по частоті

Після поділу даних на потоки необхідно вибрати канали передачі кожного потоку. Це можуть бути фізичні інтерфейси, частотні та віртуальні

канали:

– Пристрої IoT можуть мати кілька фізичних мережевих інтерфейсів, таких як Ethernet, Wi-Fi, Bluetooth або стільниковий зв'язок. Кожен потік даних може бути надіслано по одному з цих інтерфейсів;

– У бездротових мережах можна використовувати кілька каналів для передачі різних потоків даних;

– Технології віртуалізації, такі як VLAN, можуть бути використані для створення віртуальних каналів поверх одного фізичного інтерфейсу.

Перед відправкою каналами зв'язку потоки даних мультиплексуються в єдиний потік. Це може бути зроблено за допомогою різних методів, таких як тимчасовий поділ каналів (TDMA), частотний поділ каналів (FDM) або кодування з ортогональним частотним поділом доступу (OFDMA) [16].

На приймальній стороні потоки даних демультиплексуються з єдиного потоку. Це дозволяє відновити вихідні потоки даних.

Для узгодженої передачі даних між пристроями IoT необхідно синхронізувати потоки даних. Це можна зробити за допомогою різних методів, таких як синхронізація за часом або синхронізація за послідовністю пакетів.

Під час передачі можуть виникати помилки, такі як втрата пакетів або спотворення даних [17]. Для забезпечення надійної передачі даних потрібно обробляти помилки. Це можна зробити за допомогою різних методів, таких як повторна передача пакетів або перевірка цілісності даних.

Переваги паралельної передачі даних для IoT:

– збільшення пропускної здатності: Поділ даних на потоки дозволяє використовувати кілька каналів одночасно, що призводить до значного збільшення загальної пропускної спроможності. Це особливо важливо для програм, де потрібна передача великих обсягів даних, таких як потокове відео або телемедицина;

– зниження затримки: При передачі великих об'ємів даних паралельна передача може знизити затримку, оскільки дані розбиваються на пакети, які надсилаються різними каналами. Це призводить до скорочення часу,

необхідного для доставки кожного пакета, що зменшує загальну затримку;

– підвищення надійності: Використання кількох потоків даних підвищує надійність системи. Якщо один із каналів стає недоступним, дані все одно можуть бути передані іншими каналами, що забезпечує безперебійну роботу системи. Це особливо важливо для критично важливих програм, де неприпустимі простої;

– підтримка додатків із високими вимогами: Паралельна передача даних ідеально підходить для додатків IoT з високими вимогами до продуктивності, де необхідні низькі затримки та висока пропускна спроможність, таких як системи керування в режимі реального часу, автономні транспортні засоби та доповнена реальність;

– ефективне використання мережевих ресурсів: Поділ трафіку на кілька потоків дозволяє ефективніше використовувати мережні ресурси, розподіляючи навантаження між різними інтерфейсами чи частотами. Це може допомогти знизити навантаження мережі та покращити загальну продуктивність;

– підтримка масштабованості: Паралельна передача даних підтримує масштабованість, що дозволяє легко додавати нові канали або потоки в міру зростання вимог до пропускної здатності. Це робить її ідеальною для мереж IoT, які поступово розширюються.

Паралельна передача забезпечує підвищену пропускну здатність, оскільки кілька потоків можуть працювати одночасно, не чекаючи завершення передачі іншими потоками. Це також зменшує затримки, оскільки кожен потік може передаватися незалежно. Крім того, паралельна передача підвищує надійність, оскільки якщо один потік з якихось причин перерветься, інші можуть продовжувати передавати інформацію, знижуючи ризик втрати даних.

Цей підхід може бути застосований у різних протоколах IoT. У MQTT паралельна передача може бути реалізована за рахунок використання кількох брокерів або каналів, що дозволяє паралельно передавати дані між різними темами. CoAP може використовувати паралельну передачу через одночасні

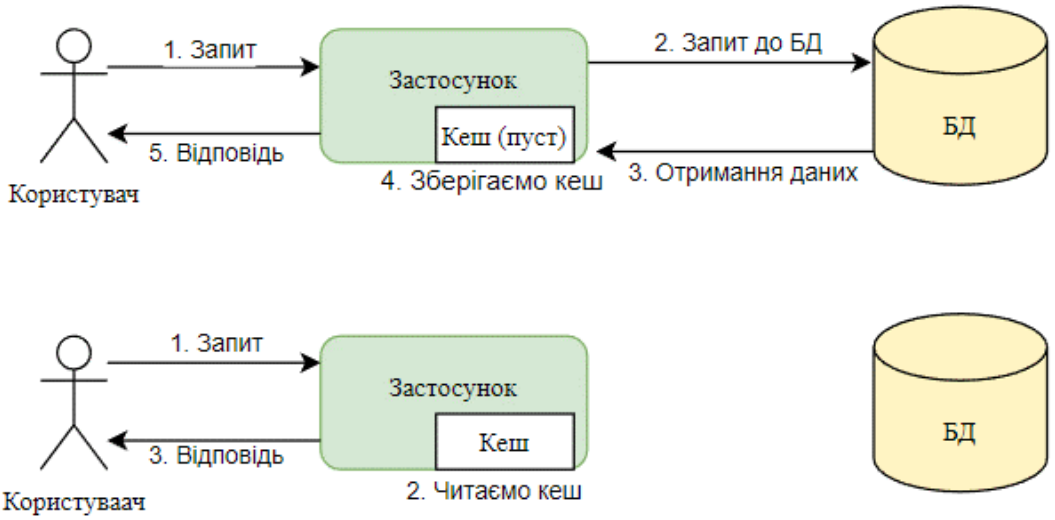
запити, що підвищує швидкість передачі. У LoRaWAN паралельна передача допомагає розподілити навантаження між кількома шлюзами, зменшуючи ризик перевантаження окремих вузлів.

Варто однак зауважити, що використання паралельної передачі даних має свої виклики. З одного боку, вона може ускладнити мережу та вимагати складних алгоритмів для координації передачі даних. З іншого боку, паралельна передача може призвести до конкуренції за мережеві ресурси, що потребує ретельного управління для запобігання конфліктів. Також є ризик фрагментації даних, коли розподіл інформації на кілька потоків може ускладнити відновлення контексту або порядок передачі.

## 3.6 Кешування

Кешування – це техніка, яка використовується в комп'ютерних системах і мережах для зберігання копій даних або ресурсів у швидкому доступі з метою підвищення швидкості та ефективності передачі даних. У контексті Інтернету речей (IoT), кешування може бути потужним інструментом для оптимізації протоколів передачі даних, оскільки воно допомагає зменшити навантаження на мережу, знизити затримки та підвищити продуктивність. Кешування може бути реалізовано на різних рівнях архітектури IoT, від пристроїв до мережевих компонентів [18].

Суть кешування полягає в тому, щоб зберігати копії даних, які можуть часто використовуватися або запитуватися, у швидкодоступному сховищі (рисунок 3.3).




БД – база даних

Рисунок 3.3 – Приклад роботи кешу. На першій схемі відображена первинна робота з кешем, коли він пустий. Після запиту до БД дані зберігаються у кеш, таким чином на другій схемі на вже не потрібно звертатись знову до БД, що пришвидчує відповідь та економить ресурси

Коли дані вже закешовані, системи IoT можуть отримувати до них доступ без повторного запиту через мережу або обчислювальних операцій, які були б необхідні для їх відновлення. Це значно знижує навантаження на мережу і зменшує затримки[19].

Для пристроїв IoT кешування може бути використано для зберігання часто використовуваних даних або результатів обчислень, що зменшує потребу в передачі даних через мережу. Також кешування може бути реалізовано на рівні шлюзів або проміжних пристроїв, щоб зберігати дані, які можуть знадобитися для кількох пристроїв або запитів. Це допомагає знизити навантаження на мережу та підвищити швидкість обробки. У хмарних інфраструктурах кешування може бути використано для зберігання результатів обчислень або даних, які часто запитуються, що дозволяє знизити навантаження на сервери та підвищити швидкість відповіді (рисунок 3.4).

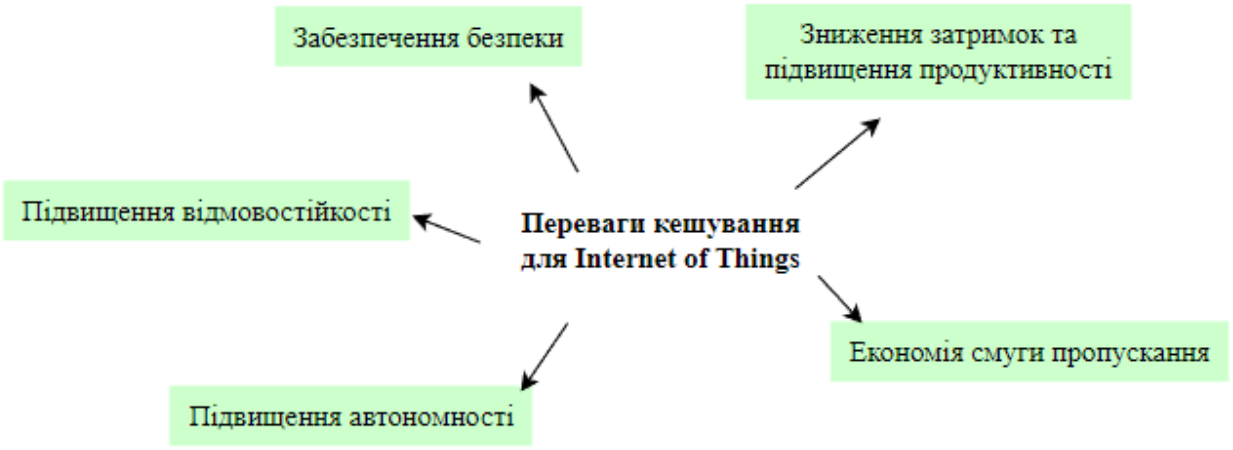



Рисунок 3.4 – Список переваг кешування

Пристрої IoT генерують і передають величезні обсяги даних. Для забезпечення безперебійної роботи, низької затримки та аналітичних можливостей у режимі реального часу необхідні ефективні рішення щодо управління даними. Кешування даних виступає як потужний інструмент, здатний оптимізувати роботу IoT-систем на різних рівнях.

Кешування даних, що часто використовуються, локально на пристрої або на прикордонному сервері значно скорочує час доступу до них, що призводить до зниження затримок і підвищення загальної продуктивності системи. Це особливо важливо для програм, де критична швидкість реагування, таких як системи керування в режимі реального часу або автономні транспортні засоби.

Передача даних у пристроях IoT може бути дорогою, особливо в мережах з обмеженою пропускною здатністю. Кешування даних, до якого часто звертаються, може допомогти знизити обсяг передаваних даних, тим самим заощаджуючи смугу пропускання та витрати.

Багато IoT-пристроїв працюють від батарей. Кешування даних локально може допомогти знизити енергоспоживання, пов'язане з доступом до даних віддалених джерел, тим самим продовжуючи термін служби батареї.

У разі збоїв мережі або перебоїв у роботі сервера кешовані дані будуть доступні локально, що забезпечить безперебійну роботу пристрою.

Кешування даних локально може допомогти підвищити безпеку системи,

знижуючи залежність від віддалених серверів, які можуть бути вразливими для кібератак.

Тож основні переваги кешування в IoT включають зниження навантаження на мережу, зменшення затримок, підвищення продуктивності та зниження енергоспоживання. Кешування також може підвищити надійність, оскільки дозволяє швидко отримувати дані з локального сховища, уникаючи можливих збоїв у мережі. Реалізація ефективного кешування може вимагати складних алгоритмів для управління даними та їх оновлення, що може збільшити накладні витрати. Також існують проблеми з узгодженістю, оскільки закешовані дані можуть застаріти, призводячи до використання некоректної інформації. Крім того, зберігання даних у кеші може підвищити ризик несанкціонованого доступу, що вимагає додаткових заходів безпеки.

## 3.7 Мінімізація запитів

Мінімізація запитів – це підхід до оптимізації протоколів передачі даних, який має на меті зменшити кількість запитів між пристроями та сервером або між різними компонентами мережі. У контексті Інтернету речей (IoT) мінімізація запитів допомагає знизити навантаження на мережу, скоротити затримки, зменшити енергоспоживання та підвищити ефективність системи в цілому. Такий підхід особливо важливий для систем IoT, де пропускна здатність та енергоспоживання можуть бути обмеженими [20].

Суть мінімізації запитів полягає в тому, щоб зменшити кількість комунікаційних операцій, потрібних для виконання певного завдання або передачі даних. Це можна досягти кількома способами, включаючи об'єднання запитів, використання кешування, зменшення розміру переданої інформації та оптимізацію протоколів.

Один зі способів мінімізації запитів – це об'єднання кількох запитів в один. Це може бути особливо корисно, коли пристрій IoT потребує отримання або відправлення кількох частин інформації. Замість того, щоб надсилати

кілька окремих запитів, можна об'єднати їх в один запит, що знижує навантаження на мережу та затримки у передачі. Наприклад, протоколи, такі як CoAP, можуть використовувати багатокадрові запити для передачі більшої кількості інформації в одному запиті.

Кешування також допомагає мінімізувати кількість запитів, оскільки часто використовувані дані можна зберігати локально, зменшуючи потребу у повторних запитах до сервера або мережі. Це не тільки знижує навантаження на мережу, але й скорочує затримки, оскільки дані можуть бути отримані з кешу.

Зменшення розміру переданої інформації, наприклад, через стиснення даних або використання більш ефективних форматів, також сприяє мінімізації запитів. Менший обсяг даних означає меншу кількість запитів або більш ефективну передачу, що позитивно впливає на пропускну здатність та енергоспоживання.

Оптимізація протоколів передачі даних для IoT може передбачати мінімізацію накладних витрат і кількості переданих заголовків, що також допомагає зменшити кількість запитів. Це особливо актуально для протоколів, які мають великі накладні витрати або вимагають багато інформації для кожного запиту.

Попри переваги, мінімізація запитів може мати деякі виклики. Об'єднання кількох запитів в один може підвищити складність обробки та вимагати більш складних алгоритмів. Кешування може призвести до проблем із узгодженістю, якщо дані застарівають або не оновлюються вчасно. Зменшення розміру даних може потребувати додаткових обчислювальних ресурсів, що може збільшити енергоспоживання або затримки.

Таким чином, мінімізація запитів – це ефективний підхід до оптимізації протоколів передачі даних в IoT, який дозволяє знизити навантаження на мережу, зменшити затримки та підвищити загальну ефективність системи. Однак для успішної реалізації необхідно враховувати можливі виклики та знаходити баланс між ефективністю, стабільністю та складністю реалізації.

# ВИСНОВКИ

Оптимізація протоколів передачі даних для Інтернету речей (IoT) є складним і багатоаспектним завданням, яке вимагає розуміння не лише технічних, але й економічних, екологічних та соціальних факторів. Розглядаючи цю тему, ми повинні враховувати різноманітні вимоги, такі як ефективність, надійність, енергоспоживання, безпека, а також узгодженість і сумісність між пристроями різних виробників. Така комплексність пояснюється унікальними умовами, в яких працюють пристрої IoT, а також їх широким спектром застосувань.

Для ефективної оптимізації протоколів передачі даних важливо враховувати характер навантаження, типи трафіку, обмеження на пропускну здатність і енергоспоживання, а також вимоги щодо безпеки. Стиснення даних може допомогти зменшити обсяг переданої інформації, але має бути збалансоване з накладними витратами на декомпресію та ризиком втрати даних. Адаптивна швидкість передачі даних дозволяє мережі реагувати на зміни умов, підвищуючи ефективність, але вимагає складних алгоритмів та може викликати затримки. Паралельна передача даних, яка використовує кілька потоків, дозволяє мережі ефективніше розподіляти навантаження та знижувати час передачі, що важливо для великих мереж IoT з високим рівнем взаємодії між пристроями. Кешування допомагає зменшити навантаження на мережу, дозволяючи пристроям зберігати часто використовувані дані у швидкодоступних сховищах, що також знижує затримки та енергоспоживання. Балансування навантаження дозволяє розподіляти трафік рівномірно по мережі, підвищуючи стабільність і надійність.

Однак кожен із цих підходів має свої виклики. Стиснення може збільшити накладні витрати на декомпресію та підвищити складність протоколів. Адаптивна швидкість передачі може вимагати складних алгоритмів та призвести до затримок при зміні швидкості. Паралельна передача даних може викликати конкуренцію за ресурси, а кешування – проблеми з

узгодженістю або безпекою. Балансування навантаження також може ускладнити мережу та потребувати додаткового управління.

Особлива увага повинна приділятися безпеці та конфіденційності. Оскільки пристрої IoT можуть передавати чутливі дані, протоколи передачі мають забезпечувати належний рівень захисту, включаючи шифрування, аутентифікацію та контроль доступу. Важливо, щоб оптимізація не призводила до зниження рівня безпеки, особливо в контексті кібербезпеки.

Інший аспект, який потребує уваги, – це сумісність та інтероперабельність між пристроями різних виробників. Оптимізація протоколів передачі даних повинна враховувати потребу в стандартизації та сумісності, щоб уникнути створення "островів" окремих систем, які не можуть взаємодіяти між собою.

Соціальні та екологічні аспекти також є важливими. Оптимізація повинна сприяти більш ефективному використанню ресурсів, зменшенню енергоспоживання та зниженню викидів вуглецю. Це особливо важливо в контексті розумних міст та екологічних застосувань IoT, де стійкість і екологічна відповідальність є пріоритетами.

Отже, оптимізація протоколів передачі даних для IoT є багатогранним завданням, що вимагає комплексного підходу. Розглядаючи всі ці аспекти, можна розробити ефективні, надійні та безпечні рішення, які сприятимуть розвитку IoT та його застосуванню в різних сферах, таких як промисловість, медицина, розумні міста, сільське господарство та інші. Таким чином, головним завданням оптимізації є забезпечення балансу між ефективністю, надійністю, енергоспоживанням, безпекою та сумісністю, що дозволить розширити можливості IoT та забезпечити стійкий розвиток технологій.

# ПЕРЕЛІК ПОСИЛАНЬ


1. What Is IoT (Internet of Things)?. Cisco. [Електронний ресурс]. – Режим доступу: https://www.cisco.com/c/en/us/solutions/internet-of-things/what-is-iot.html (дата звернення 11.05.2024).
2. The Rise of “Internet of Things”: Review and Open Research Issues Related to Detection and Prevention of IoT-Based Security Attacks. Publishing Open Access research journals & papers | Hindawi. [Електронний ресурс]. – Режим доступу: https://www.hindawi.com/journals/wcmc/2022/8669348/ (дата звернення 11.05.2024).
3. Elgar Fleisch,Marc Langheinrich,Christian Floerkemeier. The Internet of Things. Springer, 2008.
4. Number of IoT Devices (2024). [Електронний ресурс]. – Режим доступу: https://explodingtopics.com/blog/number-of-iot-devices (дата звернення 27.05.2024).
5. Комунікаційний протокол. [Електронний ресурс]. – Режим доступу: https://en.wikipedia.org/wiki/Communication_protocol (дата звернення 27.05.2024).
6. MQTT - The Standard for IoT Messaging. MQTT - The Standard for IoT Messaging. [Електронний ресурс]. – Режим доступу: https://mqtt.org/ (дата звернення: 14.05.2024).
7. Security as a CoAP resource: An optimized DTLS implementation for the IoT / A. Capossele et al. 2015 IEEE International Conference on Signal Processing for Communications (ICC), London, 8–12 June 2015. 2015. [Електронний ресурс]. – Режим доступу: https://doi.org/10.1109/icc.2015.7248379 (дата звернення: 14.05.2024).
8. Understanding the Limits of LoRaWAN / F. Adelantado et al. IEEE Communications Magazine. 2017. Vol. 55, no. 9. P. 34–40. [Електронний ресурс]. – Режим доступу: https://doi.org/10.1109/mcom.2017.1600613 (дата звернення 15.05.2024).

9. David Barnett. Comparison of MQTT and DDS as M2M Protocols for the Internet of Things. Published on May 29, 2013.
10. Optimization method of Data interaction in power IoT based on particle swarm algorithm. IEEE Xplore. [Електронний ресурс]. – Режим доступу: https://ieeexplore.ieee.org/document/9712745 (дата звернення: 16.05.2024)
11. Lu Tan, Neng Wang. Future internet: The Internet of Things // 2010 3rd International Conference on Advanced Computer Theory and Engineering(ICACTE). — IEEE, 2010-08
12. Curley R. Breakthroughs in telephone technology: From Bell to smartphones. New York, NY : Britannica Educational Pub. in association with Rosen Educational Services, 2012.
13. Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, Marimuthu Palaniswami. Internet of Things (IoT): A vision, architectural elements, and future directions // Future Generation Computer Systems. — 2013-09. — Т. 29, вып. 7. — С. 1645–1660. — ISSN 0167-739X.
14. Agustina Calatayud. The Connected Supply Chain: Enhancing Risk Management in a Changing World. — Inter-American Development Bank, 2017-03.
15. Home Automation System // Embedded Systems and Robotics with Open Source Tools. — Boca Raton : CRC Press, 2016.: CRC Press, 2018-09-03. — С. 109–120.
16. Брайчевський, С. М. (2019). Проблема персональних даних в системах Інтернету речей з елементами штучного інтелекту. Інформація і право. 4 (31). doi:10.37750/2616-6798.2019.4(31).194348. Архів оригіналу за 2 жовтня 2021. Процитовано 2 жовтня 2021.
17. Roy H. Data Compression in Digital Systems. Springer Science & Business Media, 2012. 431 p.
18. Bilal M., Kang S.-G. A Cache Management Scheme for Efficient Content Eviction and Replication in Cache Networks. IEEE Access. 2017. Vol. 5. P. 1692–1701. [Електронний ресурс]. – Режим доступу:

https://doi.org/10.1109/access.2017.2669344 (дата звернення: 14.05.2024).

19. Вейвлет-перетворення у компресії та попередній обробці зображень / О. В. Капшій, О. І. Коваль, Б. П. Русин ; НАН України, Фіз.-мех. ін-т ім. Г. В. Карпенка. — Львів : Сполом, 2008. — 206 с. : іл., табл., портр. ; 22 см. — Бібліогр.: с. 187—203 (238 назв). — 300 пр. — ISBN 978-966-665-554-0