Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет __Комп'ютерної інженерії та управління_____
(повна назва)

Кафедра Автоматизації проектування обчислювальної техніки_____
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти ____другий (магістерський)_____
(рівень вищої освіти)

_____Моделі підвищення надійності IoT-мереж_____
_____
(тема)

Виконав: студент 2 курсу, групи СКСм-22-1

_____Русінов Ю. М._____
(прізвище, ініціали)

Спеціальність 123 Комп'ютерна інженерія_
(код і повна назва спеціальності)

Тип програми __освітньо-професійна_____
(освітньо-професійна або освітньо-наукова)

Освітня програма _____
Спеціалізовані_комп'ютерні системи_____
(повна назва освітньої програми)

Керівник ____проф. Немченко В. П._____
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ __Чумаченко С. В._
(підпис) (прізвище, ініціали)

2023 р.

Харківський національний університет радіоелектроніки

Факультет_____Комп'ютерної інженерії та управління _____

Кафедра_____Автоматизації проектування обчислювальної техніки__

Рівень вищої освіти____другий (магістерський)_____

Спеціальність_____123_ Комп'ютерна інженерія _____
(шифр і назва)

Тип програми _____Освітньо-професійна_____
(освітньо-професійна або освітньо-наукова)

Освітня програма _____Спеціалізовані комп'ютерні системи_____
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«03»___вересня__ 2023 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові ____Русінову Юрію Миколайовичу_____
(прізвище, ім'я, по батькові)

1. Тема роботи Моделі підвищення надійності IoT-мереж_____
_____
_____
затверджена наказом по університету від _03.11.2023 р. № __1288СТ__
2. Термін подання студентом роботи до екзаменаційної комісії 12.04. 2023 р.
3. Вихідні дані до роботи _____
Тип IoT-мереж — стаціонарні та мобільні;_____
Периферія IoT-мережі — Туманне та Хмарне оточення;_____
Канали передачі — IP-мережі (стаціонарні та мобільні)_____
_____
_____
4. Перелік питань, що потрібно опрацювати в роботі _____
Аналіз мережі Інтернету речей_____
Надійність і безпека IoT мереж_____
Методи підвищення надійності IoT мереж_____
Blockchain технології в IoT мережах_____
Аналіз існуючих платформ IoT_____
Створення концепції мережі IoT_____
_____

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) ___15 слайдів_____
_____
_____
_____
_____
_____
_____

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
Спец. частина	проф Немченко В. П.		24.01.24

7. Дата видачі завдання __02.09.2023_____

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	02.09.2023–05.09.2023	
2	Аналіз літератури за темою	06.09.2023–30.09.2023	
3	Суть технічної проблеми	01.10.2023–07.10.2023	
4	Постановка задачі	08.10.2023–10.10.2023	
5	Існуючі методи для вирішення задачі	11.10.2023–31.10.2023	
6	Розробка концепції мережі IoT	01.11.2023–15.11.2023	
7	Оформлення пояснювальної записки	16.11.2023–14.12.2023	
8	Оформлення графічної частини	15.12.2023–05.01.2024	

Студент _____ _____
(підпис)

Керівник роботи _____ _____проф. Немченко В. П._____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи містить 62 сторінок, 1 таблицю, 11 рисунків, 16 джерел за переліком посилань.

INTERNET OF THINGS, МЕРЕЖІ, ПРОТОКОЛИ ЗВ'ЯЗКУ, НАДІЙНІСТЬ, ЗАХИСТ, КОНТРОЛЬ, БЕЗПЕКА, КОНФЕДЕНЦІЙНІСТЬ

У кваліфікаційній роботі розглянуті принципи та готові рішення для підвищення надійності IoT-мереж; різні аспекти, що впливають на функціонування IoT мереж. Проведено дослідження як зовнішніх, так і внутрішніх факторів, які можуть впливати на надійність IoT-мереж. Також розглянуто різні методи та технології, які допомагають підвищити надійність IoT-мереж, такі як дублювання даних, технології забезпечення надійності передачі даних, алгоритми маршрутизації та керування мережею. Створено концепцію мережі IoT.

ABSTRACT

The explanatory note to the qualification work contains 62 pages, 1 table, 11 figures, 16 sources according to the list of references.

INTERNET OF THINGS, NETWORKS, COMMUNICATION PROTOCOLS, RELIABILITY, PROTECTION, CONTROL, SECURITY, CONFIDENTIALITY

The qualification paper examines the principles and ready-made solutions for increasing the reliability of IoT networks were considered. Various aspects affecting the functioning of IoT networks were considered. Both external and internal factors that can affect the reliability of IoT networks were investigated. Various methods and technologies that help improve the reliability of IoT networks, such as data duplication, data transmission reliability technologies, routing algorithms, and network management, were also discussed. The concept of an IoT network has been created.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

IoT — Інтернет речей (англ., Internet of Things);

LoRaWAN — Глобальна мережа дальньої дії (англ., Long Range Wide Area Network);

IWF — Всесвітній форум (англ., International World Forum);

HTTP — Протокол передачі гіпертексту (англ., HyperText Transfer Protocol);

URL — Уніфікований покажчик ресурсу (англ., Uniform Resource Locator);

MQTT — Протокол передачі повідомлень про телеметрію за допомогою черги (англ., Message Queuing Telemetry Transport);

QoS — Якість обслуговування (англ., Quality of Service);

LWT — Остання воля і заповіт (англ., Last Will and Testament);

SMQTT — Захищений протокол передачі повідомлень про телеметрію за допомогою черги (англ., Secure Message Queue Telemetry Transport);

CoAP — Протокол обмеженого застосування (англ., Constrained Application Protocol);

UDP — Протокол користувальницьких датаграм (англ., User Datagram Protocol);

XML — Мова розмітки, що розширюється (англ., eXtensible Markup Language);

JSON — Текстовий формат обміну даними на основі JavaScript (англ., JavaScript Object Notation);

CBOR — Стисле бінарне подання об’єкта (англ., Concise Binary Object Representation);

ECC — Еліптична криптографія (англ., Elliptic Curve Cryptography);

IEEE — Інститут інженерів електротехніки та електроніки (англ., Institute of Electrical and Electronics Engineers);

TCP — Протокол управління передачею (англ., Transmission Control Protocol)

TCP/IP — Протокол управління передачею / протокол Інтернету (англ., Transmission Control Protocol/Internet Protocol);

IoT AEP — Платформи підтримки застосунків Інтернету речей (англ., Internet of Things Application Enablement Platforms)

API — Програмний інтерфейс програми (англ., Application Programming Interface)

ВСТУП

Інтернет речей (Internet of Things, IoT) — технологічна галузь, яка розвивається найшвидшими темпами. Поширення IoT зумовлене різким збільшенням в останній час кількості пристроїв з електронними компонентами, програмним забезпеченням і комунікаційними можливостями, які збирають і передають дані.

У промисловості технології Інтернету речей застосовуються для збільшення терміну експлуатації продуктів, оптимізації поточних витрат та покращення добробуту споживачів. «Речами» в Інтернеті речей є глибоко вбудовані пристрої з такими відмітними особливостями, як вузька смуга пропускання, збір даних з низькою повторюваністю й малий обсяг використовуваних даних. Ці пристрої обмінюються даними один з одним і надають дані через інтерфейси. Проте разом зі зростанням значення та розповсюдженням IoT-мереж постає виклик їх надійності. Оскільки мережа містить безліч різноманітних пристроїв і вузлів, які працюють у різних умовах і середовищах, надійність стає критичним аспектом успішної реалізації IoT-рішень. Непередбачуваність та складність мережевих зв’язків, обмежені ресурси пристроїв та питання забезпечення безпеки створюють проблеми, які потребують глибокого розуміння та застосування відповідних методів для забезпечення надійності IoT-мереж.

У кваліфікаційній роботі пропонується розглянути актуальну тему існуючих для розгортання IoT засобів та визначення моделей підвищення надійності IoT-мереж. Тема актуальна, оскільки ринок насичений, з одного боку, численними об’єктами, які підлягають автоматизації, а з іншого — компаніями, що надають різноманітні рішення для задоволення потреб замовників. Глобальні тенденції на автоматизацію, віддалене управління виробничими активами або житловими приміщеннями та спостереження за ними залишаться актуальними ще протягом тривалого часу.

1 АНАЛІЗ МЕРЕЖІ ІНТЕРНЕТУ РЕЧЕЙ

1.1 Постановка задачі

Об’єктом досліджень є захист інформації в Internet of Things.

Предмет досліджень — метод захисту інформації в мережі Internet of Things.

Мета досліджень — підвищити рівень інформаційної безпеки в IoT-мережах за допомогою існуючих методів.

Задачі:

1) Проаналізувати проблеми безпеки та визначити можливі ризики IoT.

2) Дослідити існуючі методи забезпечення інформаційної безпеки.

1.2 Поняття, концепція та ідея Інтернету речей

Інтернет речей (IoT) є концепцією, у рамках якої Інтернет розвивається не лише як мережа, що забезпечує зв’язок між комп’ютерами та між комп’ютерами і людьми, а й між «розумними» об’єктами чи речами. З розвитком технологій у сфері IoT швидко впроваджуються різноманітні інновації, наслідком чого є трансформація Інтернету в глобальну мережу, у якій усе взаємопов’язане. IoT безперервно розширює свої можливості та стає об’єктом активних досліджень. Традиційний Інтернет якісно змінюється, адаптуючись до нових потреб користувачів. Кількість пристроїв з доступом до Інтернету збільшується, а їх взаємодія вимагає від мережі забезпечення доступу до все більшого обсягу даних. Основна ідея IoT полягає у взаємодії між розумними пристроями за допомогою новітніх технологій, хоча самі ці технології вже існують протягом довгого часу.

Інтернет Речей дозволяє інтегрувати дані, отримані з різних джерел, на різних віртуальних платформах або в мережі Інтернет. Ця ідея була

започаткована ще в 1982 році, коли до Інтернету було підключено автомат для газованої води, однією з функцій якого було інформування про стан напоїв усередині нього. У 1991 році Марк Вайзер передбачив майбутнє IoT, і вже в 1999 році Білл Джой висловив думку про взаємодію між пристроями. У тому ж році Кевін Ештон вперше застосував термін «Інтернет Речей». Основний зміст IoT полягає в обміні важливою інформацією між розрізненими пристроями, які підтримують сучасні технології. Такий обмін дозволяє їм автономно реагувати на різні ситуації.

Концепція Інтернету речей останніми роками стала досить обговорюваною темою. Google-тренди свідчать, що останніми роками люди виявляють постійний інтерес до терміну IoT. Відсоток зацікавленості із часом залишається в середньому на рівні 23 пунктів протягом 2023 року, досягаючи 65 на піку. За даними McKinsey Digital, у 2020 році 127 пристроїв підключалися до Інтернету вперше кожну секунду. А у 2022 році було продано 173 мільйони розумних годинників. Це демонструє, що люди у всьому світі продовжують підключатися до Інтернету речей або для професійного використання (виробники), або для особистого використання (розумні годинники / браслети). Це, своєю чергою, означає, що дедалі більше людей стають потенційними ентузіастами Інтернет-технологій [1].

Інтернет речей все частіше використовують у різних галузях для більш ефективної роботи, покращення обслуговування клієнтів, полегшення процесу прийняття рішень та підвищення цінності бізнесу.

Завдяки Інтернету речей дані можна передавати через мережу, не вимагаючи взаємодії людини з людиною або людини з комп’ютером.

Предметом в Інтернеті речей може бути людина з імплантатом кардіомонітора, сільськогосподарська тварина з біочіпом, автомобіль із вбудованими датчиками, що попереджають водія, наприклад, про низький тиск у шинах, або будь-який інший створений природою або антропогенний об’єкт, якому можна призначити адресу Інтернет-протоколу і який може надсилати дані мережею [2].

1.3 Екосистема IoT

Розглядаючи будь-яку екосистему, ми, по суті, бачимо складну мережу взаємопов’язаних компонентів та середовище, у якому вони існують. Зв’язок цих елементів один з одним відбувається за допомогою різних каналів, таких як енергетичні потоки або цикли, які дуже схожі на цикли поживних речовин у біологічних системах. На відміну від простої системи, екосистема тісно пов’язана з навколишнім середовищем. Тобто система створює єдине, хоч і складне, ціле, натомість екосистема глибоко переплетена із середовищем, у якому вона існує.

Термін «екосистема Інтернету речей» привабливіший, ніж «система Інтернету речей», оскільки цінність пристроїв Інтернету речей визначається середовищем, у якому вони працюють. Основний внесок цих пристроїв у систему (екосистему) — це дані, які вони генерують, дані, пов’язані з умовами навколишнього середовища або зовнішніх явищ, а також внутрішні показники системи. Ці пристрої не лише перебувають у взаємодії не лише з навколишнім середовищем, вони також взаємопов’язані між собою, що забезпечую обмін даними та функціями. У підсумку, кінцевою ланкою, тобто кінцевим користувачем цієї складної мережі даних є людина-оператор, яка використовує її для різних цілей.

Спираючись на ці три компоненти (довкілля, дані, люди), можна сформулювати визначення екосистеми Інтернету речей. Це мережа взаємопов’язаних пристроїв, що існує в деякому середовищі, збирає дані та передає їх людям, які обробляють та аналізують ці дані із використанням сучасних технологій для досягнення певної мети, наприклад, створення розумного будинку.

Оскільки різні групи людей мають різні цілі, вони створюють різні програми Інтернету речей, наслідком чого є численні екосистеми Інтернету речей, розроблені за допомогою різного програмного забезпечення. Так, екосистемою може бути як проста мережа з 20 підключеними пристроями,

наприклад «розумний дім», так і багаторівнева структура зі складною й широкою мережею пристроїв, яка потребує складної платформи для керування всіма рівнями.

Якщо розкласти найскладнішу екосистему Інтернету речей із середнім шаром на складові блоки, ми отримаємо таку схему: пристрої Інтернету речей збирають дані та безпечно передають їх через мережу на шлюз, підключений до Інтернету, шлюз стискає інформацію та передає її на хмару для подальшого аналізу, результати якого потім будуть відображені в додатку, де користувачі й отримують необхідну інформацію.

Отже, маємо 7 компонентів екосистеми Інтернету речей (рис. 1.1):

– IoT-пристрій;

– безпека;

– мережа;

– шлюз;

– хмара;

– додаток;

– користувачі.

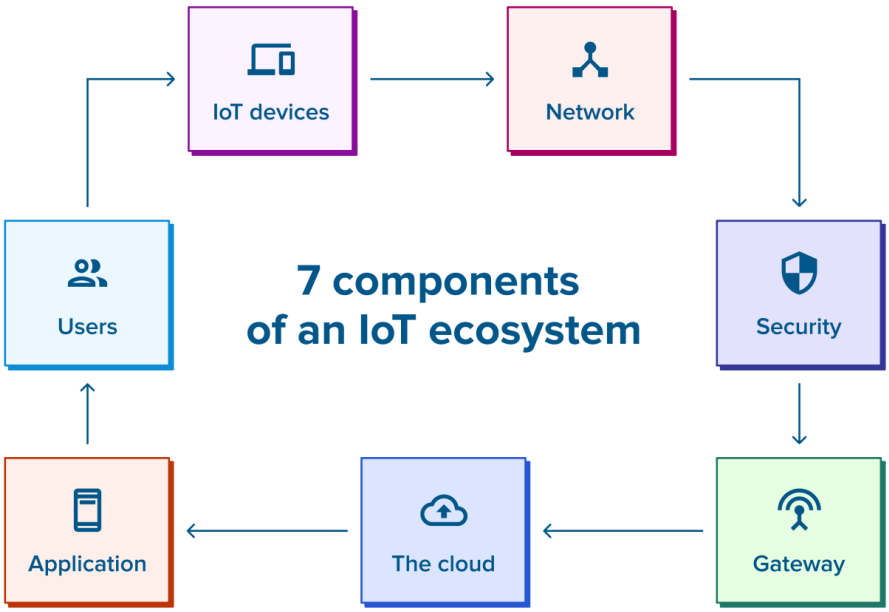

Рисунок 1.1 — 7 компонентів екосистеми IoT

1.4 Архітектура IoT

Архітектури IoT розрізняються за складністю та кількістю архітектурних шарів залежно від конкретного бізнес-завдання, проте комітет з архітектури Всесвітнього форуму IoT, до складу якого входять лідери індустрії, зокрема IBM, Intel і Cisco, у жовтні 2014 року опублікував еталонну модель IoT, що має 7 шарів: фізичні пристрої та контролери; зв'язок; туманні обчислення; накопичення даних; абстракція даних; додатки; взаємодія та процеси [3]. Запропонована семирівнева модель зображена на рис. 1.2.

Рисунок 1.2 — Еталонна модель Всесвітнього форуму IoT

Розглянемо більш детально кожний рівень.

Фізичні пристрої і контролери: цей шар включає фізичні пристрої — датчики, актуатори, розумні пристрої, мікроконтролери та інші засоби збору

даних або керування пристроями. Вони забезпечують сприйняття та моніторинг фізичних параметрів, а також керування актуаторами, що дозволяє реалізувати функціональні можливості IoT-пристроїв.

Зв'язок: шар зв'язку відповідає за передачу даних між пристроями IoT та іншими компонентами системи. Це може відбуватися з використанням бездротових технологій, таких як Wi-Fi, Bluetooth, Zigbee, або провідних засобів передачі даних, таких як Ethernet або LoRaWAN. Шар зв'язку забезпечує надійне та безперервне з'єднання між пристроями та мережею.

Туманні обчислення: шар туманних обчислень (Fog Computing) дозволяє обробляти та аналізувати дані на ближчих до пристроїв ресурсах, таких як шлюзи або локальні сервери, замість передачі всіх даних до центральних хмарних серверів. Це приводить до зниження затримки в обробці даних, підвищення безпеки та забезпечення швидкої реакції на події в реальному часі.

Накопичення даних: шар накопичення даних відповідає за збереження та управління великим обсягом даних, зібраних від пристроїв IoT. Це можуть бути бази даних, системи зберігання або хмарні платформи, які забезпечують надійне зберігання, індексацію та доступ до даних.

Абстракція даних: шар абстракції даних дозволяє створювати високорівневі подання та моделі для зрозумілого аналізу та використання даних. Він включає в себе методи агрегації, фільтрації, перетворення та інтерпретації даних, щоб спростити їх розуміння та використання в різних додатках та сервісах.

Додатки: шар додатків охоплює різноманітні програмні рішення, які використовують дані з пристроїв IoT для реалізації конкретних функціональних можливостей. Це можуть бути моніторингові системи, системи керування, аналітичні інструменти, системи безпеки тощо. Додатки забезпечують інтеграцію даних та доступ користувачам до результатів аналізу та управління пристроями IoT.

Співробітництво та процеси: шар взаємодії та процесів відповідає за

встановлення способів взаємодії між різними компонентами системи IoT, а також за організацію бізнес-процесів та логіки роботи системи. Він включає в себе протоколи комунікації, стандарти, механізми керування та автоматизації, що забезпечують синхронізацію та координацію всіх компонентів IoT системи.

Документальний опис моделі IWF, опублікований Cisco, вказує, що розроблена модель відрізняється такими характеристиками:

спрощує: допомагає розбити складні системи на частини так, щоб кожна із цих частин стала більш зрозумілою;

прояснює: надає додаткові відомості для точної ідентифікації рівнів IoT і вироблення загальної термінології;

ідентифікує: ідентифікує аспекти, у яких ті чи інші типи обробки оптимізовані в різних частинах системи;

стандартизує: є перший крок до того, щоб постачальники могли створювати продукти IoT, здатні взаємодіяти один з одним;

організовує: робить IoT реальним і доступним, а не просто абстрактною концепцією.

Приклад IoT-мережі наведений на рис. 1.3.

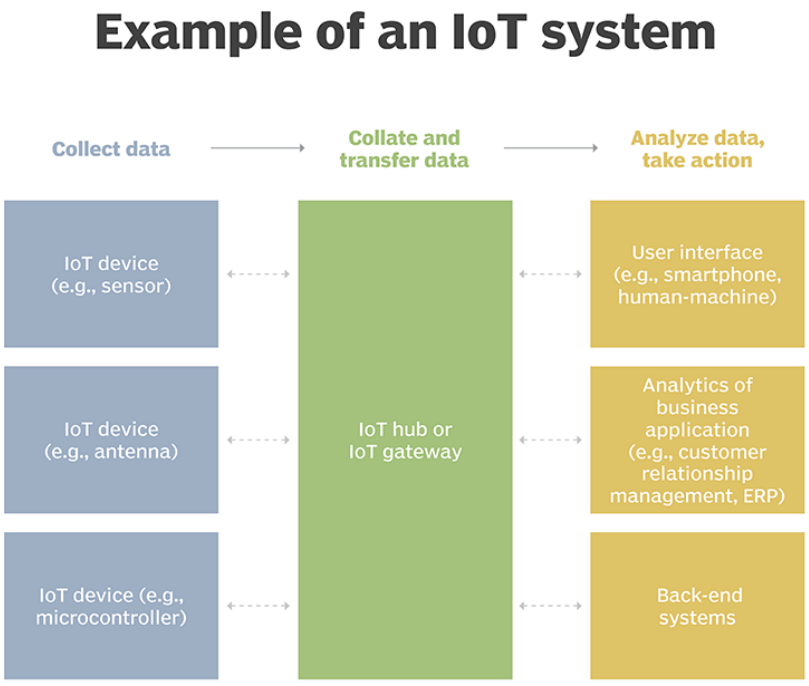

Рисунок 1.3 — Приклад IoT-мережі

1.5 Основні протоколи зв’язку в IoT-мережах

В IoT-мережах використовуються різні протоколи зв’язку, які забезпечують передачу даних між пристроями IoT, шлюзами та хмарними серверами. Залежно від конкретних вимог та сценаріїв застосування протоколи зв’язку в IoT-мережах можуть використовуватися окремо або в комбінаціях. Вибір протоколу зв’язку залежить від функціональності, пропускної здатності, енергоефективності та вимог до безпеки, висунутих до системи IoT. Розглянемо основні протоколи зв’язку в IoT-мережах.

1.5.1 Протокол HTTP

HTTP (HyperText Transfer Protocol) був винайдений як компонент Всесвітньої павутини для передачі документів. Він найбільш знайомий користувачам як одна з основних технологій, що дозволяє працювати веб-браузерам. Сервери містять ресурси, ідентифіковані за URL-адресами.

У середовищі IoT загальне використання HTTP полягає в тому, щоб дозволити пристроям здійснювати POST до ресурсу, який відображує стан пристрою в службі IoT (рис. 1.4).

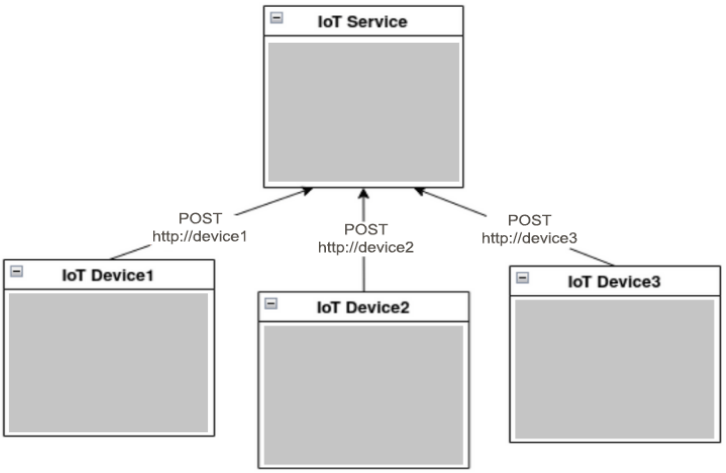

Рисунок 1.4 — HTTP протокол в IoT

Основною перевагою HTTP для використання в IoT є те, що він добре знайомий розробникам, багато з яких запровадили ті чи інші веб-рішення. Наслідком цього є доступність клієнтських бібліотек і серверів [4].

1.5.2 Протокол MQTT

MQTT — це міжмашинний протокол підключення до Інтернету речей. Це легкий транспортний протокол для обміну повідомленнями за типом публікація / підписка. Його використання доцільне для підключення до віддаленого місця у випадках, коли треба враховувати пропускну здатністю. Цей протокол корисний у різних ситуаціях, зокрема в постійному середовищі, наприклад, для зв’язку між машинами та контекстами Інтернету речей. Це система публікації та підписки, яка дозволяє і публікувати, і отримувати повідомлення як клієнт. Завдяки цьому полегшується зв’язок між кількома пристроями. Це простий протокол обміну повідомленнями,

розроблений для обмежених пристроїв із низькою пропускною здатністю, тому він є ідеальним рішенням для програм Інтернету речей.

MQTT має деякі унікальні функції, які навряд чи можна знайти в інших протоколах. Це міжмашинний протокол, тобто він забезпечує зв'язок між пристроями. Він розроблений як простий і легкий протокол обміну повідомленнями, який використовує систему публікації / підписки для обміну інформацією між клієнтом і сервером. Для цього не потрібно, щоб і клієнт, і сервер встановлювали з'єднання одночасно. Він забезпечує швидшу передачу даних. Це протокол обміну повідомленнями в реальному часі. Це дозволяє клієнтам підписатися на вузький вибір тем, щоб вони могли отримати інформацію, яку вони шукають.

Архітектура MQTT складається з клієнтів MQTT (видавців та підписників), які взаємодіють із брокером MQTT, який діє як брокер повідомлень (рис. 1.5). Теми використовуються для адресації повідомлень, а MQTT підтримує різні рівні якості обслуговування (QoS) для доставки повідомлень. MQTT також підтримує збережені повідомлення та функцію «Остання воля і заповіт» (LWT), що підвищує його гнучкість та надійність як протокол обміну повідомленнями для Інтернету речей та інших програм [5].

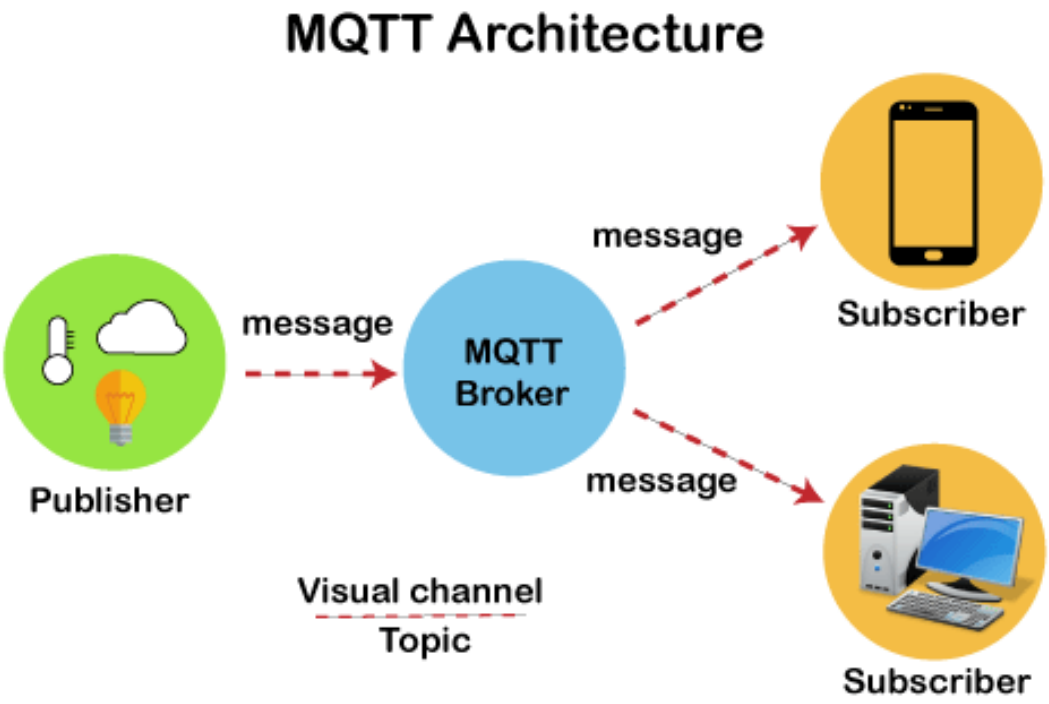

Рисунок 1.5 — Архітектура MQTT

Клієнти MQTT — це кінцеві точки, які взаємодіють одна з одною з використанням протоколу MQTT. Клієнтом може бути будь-який пристрій або програма, яка реалізує протокол MQTT і може виступати як видавцем, так і підписником. Клієнти MQTT можуть бути класифіковані на такі два типи.

1) Видавці MQTT — це клієнти, які надсилають брокеру MQTT повідомлення з певною темою. Вони публікують повідомлення в розділах, які є рядками, що представляють канал повідомлень або механізм логічної адресації.

2) Підписники MQTT — це клієнти, які отримують від брокера MQTT повідомлення на основі своєї підписки на одну або кілька тем. Підписники вказують теми, які їх цікавлять, і брокер MQTT пересилає їм повідомлення, які відповідають цим темам.

Брокер MQTT — основний компонент архітектури MQTT. Він діє як посередник повідомлень — отримує повідомлення від видавців та пересилає їх підписникам (на основі їхніх підписок). Функції брокера MQTT: він

відповідає за управління темами, обробку клієнтських підключень, управління рівнями якості обслуговування (QoS) та забезпечення надійної доставки повідомлень. Брокер MQTT також може зберігати отримані раніше повідомлення та доставляти їх новим підписникам після їх підключення.

Теми — це позначки, пов'язані з каналом повідомлення або механізмом логічної адресації MQTT. Видавці надсилають повідомлення, позначаючи теми, а підписники вказують теми, які їм цікаві. Теми MQTT можуть бути організовані ієрархічно, завдяки цьому забезпечується гнучка і масштабована система обміну повідомленнями. Теми можуть мати один або декілька рівнів, що розділяються косою рискою (/), а також можуть містити знаки для встановлення гнучких підписок.

Рівень якості обслуговування (QoS): для доставки повідомлень MQTT підтримує різні рівні якості обслуговування (QoS). Для кожного повідомлення видавці та підписники можуть вказати бажаний рівень QoS [6]. MQTT підтримує три рівні QoS:

1) QoS 0 (не більше одного разу): повідомлення надсилаються один раз без будь-якої гарантії доставки або підтвердження.

2) QoS 1 (принаймні один раз): повідомлення будуть гарантовано доставлені принаймні один раз, але можуть бути доставлені й кілька разів, якщо відбувся збій або виникли проблеми з мережею.

3) QoS 2 (одноразово): повідомлення гарантовано доставляються рівно один раз, це забезпечує найвищий рівень надійності.

Збережені повідомлення: MQTT дозволяє зберігати повідомлення, які зберігаються на брокері та доставляються новим підписникам під час їх підключення. Збережені повідомлення можна використовувати для збереження останньої відомої інформації про стан або конфігурацію, дозволяючи підписникам отримувати найсвіжішу інформацію, навіть якщо вони не були підключені до мережі, коли повідомлення було опубліковано спочатку.

Остання воля та заповіт (LWT): MQTT підтримує функцію «Остання

воля та заповіт» (LWT), завдяки якій пристрій може вказати повідомлення, яке буде опубліковано брокером, якщо цей пристрій несподівано відключиться від мережі. Повідомлення LWT зазвичай використовуються для сповіщення інших про несподіване вимкнення або збій в роботі пристрою.

SMQTT (Secure Message Queue Telemetry Transport) — це розширення протоколу MQTT, в якому використовується шифрування на основі змінених атрибутів шифрування. Основною перевагою цього шифрування є те, що воно має функцію широкого шифрування — одне повідомлення шифрується і додається на кілька інших вузлів [7]. Процес передачі та отримання повідомлень складається з чотирьох основних етапів:

1) налаштування: на цьому етапі видавці й підписники реєструються у брокера та отримують секретний майстер-ключ;

2) шифрування: коли дані публікуються у брокера, вони шифруються брокером;

3) публікація: брокер публікує зашифроване повідомлення підписникам;

4) розшифрування: отримане повідомлення розшифровується підписниками з тим самим майстер-ключем.

SMQTT застосовується лише для покращення функцій безпеки MQTT.

1.5.3 Протокол CoAP

CoAP (Constrained Application Protocol) є полегшеним протоколом, він призначений для роботи з пристроями з низьким енергоспоживанням та обмеженими мережами — такими, як ті, що зустрічаються в пристроях IoT (Internet of Things). Прогнозується, що він стане простішою та ефективнішою альтернативою HTTP для обмежених пристроїв — з обмеженою обчислювальною потужністю, пам’яттю та часом автономної роботи. CoAP побудований поверх UDP (User Datagram Protocol), тобто використовує UDP

як базовий транспортний протокол. Він надає набір методів для виявлення ресурсів, виконання операцій над ресурсами та забезпечує спостереження, а також підтримку асинхронного зв'язку та кешування. Завдяки низьким накладним витратам та простоті CoAP набуває все більшої популярності в додатках IoT і, як очікується, відіграє важливу роль у майбутньому Інтернету речей [8].

Архітектура CoAP складається із чотирьох основних компонентів.

• Клієнт CoAP — це будь-який пристрій, який ініціює запит до сервера CoAP для вилучення, створення, оновлення або видалення ресурсу. Він може також спостерігати за ресурсом, щоб отримувати сповіщення при його зміні.

• Сервер CoAP — це пристрій, який надає ресурси для доступу клієнтів CoAP. Сервер обробляє вхідні запити і надсилає відповіді клієнту. Він може також надсилати сповіщення клієнтам, які спостерігають за ресурсом.

• Проксі-сервер CoAP — це проміжний пристрій між клієнтом CoAP і сервером CoAP, який може використовуватися для фільтрації або зміни запитів та відповідей, а також для кешування відповідей з метою зменшення мережевого трафіку.

• Ресурс CoAP — це будь-яка частина даних або функціональних можливостей, до яких можна отримати доступ через CoAP. Він ідентифікується за допомогою URI і може мати одне або кілька представлень у різних форматах, таких як XML, JSON або CBOR. Ресурси можуть бути статичними, наприклад, показання датчика, або динамічними, наприклад, функція управління.

Архітектура CoAP також включає кілька інших ключових концепцій.

• Повідомлення: CoAP використовує модель зв'язку на основі повідомлень, де кожен запит та відповідь інкапсулюються в повідомленні CoAP. Повідомлення CoAP невеликі, їх максимальний розмір складає 1024 байти.

• Опції: повідомлення CoAP можуть містити параметри, які надають додаткову інформацію про запит або відповідь. Приклади опцій включають

формат вмісту, спостереження, ETag та максимальний вік.

• Спостерігаючі ресурси: клієнти CoAP можуть спостерігати за ресурсами, надіславши спеціальний запит на спостереження. Потім сервер надсилатиме повідомлення клієнту щоразу, коли ресурс змінюється.

• Передавання по блоках: CoAP підтримує передачу даних по блоках, коли великі ресурси можна розділити на дрібніші блоки і передати в послідовності запитів і відповідей. Це дозволяє пристроям з обмеженою пам’яттю та пропускною здатністю отримувати доступ до великих ресурсів.

1.5.4 Протокол Zigbee

ZigBee — це бездротовий протокол, який використовує для підключення пристроїв малопотужні радіосигнали. Він заснований на стандарті IEEE 802.15.4, який визначає фізичний рівень та рівень керування доступом до середовища для низькошвидкісних бездротових персональних мереж (LR-WPAN). Ці мережі призначені для малопотужних пристроїв з обмеженими обчислювальними можливостями та пам’яттю [9].

ZigBee — це технологія мереж, тобто вона може створювати мережу пристроїв, у якій кожен може виступати як маршрутизатор для інших пристроїв, що входять до цієї мережі. Це дозволяє ZigBee забезпечувати надійний зв’язок навіть у складних умовах. Причому в одній мережі може підтримуватися функціонування до 65 000 пристроїв. Радіус дії ZigBee становить до 100 метрів.

Протокол ZigBee широко використовується в пристроях IoT, наприклад, у пристроях розумного будинку, розумного освітлення та промислової автоматизації. Протокол ZigBee розрахований на роботу з пристроями, що мають низьке енергоспоживання, тобто є ідеальним для таких, що працюють від батарей.

ZigBee працює, створюючи пористу мережу пристроїв, зв’язок між якими відбувається за допомогою малопотужних радіосигналів. Кожен

пристрій у мережі може діяти як маршрутизатор, тобто приймати та пересилати повідомлення іншим пристроям у мережі. Завдяки цьому протокол створює надлишкові шляхи зв'язку між пристроями, що підвищує надійність мережі. Крім цього, ZigBee використовує унікальну схему адресації, яка дозволяє однозначно ідентифікувати пристрої мережі і захистити мережу від несанкціонованого доступу. Тобто ця схема адресації забезпечує функції безпеки протоколу, такі як шифрування та автентифікація.

На рис. 1.4. показані різні топології структури мережі ZigBee.

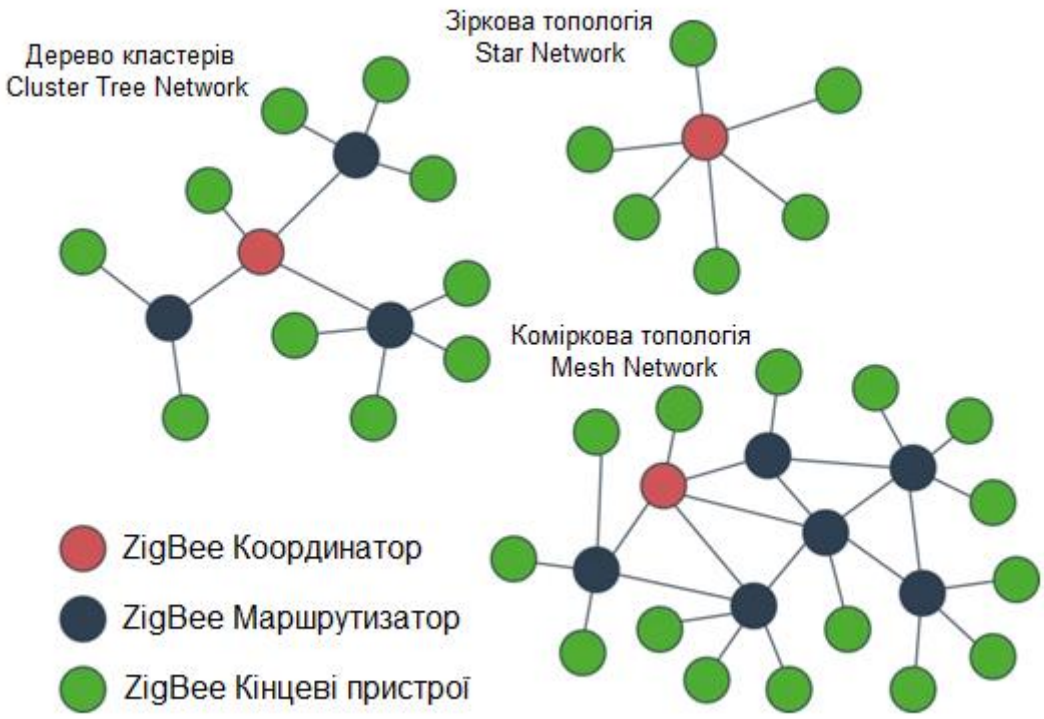

Рисунок 1.4 — Структура мережі Zigbee

Координатор — вузол, який є організатором мережі. Він обирає політику безпеки мережі, контролює (дозволяє або забороняє) підключення нових пристроїв до мережі, а також, у випадку перешкод радіозв'язку, ініціює процес переведення на інший частотний канал усіх пристроїв мережі.

Маршрутизатор (роутер) — вузол, який має стаціонарне живлення, отже, може постійно працювати в мережі. Ці вузли відповідають за

маршрутизацію мережевого трафіку. Координатор також є роутером. Для прокладання оптимального маршруту та пошуку нового, якщо один із пристроїв мережі вийшов з ладу, роутери використовують спеціальні таблиці маршрутизації. Роутерами в мережі ZigBee можуть бути, наприклад, блоки керування освітлювальними приладами, розумні розетки або будь-які інші пристрої, що мають підключення до електромережі.

Кінцевий пристрій — це пристрій, який підключається до мережі через «батьківський» вузол (роутер або координатор) і не бере участі в маршрутизації трафіку. Для нього все спілкування з мережею обмежується передачею пакетів на «батьківський» вузол чи зчитуванням даних, що надійшли з нього. «Батьком» для таких пристроїв може бути будь-який роутер або координатор. Кінцеві пристрої перебувають у сплячому режимі і надсилають керуючі або інформаційні повідомлення тільки внаслідок певної події. Це дозволяє їм довго зберігати енергію вбудованого джерела живлення.

Таблиця 1.1 – Порівняльна характеристика основних мережевих технологій

Технологія	ZigBee (IEEE 802.1.5.4)	Wi-Fi (IEEE 802.11 b)	Bluetooth (BLE) (IEEE 802.15.1)
Частотний діапазон, ГГц	2,4–2,483	2,4–2,483	2,4–2,483
Пропускна здатність, Кбіт/сек	250	11000	723,1
Розмір стеку протоколу, Кбайт	32–64	>1000	>250

Продовження таблиці 1.1

Технологія	ZigBee (IEEE 802.1.5.4)	Wi-Fi (IEEE 802.11 b)	Bluetooth (BLE) (IEEE 802.15.1)
Час неперервної автономної роботи від акумулятора, дні	100–1000	0,5–5	1–10
Максимальна кількість вузлів у мережі	65536	10	7
Середнє значення діапазону дії, м	10–100	20–300	10–100

1.6. Аналіз елементів архітектури Інтернету речей

Для визначення потрібного обладнання та кращого розуміння роботи Інтернету речей здійснимо дослідження його архітектури. Інтернет речей привертає велику увагу завдяки своєму величезному потенціалу. Хоча існує багато оптимістичних очікувань щодо того, як IoT може докорінно змінити наше життя, реальний прогрес може бути повільнішим, ніж очікується. Головна причина полягає в складності та різноманітності систем IoT, що гальмує їх розвиток.

Для побудови мережі Інтернету речей використовуються такі пристрої.

– Вбудовані системи. До їх складу входять як фізичне обладнання, так і програмні рішення. Ці системи контролюють конкретні завдання в рамках загальної системи. Вони базуються на мікропроцесорах чи мікроконтролерах.

– Розумні системи. Ці пристрої здатні проводити розрахунки та зазвичай базуються на мікроконтролерах.

– Мікроконтролери (MCU). Це компактні комп’ютерні системи, інтегровані в мікросхеми. Вони обладнані процесором, оперативною та постійною пам’яттю та призначені для виконання простих завдань, але їхні

можливості обмежені порівняно з мікропроцесорами.

– Мікропроцесори (MPU). Це мікроелектронні програмовані пристрої, процесорні функції в яких централізовані на одній або декількох мікросхемах. Незважаючи на необхідність додаткових периферійних пристроїв для роботи мікропроцесорів, вони є економічно вигідними через низьку вартість основних обчислювальних компонентів.

– Обчислювально-незалежні пристрої. Ці компоненти здійснюють лише з'єднання та передачу даних, без можливості обробки інформації.

– Конвертори. Це пристрої, що перетворюють одну форму енергії на іншу. У межах Інтернету речей це — датчики та актуатори, які надсилають інформацію на основі взаємодії об'єктів з оточуючим середовищем.

1.6.1. Ключові елементи архітектури Інтернету речей

Незалежно від специфікації кожної системи IoT, основні принципи архітектури Інтернету речей та загальна логіка обробки даних майже збігаються.

Перше за все, система IoT складається з речей, тобто об'єктів, підключених до Інтернету, які за допомогою вбудованих датчиків і виконавчих механізмів можуть відчувати навколишнє середовище та збирати відповідну інформацію, яка потім передається шлюзам IoT.

Наступний шар включає системи збору даних IoT та шлюзи. Ці компоненти обробляють великі масиви даних — переводять їх у структуровані цифрові формати, виконують первинну фільтрацію та опрацювання для подальшого аналізу.

Третій рівень складається з кінцевих пристроїв, які забезпечують додаткову обробку та розширений аналіз даних. На цьому рівні можливе використання засобів візуалізації та алгоритмів машинного навчання.

Після обробки на цьому рівні дані передаються до дата-центрів (центрів обробки даних), які можуть бути розміщені як у хмарі, так і

локально. Це «серце» системи, де дані зберігаються, опрацьовуються та аналізуються для виокремлення корисної інформації [10]. Усі чотири рівні архітектури IoT детально представлені на рис. 1.5.

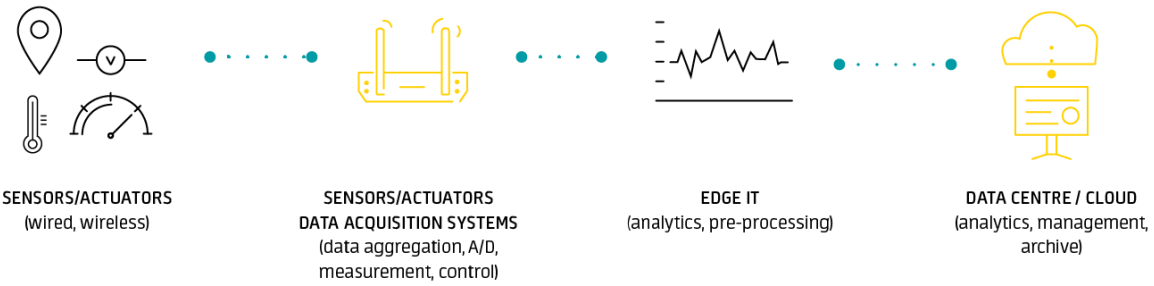

Рисунок 1.5 — Основні складові архітектури IoT

1.6.2. Сенсори та контролери

Підключені до кожної системи IoT пристрої є її основою, оскільки відповідають за отримання даних, які і є суттю Інтернету речей. Щоб зафіксувати фізичні показники зовнішнього середовища або характеристики самих пристроїв, ці пристрої оснащені сенсорами, показаними на рис. 1.6. Ці сенсори можуть бути або інтегровані безпосередньо в пристрої, або виконані як окремі модулі для вимірювання та збору телеметричних даних.

Рисунок 1.6 — Апаратні складові IoT

Окрім сенсорів, іншими важливими елементами пристроїв IoT є виконавчі механізми. Вони взаємодіють із датчиками та залежно від отриманих від них даних виконують конкретні дії. Усе це відбувається автоматично, без участі людини.

Також важливо зазначити, що пристрої в мережі IoT можуть комунікувати не лише з основними системами або шлюзами, але й обмінюватися даними між собою, координуючи свої дії в реальному часі. Однак це може створити труднощі для пристроїв з обмеженими ресурсами та живленням від батареї, оскільки така комунікація потребує обчислювальних ресурсів і енергії. Тому важливо оптимізувати архітектуру IoT з використанням легких і безпечних комунікаційних протоколів.

1.6.3. Шлюзи та агрегація даних

Незважаючи на те, що на певних пристроях цей рівень функціонує неподалік від датчиків та виконавчих компонентів, необхідно розглядати його як виокремлений шар у структурі IoT. Це пов’язано з його ключовою роллю у зборі, фільтруванні та передачі даних до кінцевих систем і хмарних платформ. З огляду на величезний потік даних, зумовлений наявністю мільйонів пристроїв, основна увага має приділятися збору, обробці та передачі інформації.

Виступаючи зв’язуючою ланкою між IoT-пристроями та хмарними ресурсами, шлюзи та системи агрегації даних стають критичними точками зв’язку, об’єднуючи різні рівні системи (рис. 1.7). Розташовані на межах, шлюзи виконують первинну обробку даних, одержаних від датчиків, адаптуючи їх для подальшого використання в системі. Вони також можуть вибірково відсіювати та оптимізувати дані, спрощуючи інтеграцію, тобто зменшуючи кількість інформації, яка передається в хмару, що сприяє

ефективності та швидкодії системи.

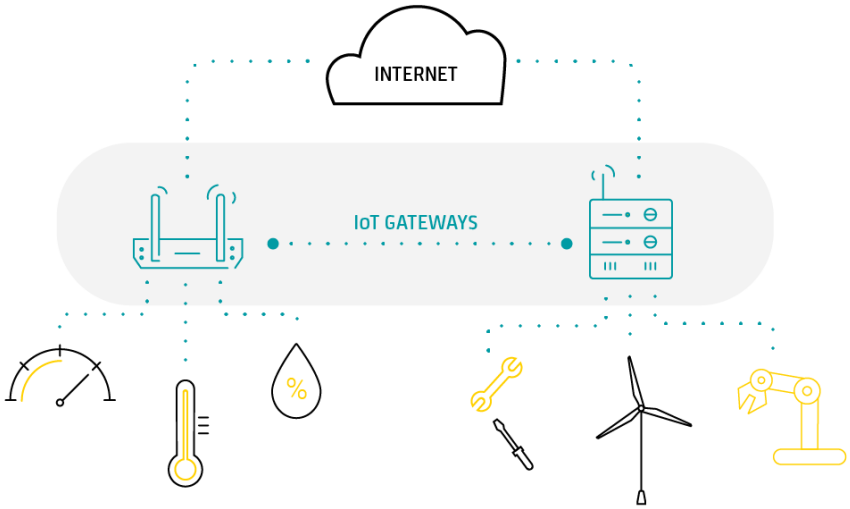

Рисунок 1.7 — Шлюзи та системи агрегації даних

Безпека є ще однією важливою функцією шлюзів. Вони контролюють потік даних у двох напрямках і зменшують ризик витоку даних та потенційних атак на IoT-пристрої за допомогою відповідних шифрувальних та захисних механізмів.

1.6.4 Дата-центр

Якщо розглядати датчики як нейрони, а шлюз як основу IoT, то хмару можна порівняти з мозком Інтернету речей. На відміну від периферії (кінцевих систем), дата-центр, або хмарна платформа, призначена для збереження, опрацювання та детального аналізу величезних масивів даних (рис. 1.8). Вони використовують високоефективні засоби аналітики та машинного навчання, які перевищують можливості кінцевих систем.

Рисунок 1.8 — Дата-центр

Ураховуючи активний розвиток хмарних технологій протягом останніх років (особливо в контексті промислового IoT), хмарні обчислення стають основою підвищення продуктивності, зменшення непередбачуваних затримок, оптимізації споживання енергії та інших бізнес-переваг.

Якщо хмарні сервіси оснащені відповідними програмними рішеннями, вони можуть надавати інструменти для бізнес-аналітики та інтерактивних засобів візуалізації. Завдяки ним користувачі можуть спілкуватися із системою, керувати нею та приймати обґрунтовані рішення на основі звітів, інтерактивних панелей та даних у реальному часі.

Одним із ключових викликів, який постав перед IoT (разом із питаннями безпеки), є фрагментація. Вона виникає внаслідок того, що спектр пристроїв та систем, які Інтернет речей намагається об’єднати, є дуже широким. Щоб запустити коректно працюючу систему IoT, необхідно інтегрувати різні ресурси, обладнання, програмне забезпечення в єдиний механізм. Це вимагає продуманої архітектури IoT.

2 НАДІЙНІСТЬ І БЕЗПЕКА ІОТ-МЕРЕЖ

2.1 Зовнішні фактори

Зовнішні фактори можуть мати значний вплив на надійність IoT-мереж. Тому важливо враховувати їх в ході проєктування, розгортання мережі та управління нею. Заходи, призначені для підвищення надійності IoT-мереж, можуть включати використання антен та підсилювачів сигналу, засобів захисту від погодних умов, вибір оптимальних місць розташування пристроїв, а також керування використовуваним радіочастотним спектром. Розглянемо деякі зовнішні фактори, вплив яких варто враховувати.

2.1.1 Вплив зовнішнього середовища

Зовнішнє середовище, у тому числі кліматичні умови, може впливати на якість зв’язку та роботу пристроїв IoT. Наприклад, сильні дощі, вітер, сніг, температурні зміни можуть вплинути на бездротовий зв’язок та призвести до зниження надійності мережі. Фізичні перешкоди, такі як стіни, будівлі або географічні об’єкти, можуть заважати передачі сигналу та погіршувати якість зв’язку.

2.1.2 Електромагнітні перешкоди

Лінії електричних передач, металеві конструкції, електронні пристрої, які не входять до даної мережі, можуть виступати як електромагнітні перешкоди та впливати на сигнали IoT-пристроїв внаслідок інтерференції. Наслідком цього може стати переривання зв’язку або спотворення даних, що передаються. Для забезпечення надійності мережі IoT необхідно проводити

аналіз електромагнітного спектра та уникати впливу джерел електромагнітних сигналів та інших пристроїв.

2.1.3 Радіочастотні спектри

Оскільки багато IoT-пристроїв працюють у бездротовому режимі, на надійність зв'язку можуть впливати конфлікти радіочастот. Обмежена кількість доступних радіочастотних діапазонів може призвести до перенасиченості етеру та створення взаємних перешкод пристроями, що використовують одну й ту саму радіочастоту. Наслідком може стати зниження якості зв'язку та зменшення пропускної здатності мережі. Ефективне керування радіочастотним спектром та вибір оптимальних частот можуть допомогти уникнути цих проблем.

2.2 Внутрішні фактори

Внутрішні фактори впливають на надійність IoT-мереж із точки зору їх внутрішньої архітектури та характеристик пристроїв.

2.2.1 Енергоспоживання та живлення

Управління живленням — дуже широка тема, вона стосується як програмного, так і апаратного забезпечення. При розгортанні IoT-мережі дуже важливо розуміти роль ефективного управління енергоспоживанням віддалених пристроїв і пристроїв довготривалої експлуатації та володіти прийомами управління ними. Розраховуючи бюджет енергоспоживання, проєктувальник має враховувати такі фактори:

– активна потужність датчика; частота збору даних;

– споживана потужність бездротового радіозв'язку, необхідна для забезпечення покриття заданої площі;

– частота зв’язку; потужність мікроконтролера або мікропроцесора залежно від тактової частоти ядра;

– потужність пасивної складової; втрати енергії внаслідок витоків або неефективності живлення;

– резервування електроенергії для живлення приводів і двигунів.

Бюджет відображає сумарне енергоспоживання всіх споживачів, що забезпечуються джерелом живлення (батареєю). Із плином часу батарея розряджається. При цьому батарея втрачає енергоємність, а напруга на ній падає криволінійно. Це створює проблеми для систем бездротового зв’язку. Якщо акумулятор розрядиться до деякої критичної напруги, мікропроцесор, не маючи живлення, просто відключиться.

Отже, ключовим фактором для забезпечення тривалої роботи мережі є ефективне управління енергоспоживанням. Воно включає в себе оптимізацію споживання енергії пристроями, використання режимів сну та глибокого сну для економії енергії, а також застосування енергоефективних компонентів.

2.2.2 Обмежені ресурси вузлів IoT

Багато пристроїв IoT мають обмежені ресурси: обчислювальні, мережеві та ресурси пам’яті. Це може вплинути на надійність мережі. Важливо розробляти оптимізовані протоколи та алгоритми, які враховують обмежені ресурси пристроїв. Крім того, вирішити проблему обмежених ресурсів може допомогти використання розподіленого обчислення та облікових систем.

2.2.3 Проблема конфіденційності

Конфіденційність і повага до права на конфіденційність мають ключове значення для завоювання довіри користувача по відношенню до Інтернету, підключеним пристроям і пов’язаним із ними послугами. Інтернет

речей став причиною того, що полеміка про конфіденційність вийшла на новий рівень, оскільки його широке застосування може докорінно змінити методи збору, аналізу, застосування й захисту особистих даних. Наприклад, IoT підсилює занепокоєння внаслідок ймовірності додаткового спостереження і стеження, неможливості відмовитися від надання деяких даних і можливості об'єднання декількох потоків даних IoT для створення докладних цифрових описів користувачів. Це серйозні проблеми, але вони не є нерозв'язними.

Дотримання права на недоторканність приватного життя і забезпечення конфіденційності є невід'ємною частиною вирішення проблеми довіри до Інтернету. Питання забезпечення прав і задоволення очікувань користувачів іноді зводяться до проблеми етичної обробки даних, дотримання конфіденційності й сумлінного використання даних. Інтернет речей здатний поставити під сумнів традиційні очікування дотримання прав приватного життя. Вирішити проблеми конфіденційності, що виникли з появою Інтернету речей, дуже важливо, оскільки вони стосуються основних прав людини і здатності нашого суспільства довіряти Інтернету й підключеним до нього пристроям. Інтернет речей часто уявляють як масштабну мережу сенсорних пристроїв, які збирають дані про зовнішнє оточення і нерідко — про самих людей. Звісно, ці дані можуть бути корисними для власників пристроїв, але дуже часто вони мають інтерес і для виробників та постачальників пристроїв. Збір і використання IoT-даних перетворюється на справжню проблему конфіденційності, коли уявлення людей, які перебувають під наглядом IoT-пристроїв, про масштаб і шляхи використання даних відрізняються від міркувань збирача даних.

У цілому проблеми конфіденційності посилюються тим, що Інтернет речей значно розширює можливості й доступність стеження і спостереження. Удосконалення характеристик IoT-пристроїв і методів їх використання спрямовує дискусії щодо проблем конфіденційності в нове русло, оскільки відбуваються серйозні зміни методів збору, аналізу, використання і захисту

персональних даних.

2.2.4 Проблема безпеки

Забезпечення безпеки, надійності, стійкості й стабільності додатків і послуг Інтернету має критично важливе значення для створення довіри до використання Інтернету.

Безпека IoT пов’язана, в основному, з довірою користувачів до середовища. Якщо користувачі не вірять у захищеність підключених пристроїв і отриманої інформації від неприпустимого використання, цей брак довіри призводить до відмови від використання Інтернету. Цей фактор глобально впливає на електронну комерцію, технічні інновації, свободу висловлювань і практично всі інші аспекти онлайн-діяльності. Забезпечення безпеки продуктів і послуг IoT має бути головним пріоритетом у цій галузі.

У міру постійного збільшення числа підключених до Інтернету пристроїв виникають нові потенційно вразливі місця. Недостатньо захищені пристрої можуть слугувати точками доступу для кібератак, дозволяючи зловмисникам перепрограмувати пристрій або викликати його несправність. Пристрої недосконалої конструкції можуть бути джерелом ризику щодо викрадення даних користувачів унаслідок недостатнього захисту потоків даних. Несправні або дефектні пристрої також можуть створювати вразливі точки. Для поширених недорогих пристроїв невеликого розміру ці проблеми стоять так само гостро або навіть гостріше, ніж для комп’ютерів, які традиційно використовувалися для підключення до Інтернету. Конкурентоспроможна вартість і технічні обмеження пристроїв IoT змушують виробників вбудовувати в ці пристрої спеціальні функції, які мають забезпечити рівень безпеки і довгострокового захисту вразливих місць, що перевищує аналогічні показники комп’ютерів. Суттєве збільшення кількості й типів пристроїв IoT також може сприяти збільшенню ймовірності кібератак. З урахуванням функції взаємопідключення пристроїв IoT, кожен

підключений пристрій, що не має достатнього захисту, потенційно негативно впливає на безпеку і стійкість Інтернету не тільки локально, а й у глобальному масштабі.

Розглядаючи пристрої, підключені до Інтернету речей, необхідно розуміти, що їх безпека не є абсолютною. Безпека пристрою IoT не визначається поняттям захищеності або незахищеності. Безпеку IoT слід розглядати швидше як межі вразливості пристрою. Ці межі охоплюють як незахищені пристрої, що не мають функцій безпеки, так і надзвичайно безпечні системи з декількома рівнями захисту.

Безпека і стійкість Інтернету речей визначається ефективністю оцінки ризиків та їх усунення. Безпека пристрою — це функція управління ризиком того, що пристрій буде зламано, з урахуванням збитків, які виникли в результаті, а також часу і ресурсів для забезпечення необхідного рівня захисту.

Безпеку Інтернету речей можна побудувати на фундаменті із чотирьох наріжних каменів: безпека зв’язку, захист пристроїв, контроль пристроїв та контроль взаємодій у мережі [11].

Канал зв’язку має бути захищеним, для цього застосовуються технології шифрування та автентифікації, — щоб пристрої знали, чи можуть вони довіряти віддаленій системі. Нові криптографічні технології, наприклад ECC (Elliptic Curve Cryptography), працюють на порядок краще за попередників у слабо потужних чіпах IoT 8-bit 8MHz. Не менш важливим є керування ключами для перевірки автентичності даних та достовірності каналів їх отримання. Провідні центри сертифікації вже вбудували «сертифікати пристроїв» у понад мільярд пристроїв IoT, надавши можливість виконувати перевірку справжності широкого спектру пристроїв, у тому числі стільникові базові станції, телевізори та багато іншого.

Захист пристроїв — це насамперед забезпечення безпеки та цілісності програмного коду. Тема безпеки коду виходить за межі цієї роботи, натомість звернемо увагу на його цілісність. Підписання коду потрібне для

підтвердження правомірності його запуску. Також необхідний захист коду під час його виконання, щоб злочинці не мали можливості перезаписати код під час завантаження. Підписання коду криптографічно гарантує, що він не був зламаний після підписання та є безпечним для пристрою. Таке підписання може бути реалізовано на рівнях «application» і «firmware» навіть у пристроях з монолітним типом прошивки. Всі критично важливі пристрої — датчики, контролери тощо, мають бути налаштовані на запуск тільки підписаного коду.

Пристрої мають бути захищеними й на наступних після запуску коду етапах. Тут допоможе захист на основі хоста, який забезпечує «hardening», розмежування доступу до системних ресурсів та файлів, контроль підключень, пісочницю, захист від вторгнень, захист на основі поведінки та репутації. Також треба згадати такі можливості хостового захисту, як блокування, протоколювання та оповіщення для різних операційних систем IoT. Останнім часом багато засобів хостового захисту були адаптовані для IoT, і тепер вони добре опрацьовані та налагоджені, не вимагають доступу до хмари та турботливо витрачають обчислювальні ресурси IoT-пристроїв.

Повністю позбавитися вразливостей у пристроях IoT неможливо. Їх треба усувати, і це може відбуватися протягом тривалого часу після передачі обладнання споживачеві. Механізм оновлення через повітря (over-the-air) вимагає програмного та апаратного забезпечення, що підтримує таке оновлення, а саме потрібна підтримка одержання та встановлення патчів, отриманих від провайдера через бездротову мережу. Зазвичай нове програмне забезпечення встановлюється, замінюючи застарілі файли на нові версії.

Деякі загрози зможуть подолати будь-які вжиті заходи незалежно від того, наскільки добрим є захист. Тому важливо мати можливості аналітики безпеки в IoT. Системи для аналітики безпеки допоможуть краще зрозуміти мережу, помітити підозрілі, небезпечні чи зловмисні аномалії.

3 МЕТОДИ ПІДВИЩЕННЯ НАДІЙНОСТІ ІОТ-МЕРЕЖ

3.1 Дублювання даних та резервування

Дублювання даних та резервування є важливими методами забезпечення надійності в IoT-мережах. Ці методи передбачають створення копій даних або пристроїв для запобігання втраті даних у випадку виникнення помилок або відмов. Розглянемо ці методи детальніше.

Дублювання даних — це створення копій даних та їх збереження на різних пристроях або вузлах мережі. Якщо одна копія даних стає недоступною або пошкодженою, інша може бути використана для відновлення втрачених даних. Це забезпечує збереження інформації та неперервну доступність даних навіть у разі відмови одного з пристроїв.

Дублювання даних може бути реалізоване на різних рівнях мережі. Так, на рівні даних створюють дублікати файлів або баз даних, щоб забезпечити їх безпеку та доступність. На рівні мережі можуть використовуватися резервні маршрути або дублювання мережевих вузлів, щоб забезпечити роботу мережі навіть у разі відмови деяких компонентів.

Резервування — це створення резервних пристроїв або компонентів, які можуть бути активовані в разі відмови основного пристрою. Це дозволяє забезпечити надійну безперебійну роботу мережі навіть у випадку відмови окремих компонентів.

Наприклад, в IoT-мережах встановлюють резервні пристрої або сенсори, які можуть активуватися автоматично, якщо основний пристрій несправний. Це забезпечує безперервне збирання даних та функціонування системи, навіть якщо деякі компоненти вийшли з ладу.

Крім того, використовують резервні мережеві маршрути або комутатори, які можуть бути активовані автоматично у випадку відмови

основних мережевих шляхів або пристроїв.

3.2 Технології забезпечення надійності передачі даних

Технології забезпечення надійності передачі даних в IoT-мережах важливі для забезпечення безперебійного обміну інформацією між пристроями та системами. Розглянемо декілька технологій, які для цього використовуються.

3.2.1 Коригуючий код

Коригуючий код є методом, який дозволяє виявляти та виправляти помилки, що виникають під час передачі даних. До інформації, що передається, додають додаткові біти, які дозволяють виявити та виправити помилки при отриманні даних. Найпоширеніші коригуючі коди включають коди Хемінга, коди Боуза-Чоджера та циклічні повторювані коди.

3.2.2 Протоколи корекції помилок

Протоколи корекції помилок дозволяють виявляти та виправляти помилки без необхідності повторної передачі даних. Ці протоколи використовуються для додаткового захисту даних та забезпечення надійності передачі. Вони базуються на математичних алгоритмах, які дозволяють відновити втрачену або пошкоджену інформацію. Протоколи корекції помилок використовуються в різних бездротових технологіях, таких як Wi-Fi, Bluetooth, для забезпечення надійної передачі даних.

3.2.3 Передавачі зворотного зв’язку

Передавачі зворотного зв’язку є важливою технологією, що допомагає забезпечити надійність передачі даних в мережах Інтернету речей (IoT). Ці передавачі дозволяють підтверджувати успішне отримання даних та виявляти помилки, забезпечуючи двосторонній обмін інформацією між відправником та отримувачем. Існують такі ключові аспекти передавачів зворотного зв’язку.

Підтвердження доставки. Передавач зворотного зв’язку дозволяє відправнику отримувати підтвердження про успішне отримання даних від отримувача. Тобто після відправки даних відправник очікує підтвердження від отримувача. Якщо підтвердження не надходить або надходить з помилками, відправник повторює передачу даних.

Повторна передача. У випадку, коли отримувач не підтверджує отримання даних або виявляється помилка, передавач зворотного зв’язку повторно передає дані. Це дозволяє забезпечити надійну передачу навіть при втраті пакетів або появі помилок у каналі зв’язку.

Підтвердження достовірності. Передавач зворотного зв’язку також може використовуватися для підтвердження достовірності інформації. Отримувач може повернути підтвердження, що дані були отримані без спотворень або змін. Це дозволяє підтвердити цілісність даних та запобігти несанкціонованій модифікації.

Передавачі зворотного зв’язку використовуються в різних протоколах та системах IoT для забезпечення надійності передачі даних. Наприклад, у протоколі TCP підтвердження доставки та повторна передача є важливими компонентами для забезпечення надійності передачі даних.

Ці технології забезпечують надійну комунікацію в IoT-мережах, зменшують ризик втрати даних і допомагають підтримувати стабільне з’єднання між пристроями. Використання передавачів зворотного зв’язку дозволяє забезпечити надійну передачу даних та зменшити вплив помилок та

збоїв на функціонування IoT-мереж.

3.2.4 Механізми повторної передачі та повторного з’єднання

Ці механізми дозволяють повторно передавати дані або встановлювати з’єднання в разі його втрати або розриву. Вони використовуються для забезпечення надійності передачі даних та підтримки стабільного з’єднання між пристроями. Для цього використовуються такі протоколи, як TCP (Transmission Control Protocol) у мережах TCP/IP.

3.3 Алгоритми маршрутизації та керування мережею

Алгоритми маршрутизації та керування мережею визначають оптимальний шлях передачі даних і керують ресурсами мережі для забезпечення надійності. Ці алгоритми враховують фактори, такі як відстань, пропускна здатність, вартість передачі даних, навантаження мережі та стан пристроїв. Вони дозволяють забезпечувати ефективну передачу даних та уникати перевантажень або «вузьких місць» у мережі.

3.4 Заходи забезпечення безпеки та конфіденційності

Забезпечення безпеки та конфіденційності є критичними аспектами надійності IoT-мереж. Це включає в себе захист від несанкціонованого доступу, злому, витоку інформації та інших кібератак. Для цього використовуються різні методи, такі як шифрування даних, автентифікація, авторизація, контроль доступу та механізми виявлення та виправлення помилок. Забезпечення безпеки включає не тільки захист даних, що передаються, але й захист самого пристрою, мережі та зв’язку між ними.

4 BLOCKCHAIN ТЕХНОЛОГІЇ В IOT-МЕРЕЖАХ

4.1 Визначення технології Blockchain

Технологія Blockchain має значний потенціал застосування в різних сферах діяльності, однак найбільш перспективою галуззю її застосування є Інтернет речей і кіберфізичні системи. Технологія Blockchain пропонує рішення проблеми безпеки і конфіденційності в середовищі Інтернету речей, забезпечуючи новий обчислювальний шар, де дані можуть бути безпечно оброблені та проаналізовані, залишаючись приватними [12].

Технологія Blockchain — це система запису та передачі інформації, що дозволяє зберігати дані у вигляді ланцюжка блоків. Кожен блок містить інформацію про певну кількість транзакцій та хеш попереднього блоку. У такий спосіб кожен блок забезпечує взаємозв'язок із попереднім блоком, утворюючи ланцюжок. Першим і найбільш відомим прикладом використання технології Blockchain є криптовалюта — Bitcoin. На сьогодні криптовалюта перетворилась на визнаний суспільством платіжний засіб, віртуальну валюту, яку приймають великі й малі підприємства, корпорації та сервіси по всьому світу.

Blockchain працює як розподілена база даних, яка здійснює облік усіх операцій у мережі. Операції мають позначку часу і зберігаються у блоках, кожен із яких ідентифікується власним криптографічним хешем (рис. 4.1). Blockchain повністю зберігається в кожному вузлі мережі. Blockchain не потребує для своєї роботи довіри між вузлами мережі, оскільки будь-який вузол може самостійно перевірити, чи збігається його копія бази з копіями, які зберігаються в інших вузлах.

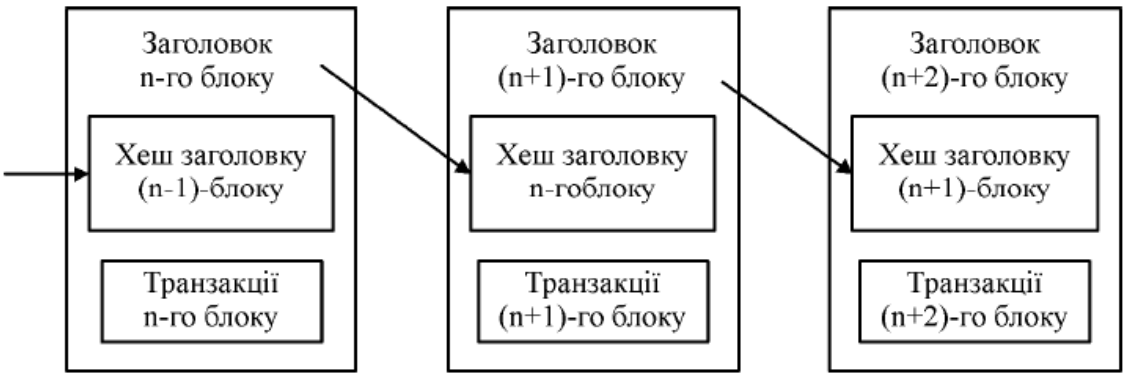

Рисунок 4.1 – Спрощена послідовність блоків

4.2 Переваги технології Blockchain

Переваги технології Blockchain, які забезпечують її ефективне використання в середовищі Інтернету речей:

1) Blockchain є публічною розподіленою базою всіх транзакцій у мережі, яка підтримується одноранговою мережею;

2) мережа Blockchain стійка до збоїв, оскільки вона функціонує без єдиної точки відмови;

3) Blockchain є незмінною і довговічною розподіленою базою, після того як транзакції записані в Blockchain, вони не можуть бути змінені або видалені;

4) усі транзакції в мережі Blockchain захищені криптографічними методами;

5) мережа Blockchain має високий ступінь масштабованості;

6) Blockchain дає можливість пристроям IoT здійснювати операції автономно без довіреної сторони.

Вказані переваги технології Blockchain роблять її перспективним інструментом для вирішення проблем у галузі безпеки й конфіденційності в IoT.

Водночас зі вказаними перевагами використання технології Blockchain у середовищі IoT має низку обмежень, які потребують реагування:

1) створення блоків потребує значних обчислювальних ресурсів, тоді

як більшість IoT-пристроїв мають обмежені апаратні ресурси;

2) створення блоків потребує багато часу, проте для більшості додатків IoT потрібна низька затримка реакції на подію;

3) протоколи Blockchain значно збільшують трафік у мережі, що може бути критичним для мереж IoT із бездротовими каналами зв'язку.

Отже, технологія Blockchain пропонує вирішення проблеми безпеки й конфіденційності в середовищі Інтернету речей, забезпечуючи новий обчислювальний шар, у якому дані можуть бути безпечно оброблені та проаналізовані, залишаючись приватними. Але для ефективного використання технології Blockchain у середовищі IoT має бути розроблена особлива архітектура Blockchain, яка враховувала би зазначені вище обмеження IoT та забезпечувала безпеку й конфіденційність даних.

5 АНАЛІЗ ІСНУЮЧИХ ПЛАТФОРМ IOT

Платформи IoT відіграють важливу роль у розробці масштабованих додатків та сервісів IoT, які з'єднують реальний та віртуальний світи, тобто об'єкти, системи та людей. Усі платформи IoT зазвичай пов'язані з платформами активації додатків IoT (AEP), які складаються з більшості таких компонентів: підключення та нормалізація, управління пристроями, база даних, управління обробкою та діями, аналітика, візуалізація, додаткові інструменти та зовнішні інтерфейси.

5.1. Amazon Web Service (AWS)

AWS IoT надає хмарні послуги, які дозволяють пристроям IoT підключитися до інших пристроїв та хмарних служб AWS. AWS IoT надає програмне забезпечення для пристроїв, яке може допомогти інтегрувати пристрої IoT у рішення на базі AWS IoT. Якщо пристрої можуть підключатися до AWS IoT, то він може підключати їх до хмарних служб, що надає AWS.

AWS IoT дозволяє задовольнити очікування користувача завдяки найсучаснішим технологіям. Щоб забезпечити керування й підтримку пристроїв IoT на місцях, AWS IoT Core підтримує протоколи MQTT, MQTT через WSS, HTTPS і LoRaWAN. AWS надає безліч хмарних сервісів для підтримки програм на основі IoT [13].

Переваги AWS:

1) дозволяє підключати будь-яку кількість пристроїв до хмари та інших пристроїв без необхідності надання серверів або управління ними;

2) дозволяє вибрати найбільш відповідний протокол зв'язку для підключення та управління пристроями IoT;

3) забезпечує автоматизовану конфігурацію під час першого

підключення пристрою до AWS IoT Core, а також повне шифрування в усіх точках підключення.

Недоліки AWS:

1) високий поріг входження для новачків;

2) потребує кращої інформаційної панелі реєстрації для перегляду телеметрії з активних підключених пристроїв.

5.2 Microsoft Azure

IoT-рішення від Microsoft — це набір хмарних сервісів, інтегрованих із середовищем Azure, які організують двонаправлений обмін даними між пристроями і хмарою. Сервіси дозволяють підключити до хмари практично будь-які мережеві пристрої, ідентифікувати їх і управляти ними [14].

Microsoft Azure підтримує всі популярні протоколи, надає локальне сховище, для кожного підключеного пристрою надає виділену чергу повідомлень та забезпечує безпеку за рахунок шифрування і підписування переданих даних.

Переваги Microsoft Azure:

1) пропонує надійний профіль безпеки;

2) пропонує добрі варіанти масштабованості;

3) реалізує декілька можливостей, що працюють на основі штучного інтелекту.

Недоліки Microsoft Azure:

1) не допомагає керувати хмарним центром обробки даних;

2) має проблеми зі зміною обчислювальних потужностей.

5.3 Apple HomeKit

Apple HomeKit — це платформа, вбудована в iOS для управління вашим розумним будинком. Основна ідея: замість того, щоб встановлювати

на свій телефон багато різних розумних домашніх додатків, які не взаємодіють між собою, можна скористатися послугами HomeKit, яка об'єднала всі додатки і запропонувала управління на пристроях, а також через Siri на iPhone, iPad, Apple Watch, HomePod Mini та Mac [15].

Apple HomeKit має два елементи: сама HomeKit є стандартною фоновою програмною технологією, якій повинні відповідати пристрої, щоб отримати доступ до сервісів. Apple Home — це аналог користувача, додаток для iOS.

HomeKit досі є однією з найскладніших екосистем розумного будинку. Ця мережа допомагає пристроям HomeKit спілкуватися між собою в кожному куточку вашого будинку. Більшість пристроїв працюють через Wi-Fi або Bluetooth, але з випуском HomePod Mini як концентратора HomeKit з'явився третій протокол: Thread. Стандарт Thread обіцяє спростити пристрої.

Переваги Apple HomeKit:

1) жорсткі стандарти безпеки та конфіденційності Apple;

2) екосистема Apple.

Недоліки Apple HomeKit:

1) сувора програма сертифікації, через яку продукти з підтримкою HomeKit повільно виходять на ринок.

5.4 Huawei OceanConnect

OceanConnect Huawei — це технологія, яка реалізує уніфікований та безпечний доступ до мережі, надає можливість керування додатками, пристроями та системами. Вона забезпечує понад 170 відкритих програм API та додатків агента серіалізації (процес трансформації структури даних в послідовність байтів), які допомагають партнерам пришвидшити та спростити доступ до терміналу, захистити мережеве підключення для безперебійного з'єднання з продуктами партнерів, одночасно забезпечуючи партнерам по співпраці єдині послуги, у тому числі технічну підтримку,

маркетингову підтримку та ділове співробітництво [16].

OceanConnect можна розділити на управління внутрішніми підключеннями та ввімкнення додатків. Управління внутрішніми підключеннями — це бездротове підключення, загальна платформа з розумними шлюзами.

Переваги Huawei OceanConnect:

1) підтримуються та надаються попередньо інтегровані додатки;

2) підтримуються різноманітні бездротові мережеві з'єднання і дротовий доступ; доступ може бути одночасно фіксованим і мобільним;

3) надаються потужні та відкриті можливості інтеграції: мережевий API, API безпеки, API трьох категорій.

Недоліки Huawei OceanConnect:

1) нові послуги розгортаються дуже повільно через складний процес прийняття рішень клієнтами щодо проєктів IoT, витрати на проєкти є дуже високими, а цикли проєктів — довгими.

6 СТВОРЕННЯ КОНЦЕПЦІЇ МЕРЕЖІ IOT

Концепція мережі IoT може виглядати так. Датчик температури й вологості передає інформацію на контролер. Контролер вмикає та вимикає розумний кондиціонер і розумний зволожувач повітря. Фітнес-браслет передає інформацію про рух і серцевий ритм людини. Аналізуючи ці дані, контролер розуміє, що людина прокинулась, і вмикає кавомашину.

6.1 Встановлення брокера MQTT

Для встановлення брокера MQTT локально можна використовувати різні рішення. Один із популярних варіантів — використання Mosquitto, який є open-source-реалізацією брокера MQTT.

Після встановлення брокера MQTT і перевірки його працездатності можна перейти до розробки програми для Arduino або ESP8266, яка взаємодіє із цим брокером.

Для створення простої програми для Arduino, використовуючи Arduino IDE та платформу ESP8266 (яка може застосовуватися для доступу до Інтернету), можна використовувати бібліотеку PubSubClient для роботи з протоколом MQTT.

Лістинг 6.1 — Надсилання повідомлення до брокера MQTT

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>

const char* ssid = "Your_SSID";
const char* password = "Your_Password";
const char* mqtt_server = "IP_MQTT_broker";

WiFiClient espClient;
PubSubClient client(espClient);

void setup_wifi() {
  delay(10);
  Serial.begin(115200);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
```

```
    Serial.println("Connecting to WiFi...");
  }
  Serial.println("Connected to WiFi");
}

void callback(char* topic, byte* payload, unsigned int length) {
  Serial.print("Message arrived [");
  Serial.print(topic);
  Serial.print("] ");
  for (int i = 0; i < length; i++) {
    Serial.print((char)payload[i]);
  }
  Serial.println();
}

void reconnect() {
  while (!client.connected()) {
    Serial.println("Attempting MQTT connection...");
    if (client.connect("arduinoClient")) {
      Serial.println("Connected to MQTT broker");
      client.subscribe("iot_topic");
    } else {
      Serial.print("Failed, rc=");
      Serial.print(client.state());
      Serial.println(" Retrying in 5 seconds");
      delay(5000);
    }
  }
}

void setup() {
  setup_wifi();
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);
}

void loop() {
  if (!client.connected()) {
    reconnect();
  }
  client.loop();
  client.publish("iot_topic", "Hello, IoT from Arduino!");
  delay(5000);
}
```

6.2 Система контролю температури

Потрібен датчик температури, наприклад DHT11, який буде вимірювати температуру.

Щоб додати можливість вмикання та вимикання кондиціонера залежно від температури, потрібно буде взаємодіяти з кондиціонером. Подальші дії залежать від його конкретної моделі та інтерфейсу.

У загальному коді для Arduino можна використати стандартні порти (наприклад, digitalWrite) для симуляції сигналів, які відповідають вмиканню і вимиканню кондиціонера. Важливо врахувати технічні особливості

конкретного кондиціонера та можливості керування ним.

Приведений нижче код надсилає HTTP-запити до кондиціонера через його IP-адресу та порт для управління станом (вмиканням чи вимиканням).

Лістинг 6.2 — Контролювання температури

```
#include <ESP8266WiFi.h>
#include <DHT.h>

const char* ssid = "Your_SSID";
const char* password = "Your_password";
const char* condIp = "IP_conditioner";

WiFiClient client;

#define DHTPIN 2
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);

const int pinCondOn = 5;
const int pinCondOff = 6;

void setup() {
  Serial.begin(115200);
  sensors.begin();
  pinMode(pinCondOn, OUTPUT);
  pinMode(pinCondOff, OUTPUT);
  connectToWiFi();
}

void loop() {
  float temperature = readTemperature();

  if (temperature < 18.0) {
    controlAirConditioner(true);
  } else if (temperature > 28.0) {
    controlAirConditioner(false);
  }
  delay(5000);
}

void connectToWiFi() {
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting to WiFi...");
  }
  Serial.println("Connected to WiFi");
}

float readTemperature() {
  sensors.requestTemperatures();
  float temperature = sensors.getTempCByIndex(0);
  Serial.print("Temperature: ");
  Serial.println(temperature);
  return temperature;
}

void controlAirConditioner(bool turnOn) {
  if (client.connect(condIp, 80)) {
    String request = turnOn ? "GET /turnOn" : "GET /turnOff";
    client.print(request);
    client.stop();
  }
}
```

У цьому коді створено функції readTemperature для зчитування температури від датчика та controlAirConditioner для керування кондиціонером через мережу.

6.3 Система контролю вологості

Існує кілька популярних типів датчиків вологості, які можна використовувати з мікроконтролерами, такими як Arduino або ESP8266. Було обрано датчик з DHT-серії, а саме DHT11. Це дешевий та досить поширений датчик від компанії Adafruit. Він дозволяє вимірювати температуру та вологість.

Щоб додати можливість вмикання та вимикання зволожувача повітря залежно від вологості, потрібно буде взаємодіяти зі зволожувачем повітря. Подальші дії залежать від його конкретної моделі та інтерфейсу.

Лістинг 6.3 — Контролювання вологості повітря

```
#include <ESP8266WiFi.h>
#include <DHT.h>

const char* ssid = "Your_SSID";
const char* password = "Your_password";
const char* humidifierIp = "IP_humidifier";

WiFiClient client;

#define DHTPIN 2
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);

const int pinHumidifierOn = 7;
const int pinHumidifierOff = 8;

void setup() {
  Serial.begin(115200);
  pinMode(pinHumidifierOn, OUTPUT);
  pinMode(pinHumidifierOff, OUTPUT);
  connectToWiFi();
}

void loop() {
  float humidity = readHumidity();

  if (humidity < 40.0) {
    controlHumidifier(true);
  } else if (humidity >= 60.0) {
    controlHumidifier(false);
  }

  delay(5000);
```

```
}

void connectToWiFi() {
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting to WiFi...");
  }
  Serial.println("Connected to WiFi");
}

float readHumidity() {
  float humidity = dht.readHumidity();
  Serial.print("Humidity: ");
  Serial.println(humidity);
  return humidity;
}

void controlHumidifier(bool turnOn) {
  if (client.connect(humidifierIp, 80)) {
    String request = turnOn ? "GET /turnOn" : "GET /turnOff";
    client.print(request);
    client.stop();
  }
}
```

У цьому коді створено функції readHumidity для зчитування вологості від датчика та controlHumidifier для керування зволожувачем повітря через мережу.

6.4 Система контролю прокидання людини та вмикання кавомашини

Фітнес-браслети визначають момент прокидання в основному за допомогою вбудованого акселерометра та інших сенсорів. Основний принцип полягає в тому, що під час сну рухи людини мають специфічні характеристики, які можна виявити за допомогою цих сенсорів.

Акселерометр реєструє зміни в рухах тіла під час сну. Під час прокидання людина частіше починає робити активні рухи та повороти, або тривалість руху рук збільшується.

Деякі фітнес-браслети використовують оптичні сенсори для вимірювання серцевого ритму. Прокидання може супроводжуватися змінами в серцевому ритмі, які можна використовувати для визначення фаз сну.

Лістинг 6.4 — Контролювання прокидання людини та вмикання кавомашини.

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_SleepyDog.h>
#include <Adafruit_MMA8451.h>

Adafruit_MMA8451 mma = Adafruit_MMA8451();
bool wasActive = false;

int getHeartRate() {
  return random(60, 100);
}

void setup() {
  Serial.begin(115200);
  if (!mma.begin()) {
    Serial.println("Could not find a valid MMA8451 sensor, check wiring!");
    while (1);
  }
}

void loop() {
  sensors_event_t event;
  mma.read();
  mma.getEvent(&event);

  if (event.acceleration.z > 9.5 && !wasActive) {
    int heartRate = getHeartRate();
    if (heartRate > 70 && heartRate < 90) {
      turnOnCoffeeMachine();
    }
    wasActive = true;
  } else if (event.acceleration.z <= 9.5) {
    wasActive = false;
  }

  delay(1000);
}

void turnOnCoffeeMachine() {
  Serial.println("Turning on the coffee machine!");
}
```

Цей код передбачає, що акселерометр може вимірювати рухи відповідно до осі Z. Поріг акселерації 9,5 вибрано на основі того, що зазвичай на Землі акселерація, яку ми відчуваємо, дорівнює приблизно 9,8 м/с^2 (величина прискорення вільного падіння на поверхні Землі). Отже, значення 9,5 встановлює достатньо високий поріг для виявлення активності, але треба експериментувати із цим значенням залежно від конкретних умов.

Якщо значення занадто велике, то буде складно виявити дрібні рухи чи тремтіння. Якщо значення занадто мале, то буде важко відрізнити активність від природного коливання датчика.

При змінах у русі під час прокидання відбудеться виклик функції

turnOnCoffeeMachine(), яку необхідно доповнити власною логікою для вмикання кавомашини. Точність цього методу залежатиме від конкретного фітнес-браслета та його здатності вимірювати активність та рухи користувача.

6.5 Хмарне збереження даних

Для того, щоб дані з мікроконтролера ESP8266 потрапляли в хмару, можна використовувати різні підходи та сервіси. Одним із популярних способів є використання платформи IoT, такої як ThingSpeak, Blynk, або використання хмарних платформ, таких як AWS IoT, Google Cloud IoT, Microsoft Azure IoT Hub.

Брокер MQTT Mosquitto використовується для збереження та обробки даних з ESP8266. Потім ці дані передаються в хмарні платформи, наприклад, ThingSpeak, для моніторингу та відображення. Загальний сценарій роботи мережі виглядає наступним чином:

1) ESP8266 публікує дані на брокері MQTT Mosquitto;

2) необхідно створити канал на ThingSpeak та отримати API ключ для запису;

3) необхідно написати скріпт або використати сервіс, який в подальшому буде відправляти дані з брокера MQTT Mosquitto на ThingSpeak за допомогою API ключа;

ВИСНОВКИ

Інтернет речей є наступним кроком на шляху до оцифрування сучасного суспільства, де предмети і люди пов’язані один з одним через комунікаційні мережі і з’являється можливість повідомляти про їх стан та стан навколишнього середовища. Розробники таких програм намагаються якомога більше удосконалювати свої роботи. Інтернет речей створює нові можливості та забезпечує конкурентні переваги для бізнесу як на поточних, так і на нових ринках.

В результаті виконання кваліфікаційної роботи було досліджено існуючі методи підвищення надійності IoT мереж та створено концепцію мережі IoT. В роботі описано поняття, концепцію та ідею Інтернету речей. Проаналізовано, які протоколи, технології та типи мереж застосовуються для побудови мережі IoT, на якій архітектурі побудовані дані мережі, які пристрої беруть участь у роботі мережі. Було розглянуто використання Blockchain технології в IoT мережах, а також проаналізовано існуючі платформи IoT.

Результати кваліфікаційної роботи можуть бути використані для створення реальної IoT мережі за створеною концепцією.

Отже, надійність мереж IoT є критично важливою, оскільки ці системи стають все більш популярними в різних галузях, включаючи промисловість, медицину та споживчі товари. Існуючі рішення надають широкий спектр засобів для забезпечення безперебійної роботи та захисту даних в мережах IoT. Моніторинг і контроль приладів є одним із важливих заходів, за яким слід пильно стежити та використовувати його в режимі реального часу для безпеки та комфорту людей.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. What is the internet of things (IoT)? [Електронний ресурс]. — Режим доступу : www / URL: https://www.techtarget.com/iotagenda/definition/Internet-of-Things-IoT (дата звернення: 17.11.2023).
2. IoT Ecosystem: Top 7 Components (+Bonus). [Електронний ресурс]. — Режим доступу : www / URL: https://sumatosoft.com/blog/iot-ecosystem-top-7-components/ (дата звернення: 17.11.2023).
3. Інтернет речей: мережева архітектура та архітектура безпеки. [Електронний ресурс]. — Режим доступу : www / URL: https://www.bizmaster.xyz/2020/12/internet-rechei-merezheva-arkhitektura-ta-arkhitektura-bezpeky.html (дата звернення: 19.11.2023).
4. MQTT Vs. HTTP for IoT. [Електронний ресурс]. — Режим доступу : www / URL: https://www.hivemq.com/article/mqtt-vs-http-protocols-in-iot-iiot/ (дата звернення: 21.11.2023).
5. MQTT protocol. [Електронний ресурс]. — Режим доступу : www / URL: https://www.javatpoint.com/mqtt-protocol (дата звернення: 21.11.2023).
6. Протокол MQTT. [Електронний ресурс]. — Режим доступу : www / URL: https://cqr.company/ua/wiki/protocols/mqtt-protocol/ (дата звернення: 23.11.2023).
7. IoT Session Layer Protocols. [Електронний ресурс]. — Режим доступу : www / URL: https://www.javatpoint.com/iot-session-layer-protocols (дата звернення: 25.11.2023).
8. Протокол обмеженого застосування (CoAP). [Електронний ресурс]. — Режим доступу : www / URL: https://cqr.company/ua/wiki/protocols/constrained-application-protocol-coap/ (дата звернення: 27.11.2023).
9. Огляд ZigBee та його застосування в IOT. [Електронний ресурс]. — Режим доступу : www / URL: https://nekta.tech/obzor-zigbee-i-primenenie-v-iot/

(дата звернення: 28.11.2023).

10. What is IoT architecture? [Електронний ресурс]. — Режим доступу : www / URL: https://www.avsystem.com/blog/iot/what-is-iot-architecture/ (дата звернення: 01.12.2023).

11. Русінов Ю. М. Надійність IoT -мереж / Ю. М. Русінов // Радіоелектроніка та молодь у XXI столітті : матеріали 27-го Міжнар. молодіж. форуму, 10–12 травня 2023 р. – Харків : ХНУРЕ, 2023. – Т. 5. – С. 23–24.

12. Яцків, Н. Г. Перспективи використання технології блокчейн у мережі Інтернет речей [Текст] / Н. Г. Яцків, С. В. Яцків // Науковий вісник НЛТУ України. — 2016. — Вип. 26.8. — С. 381–387.

13. What is IoT? [Електронний ресурс]. — Режим доступу : www / URL: https://aws.amazon.com/what-is/iot/ (дата звернення: 07.12.2023).

14. Рішення із захисту IoT. [Електронний ресурс]. — Режим доступу : www / URL: https://www.microsoft.com/uk-ua/security/business/solutions/iot-security (дата звернення: 09.12.2023).

15. Що таке HomeKit? Все що потрібно знати про «розумний будинок» від Apple. [Електронний ресурс]. — Режим доступу : www / URL: https://mac.org.ua/blog/how-homekit/ (дата звернення: 10.12.2023).

16. OceanConnect: Cloud IoT for the Future. [Електронний ресурс]. — Режим доступу : www / URL: https://www.huawei.com/en/huaweitech/publication/84/ocean-connect-cloud-iot (дата звернення: 11.12.2023).

Відомість кваліфікаційної роботи

«Моделі підвищення надійності IoT-мереж»

	Прізвище та ініціали відповідальної особи	Підпис	Дата
Роботу виконав студент групи СКСм-22-1 Структура кваліфікаційної роботи: – пояснювальна записка 62 с.; – графічний матеріал 15 арк..	Русінов Ю.М.		12.01.24
Керівник роботи	Немченко В.П.		24.01.24
Перевірка на плагіат здійснена. Оригінальність авторського тексту складає ___95___ %	Литвинова Є.І.		17.01.24
Нормоконтроль проведено :			24.01.24