МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
Факультет економіки та бізнес-адміністрування
Кафедра економічної кібернетики

Густера О.М.

КОНСПЕКТ ЛЕКЦІЙ

з дисципліни «Інтернет речей»

за спеціальністю 051 «Економіка», освітньо-професійною програмою «Цифрова економіка»

Укладачі:
асистент кафедри економічної кібернетики
к.е.н Густера О.М.

Конспект лекцій розглянутий та схвалений
на засіданні кафедри
економічної кібернетики
Протокол № № _ від _____р.

Завідувач кафедри_____Н.О. Іванченко

Рекомендовано Вченою радою ФЕБА (протокол №

Густера О.М. Інтернет речей. Конспект лекцій. – К. :НАУ, 2019. – 80 с.

Містить основні положення з дисципліни «Інтернет речей». Для студентів-магістрів спеціальності 051 «Економіка» спеціалізації «Цифрова економіка».

© Густера О.М. 2019

Вступ

Метою викладання дисципліни є теоретична та практична підготовка студентів до вивчення систем обробки даних та принципів функціонування інтернету речей.

Завданнями вивчення навчальної дисципліни є:

– вивчення та засвоєння основних понять та функцій IoT;

– вивчення та засвоєння структури і функціонування IoT;

– вивчення та засвоєння архітектури сучасних IoT та їх впровадження в сучасних підприємствах.

Компетентності, які набуває студент в результаті вивчення навчальної дисципліни:

- здатність розв’язувати складні спеціалізовані завдання та практичні проблеми у сфері економіки та у процесі навчання, що передбачає застосування теорій та методів економічної науки,
- здатність зберігати та примножувати моральні, культурні, наукові цінності та примножувати досягнення суспільства на основі розуміння історії та закономірностей розвитку предметної області, її місця у загальній системі знань про природу і суспільство та у розвитку суспільства, техніки і технологій, використовувати різні види та форми рухової активності для активного відпочинку та ведення здорового способу життя,
- здатність свідомо та соціально-відповідально діяти на основі етичних міркувань і принципів академічної доброчесності,
- здатність проводити дослідження та презентувати результати,
- здатність застосовувати науковий, аналітичний, методичний інструментарій для управління економічною діяльністю,
- здатність формулювати професійні задачі в сфері економіки, вибирати належні напрями і відповідні методи для їх розв’язання, беручи до уваги наявні ресурси,
- здатність обґрунтовувати управлінські рішення щодо ефективного розвитку суб’єктів господарювання,

- здатність оцінювати можливі ризики, соціально-економічні наслідки управлінських рішень,
- здатність до розробки сценаріїв і стратегій розвитку соціально-економічних систем.

Міждисциплінарні зв’язки: у процесі ознайомлення з курсом студенти повинні спиратися на знання, набуті під час попереднього прослуховування дисциплін, передбачених навчальним планом з даної спеціальності («Інформатика», «Основи програмування», «Бази даних», «Системи підтримки прийняття рішень»).

Модуль №1 «Архітектура та стандартизація IoT»

Лекція № 1.1. Стандартизація IoT

У цій темі починається опис архітектурних елементів Інтернету речей в його нинішньому стані. Далі описується кілька конкуруючих ключових протоколів, що були розроблені для забезпечення роботі Інтернету речей. Наостанок розглянуто існуючі платформи, які сьогодні зазвичай використовуються в середовищі Інтернету речей (IP).

Інтернет речей (IP, також відомий як Інтернет об'єктів) - концепція, яку вперше створив Кевін Ештон у 1999 році в контексті управління ланцюгами постачання, де він описав систему, в якій фізичний світ підключений до Інтернету через датчики. Ця концепція розвивалася і поширювалася у межах декількох галузей протягом багатьох років, впливаючи на здатність сутностей Інтернету речей передавати інформацію, змінювати стан одне одного, взаємодіяти один з одним без втручання людини та виконувати дії, які впливають на фізичний світ.

Дослідження і розробка цього підходу активно розлягається в декількох галузях. Програми розроблені в різних галузях, починаючи від охорони здоров'я до транспорту, і, в останні роки, інтеграція об'єктів, датчиків та хмарних обчислень, були активною областю досліджень. Проте розвиток цих технологій ще перебуває у фази становлення, і перед ними постають багато проблем . Нараз продовжуються дослідження щодо стратегій стандартизації, безпеки, інфраструктури, інтерфейсів та протоколів зв'язку, з кількома конкуруючими реалізаціями та організаціями.

Бачення Інтернету речей має кілька глибинних наслідків для нашого суспільства. Весь річний економічний вплив на ринок, спричинений IP, може досягати значення від 2 до 6 трильйонів доларів до 2025 року. Перед початком наступного десятиліття очікується розгортання більш ніж 200 мільярдів розумних сутностей IP, а очікуваний об'єм інформаційних потоків між об'єктами IP складатиме близько 45% від усього інтернет-трафіку. Більше того,

очікується, що переваги будуть не тільки економічними, а також підвищать якість життя в таких аспектах, як охорона здоров'я або життя в містах.

Деякі області Інтернету речей також включають аспекти сталого розвитку. Зокрема, сектор інформаційно-комунікаційних технологій (ІКТ) та Інтернет-речей мають потенціал для підвищення ефективності економіки шляхом надання споживачам та підприємствам можливості вимірювати та оптимізувати свої дії. За оцінками, на 15% світові викиди вуглецю може бути зменшено до 2020 року завдяки використанню інтелектуальних технологій.

Сучасний стан Інтернету речей характеризується різноманітним набором ініціатив, стандартів та реалізацій. Стандартизація та взаємодія залишаються проблемою. Початкові програми розроблені у таких областях як логістика або енергетика, з їх власними протоколами та архітектурами. Поточні зусилля зближуються зі стандартними специфікаціями, такими як OMA LWM2M, та базовими архітектурами, такими як Європейська архітектура Інтернет-речей (IoTA). Ці зусилля об'єднують ключових гравців та дозволяють використовувати реалізації, які можуть буті повторно використаними у різних доменах.

Елементи, що складають Інтернет речей в сучасних і майбутніх стосунках, широко обговорювалися у літературі та класифікуються кількома авторами. Опишемо наступні найважливіші компоненти та архітектурні елементи в Інтернеті речей.

На найнижчому рівні існують сенсорні елементи, котрі беруть початок з самих "речей": датчиків, виконавчих механізмів та інших пристроїв, що є частиною Інтернету речей. Очікується, що сотні мільйонів таких пристроїв стануть частиною Інтернету речей, а спеціальні схеми ідентифікації та адресації необхідні завдяки великому масштабу їх розгортання та їх частому обмеженому характеру з точки зору енергії та обчислювальних ресурсів.

Тема 1.2. Архітектура IoT.

Як було зазначено до цього, протокол MQTT реалізовує парадигму публікація-підписка (publish-subscriber). Ця парадигма дозволяє відокремити клієнта, що публікує повідомлення (publisher) від інших клієнтів, що отримують ці повідомлення (subscribers). Більше того, MQTT є асинхронним протоколом, а це означає, що при його використанні непотрібно блокувати клієтів, які очікують на повідомлення. Іншою цікавою властивістю MQTT є те, що він не потребує, щоб клієнт, який публікує повідомлення та клієнт, який приймає повідомлення, були одночасно підключені один до одного.

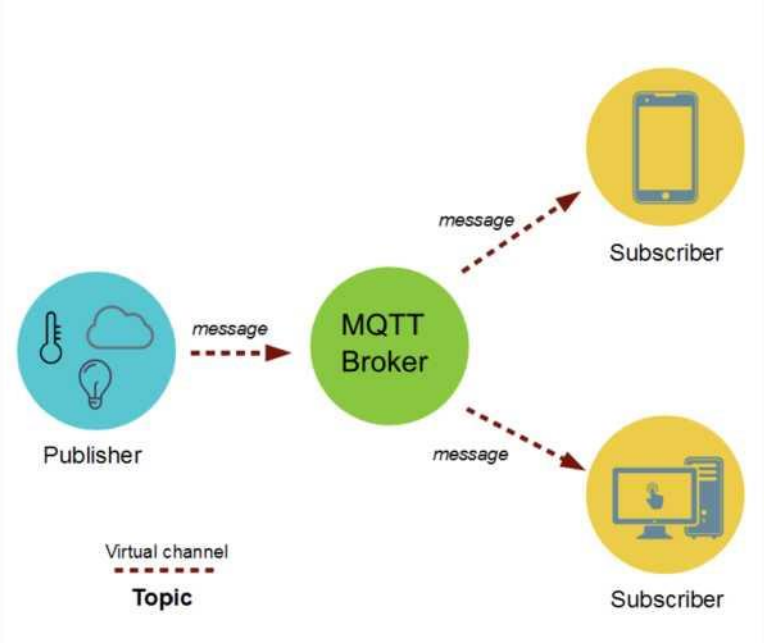

Рисунок 2.1 - Взаємодія MQTT клієнтів

Зазвичай, архітектура протоколу MQTT (рис. 2-2) може бути розділена

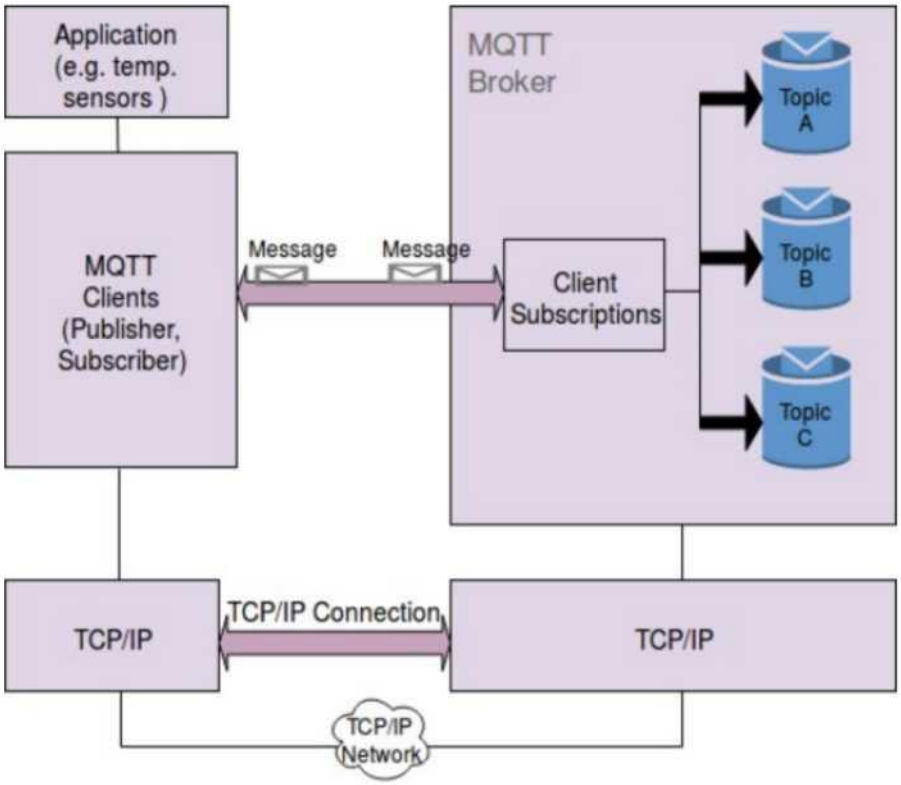

Рисунок 2.2 - Архітектура MQTT

на два головних компонента:

1. Клієнт може бути у ролі відправника або отримувача повідомлень, при цьому він завжди встановлює з'єднання з сервером (брокером повідомлень). Клієнт може виконувати наступні функції:

- публікувати повідомлення для зацікавлених користувачів
- підписуватись на відповідні теми для отримання повідомлень
- відписуватись від тем
- від'єднатись від брокера повідомлень

2. Брокер повідомлень контролює поширення інформації і є відповідальним за прийом повідомлень від публікуючих клієнтів, фільтрування повідомлень та їх надсилання всім підписаним клієнтам. Брокер може виконувати наступні функції:

- встановлювати з'єднання з клієнтами

- отримувати повідомлення від клієнтів
- правильно оброблювати різні види запитів, зокрема підписки та відписки від користувачів
- після отримування повідомлень від паблішерів надсилати їх всім зацікавленим користувачам

Брокери повідомлень

Брокер повідомлень є ключовою часиною MQTT-системи. Він забезпечує з'єднання між додатками або фізичними пристроями та корпоративними системами. Брокери відповідають за підписки, створення сесій та керування ними, пропущені повідомлення та за загальну безпеку системи, включаючи аутентифікацію та авторизацію.

Найбільш поширеними брокерами є наступні:

- Mosquitto
- Flepsi
- JoramMQ
- HiveMQ
- VerneMQ

Eclipse Mosquitto - це брокер повідомлень з відкритим сирцевим кодом (під ліцензією EPL/EDL), що реалізовує протокол MQTT версій 3.1 та 3.1.1. Mosquitto є доволі легкою системою, та внаслідок цього може використовуватись на багатьох видах пристроїв: від простих одноплатних комп' ютерів до повноцінних серверів.

Mosquitto також надає бібліотеку на мові програмування C, що дозволяє реалізовувати MQTT-клієнти, а також доволі популярні консольні клієнти: mosquitto_pub та mosquitto_sub.

Flepsi - це безкоштовний публічний MQTT-брокер, що працює у хмарі та підтримує протоколи версій 3.1, 3.1.1, 5.0. Ця система реалізовує архітектуру для обробки великих об'ємів повідомлень, ізольовані простори імен, підтримку веб- сокетів та SSL. Крім цього, вона надає можливість її використання як за комерційним договором, так і безкоштовно.

JoramMQ - це програмне забезпечення від компанії Scalagent, яке, поміж

інших функцій, реалізовує брокера повідомлень, що повністю підтримує протоколи MQTT 3.1, JMS 2.0 та AMQP 1.0. Іншими словами, цей брокер дозволяє одночасно працювати з багатьма протоколами та стандартами. MQTT може працювати з використанням різного транспорту, зокрема TCP/IP, TLS (SSL), веб-сокетів та захищених веб-сокетів.

HiveMQ - це MQTT-брокер, що був спроектований та побудований з нуля з орієнтацією на те, щоб забезпечувати максимальну масштабованість на безпеку, що задовольняє корпоративні стандарти. В ньому одразу наявна підтримка веб- сокетів та SDK з відкритим сирцевим кодом, що дозволяє значно розширити функціональність цієї системи та інтегрувати з багатьма іншими компонентами. При цьому, для тестування цієї системи можна використати публічно доступний сервер.

VerneMQ - це розподілений брокер повідомлень, що є готовим до використання в корпоративних цілях та може працювати з високими навантаженнями. Він може маштабуватись як горизонтально, так і вертикально, щоб підтримувати велику кількість паблішерів та споживачів повідомлень, які працюють одночасно. При цьому, дана система працює з мінімальними затримками та забезбечує стійкість до помилок. Доповнення до VerneMQ можуть бути розроблені багатьох мовах, наприклад, Erlang, Elixir, Lua та багатьох інших, що підтримують роботу з HTTP веб-хуками. VerneMQ використовує сучасні протоколи для розповсюдження повідомлень, а також систему LevelDB для реплікації стану у кластері. На додаток до зазначеної інформації, VerneMQ є відкритим програмним забезпеченням під ліценцією Apache 2.

Тема 1.3. Веб речей WoT.

В останні десятиліття HTTP став використовуватись значно ширше, ніж як просто механізм для навігації між веб-сторінками в інтернеті. Сьогодні цей протокол також використовується для спілкування між декількома пристроями, для автоматизації, інтернету речей та багато чого іншого. Таке значне поширення набув через свою доступність та простоту у використанні.

HTTP - це протокол типу запит-відповідь, в якому клієнти надсилають запити до сервера для отримання інформації, а сервери, в свою чергу, надсилають відповіді цим клієнтам. HTTP-запит зазвичай складається з методу, адреси ресурсу, декількох заголовків та необов'язкового корисного навантаження в тілі запиту. HTTP-відповідь складається з трицифрового статус-коду, декількох заголовків на необов'язкового корисного навантаження. Схема роботи цього протоколу наведено на рис. 2.3.

Рисунок 2.3 - Схема роботи протоколу HTTP

Кожен ресурс, який спочатку задумувався як колекція гіпертекстових чи HTML-документів, має бути ідентифікованим за його універсальною адресою (URL). Для отримання такого ресурсу з сервера клієнти можуть просто використати метод GET, зазначивши відповідну адресу ресурсу. Методи PUT та DELETE дозволяють клієнту завантажувати інформацію на сервер та видаляти її, відповідно. Метод POST, в свою чергу, дозволяє клієнту відправити необхідні дані на якийсь ресурс на сервері, наприклад, веб-форму.

Порівняння роботи протоколів.

Нижче наведено порівняння роботи протоколів при надсилання однієї

тисячі повідомлень:

Таблиця 2.1 Порівняння протоколів MQTT та HTTP

1 тисяча повідомлень	Кількість переданих байтів	Кількість пакетів	Час в секундах
MQTT	283743	265	5911
HTTP	15474263	12079	115669

Як ми бачимо MQTT є в 20 разів швидшим та потребує в 50 разів менше трафіку в контексті задачі відправки узгоджених даних, що залежать від часу.

Eclipse Ponte - це проект, метою якого є побудова мосту між CoAP, MQTT та HTTP. Ponte надає просте REST API, що дозволяє визначити потреби пристрою через REST. В майбутньому Ponte також має дозволяти виконувати перетворення даних між різними форматами (JSON, MsgPack, Byson, Binary JSON, Extensible Markup Languate), надавати засоби для аутентифікації та авторизації, але на даний момент це ще не підтримується.

Ponte може замінити брокера повідомлень в MQTT та дозволити різним пристроям комунікувати один з одним, якщо вони використовують MQTT, CoAP або HTTP. Так як більшість IoT-пристроїв використовують MQTT або CoAP, Ponte є доволі перспективним з точки зору інтеркомунікації. Крім цього, Ponte дозволяє пристрою самому вибрати протокол для спілкування, який є для нього найбільш зручним, при цьому, надаючи можливість спілкуватись з іншими пристроями, що використовують інші протоколи. Іншою суттєвою перевагою є те, що цю систему можуть використовувати веб-додатки, що працюють з використанням HTTP.

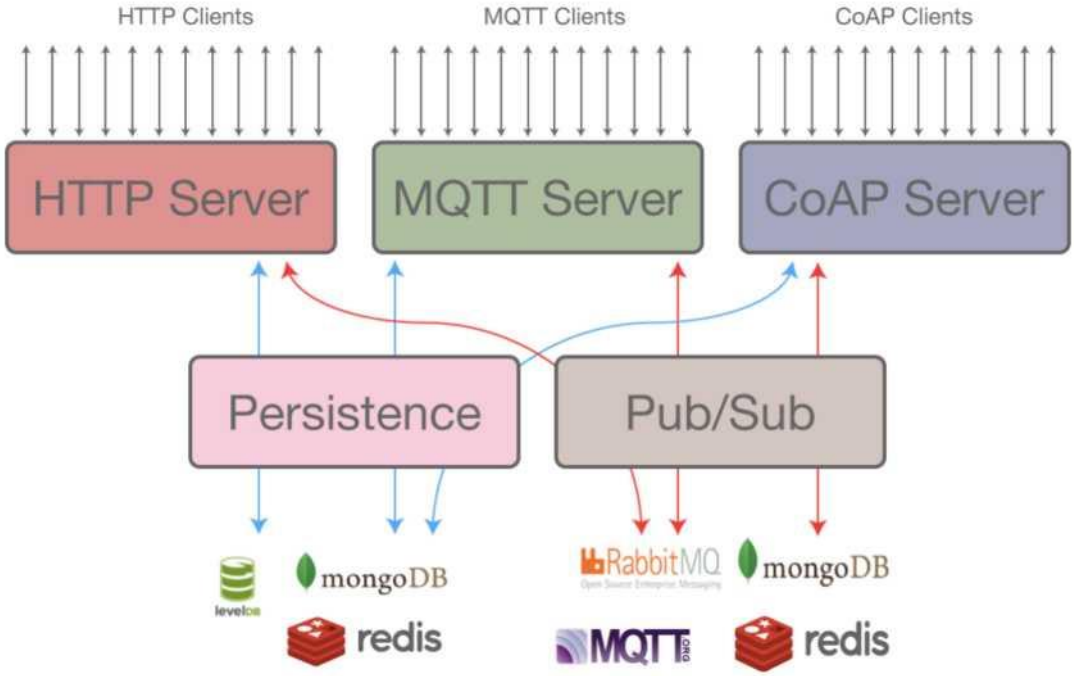

Рисунок 2.4 - Архітектура Eclipse Ponte

Якщо додати до Ponte можливість конвертування даних в різних форматах, то в результаті вийде доволі непогане рішення для IoT. В такому випадку розробники зможуть обирати не тільки найбільш зручний протокол, а і найбільш зручний формат даних. Однак, потрібно врахувати той факт, що система має конвертувати дані доволі ефективно, щоб користувачі не відчували затримок.

Тема 1.4. Когнітивний інтернет речей CIoT.

Інтернет речей є відкритою парадигмою, яка надзвичайно сприйнятлива і адаптивна для нових принципів і архітектури, що відносяться до різних напрямів розвитку науки і техніки. У зв'язку з цим надзвичайно плідним може надати використання в IoT принципів і методів когнітивності (лат. Cognitio, «пізнання, вивчення, усвідомлення») шляхом створення когнітивного Інтернету речей CIoT (Cognitive Internet of Things).

Когнітивної означає наявність у об'єкта IoT наступних загальних властивостей:

– здатність до самоаналізу і реконфігурації з рахунком наявного оточення, а також маючи на меті досягнення цілей, обумовлених виконуваних завдань;

– здатність адаптувати свій стан згідно з наявними умовами або подіями, на основі певних критеріїв і знань про попередні стани;

– можливість динамічно змінювати свою топологію і / або експлуатаційні параметри відповідно до вимог конкретного користувача, коли це необхідно в рамках поточної політики обслуговування, оптимізації пропускної здатності мережі або інших показників;

– самоконфігурація з наявністю розподіленого управління на основі правил;

– можливість самостійного визначення свого поточного стану і, з урахуванням цього стану - планування своєї роботи, приймаючи певні рішення у відповідь на ситуацію, що склалася.

На практиці когнітивні інтернет-речі зможуть:

– використовувати технології отримання знань про свою операційну і географічному середовищі, місцезнаходження, наприклад за допомогою стандартних технологій позиціонування GPS / ГЛОНАСС;

– встановлювати самостійно або використовувати готові правила взаємодії між об'єктами (інтернет-речами);

– динамічно і автономно коригувати свої операційні (робочі) параметри і протоколи відповідно до отриманих знань для досягнення заздалегідь

визначених цілей, зокрема вибирати найбільш підходящу технологію передачі радіосигналу;

– навчатися на основі досягнутих результатів з використанням кращих практик і найбільш ефективних політик для досягнення цілей створення IoT.

Концепція CIoT передбачає наявність IoT з механізмами кооперації і «розумності». Об'єкти CIoT зможуть скласти певне уявлення про стан і умови функціонування навколишніх об'єктів, сприймати знання про оточуючих об'єктах, продукувати логічні висновки з накопичених знань і здійснювати дії щодо адаптації до зовнішніх і внутрішніх умов. Відповідно, в архітектурі CIoT (рисунок 2.2) з'являються когнітивні вузли CN (cognitive node) або когнітивні елементи CE (cognitive element), які здатні автономно оптимізувати, наприклад, технічні характеристики мережі відповідно до визначених умовами. В свою чергу CE або CN об'єднуються в домени автономності AD (Autonomous Domain), де ці пристрої щодо тісно пов'язані між собою, в тому числі на певній території, і можуть поєднувати свою поведінку. При цьому кожне CE або CN зберігає властивість автономності. У свою чергу, багато домени AD можуть транскордонного взаємодіяти і кооперуватися через Мультидоменні кооперацію MDC (Multi-Domain Cooperation). Для організації такої взаємодії в кожному автономному домені використовується когнітивний агент CA (Cognitive Agent), який взаємодіє з CE або CN в своєму домені. Таким чином, взаємодія доменів можливо як в цілому, так і на рівні окремого когнітивного елемента. При цьому в кожному домені AD існують і прості, що не когнітивні вузли, які, знаходяться під контролем когнітивних вузлів.

Основою для розвитку схеми когнітивного управління є концепція віртуального об'єкта VO (Virtual Object), який є представленням фізичного об'єкта або об'єкта реального світу RWO (Real-World Object), що в принципі не суперечить вимогам Рекомендації MCE-T Y.2060. Віртуальний об'єкт динамічно створюється або видаляється, створюючи тим самим уявлення динаміки змін RWO. Для опису можливостей автоматичної агрегації VO, щоб забезпечити умови для виконання додатків в запропонованій схемі

когнітивного управління вводиться поняття концепції композитних (складені) віртуальних об'єктів CVO (Composite VO).

Розглянемо застосування концепції CIoT на прикладі оптимізації часу надання невідкладної допомоги хворому за конкретною адресою. Хворий перебуває під дистанційним контролем системи медичного моніторингу на базі послуги IoT. Нехай сенсорна система на тілі хворого («body sensor») зафіксувала різке і тривалий зміна параметрів стану людини - різке збільшення частоти дихання, пульсу, серцеву аритмію, ознаки непритомності. Показання сенсорів - RWO, призводять до зміни стану об'єктів VO, пов'язаних з RWO через шлюз. Спеціальне додаток для обробки і трансляції показань сенсорів обробляє зазначену інформацію VO і перетворює її до виду, який може бути використаний CVO, в даному випадку - медичним центром за допомогою процедури запиту і збіги ситуації RSM «Request and Situation Matching». Однак якщо в ході пошуку необхідний CVO не знайдений, або відсутній вільний медичний автомобіль (ситуація «все на виїзді»), то за допомогою процедури прийняття рішень задіється інший відповідний для даного випадку VO, наприклад сенсор пожежної сигналізації. В результаті в схемі бере участь новий CVO - служба порятунку - на основі аналізу близькості ситуації до небезпечної для здоров'я людини. В результаті швидка допомога може бути надана хворому не медичним центром, а службою порятунку, фахівці якої також мають навички медичної допомоги. З урахуванням того, що подія відбувається в «розумному місті», медична інформація про стан хворого може транслюватися паралельно на CVO медичного центру і на CVO «розумного автомобіля» служби порятунку. Одночасно тривожне повідомлення транслюється на CVO служби регулювання дорожнього руху, яка організовує «Зелену вулицю» в напрямку будинку хворого. Таким чином, автором була описана ситуація наочно показує переваги когнітивності і когнітивного управління стосовно інтернету речей.

2. Модуль №2 «Протоколи передачі даних IoT»

Тема 2.1. Способи взаємодії з інтернет-речами.

Constrained Application Protocol (CoAP) - це веб протокол передачі, спеціально розроблений для використання з обмеженими вузлами та обмеженими мережами. Він був розроблений з урахуванням вимог до малопотужних пристроїв, з можливими невеликими обсягами пам'яті, в мережевих середовищах, які можуть мати втрати та бути малопотужними. У специфікації розглядаються вузли з такими можливостями, як 8-бітні мікроконтролери та мережі з втратами, типова пропускна здатність яких дорівнює десяткам кбіт/с, наприклад IPv6, через бездротові локальні бездротові мережі (6LoWPANs). Він був розроблений з метою підтримки існуючих M2M додатків, виконуваних поверх існуючих інфраструктур та взаємодії з існуючими веб-стандартами та парадигмами.

CoAP є стандартом Internet Engineering Task Force (IETF). Стабільна специфікація визначена в RFC 7252. CoAP узгоджується з веб-парадигмою, надаючи URI для знаходження ресурсів, типи Інтернет-медіа для їх опису, а також встановлення віповідності до HTTP дієслів без збереження станів. CoAP має низькі накладні витрати і сумісний з існуючою IP-інфраструктурою з UDP прив'язкою. Інші прив'язки для SMS, TCP, TLS та Websocets існують як чернетки IETF, причому деяким з них запропоновані реалізації.

В CoAP дані та інформація моделюються як ресурси. Подібно до HTTP, ресурси розташовані та ототожнюються з URI, і їх можна отримати, створювати, модифікувати або видаляти за допомогою семантичних методів. Ресурси доступні за допомогою моделі запиту/відповіді, як у HTTP. Проте запити та відповіді зазвичай надсилаються за допомогою UDP, а не за допомогою TCP-з'єднання, як у HTTP. Повідомлення CoAP можуть бути підтверджені для надійності або не підлягають підтвердженню для програм, де достатній рівень доставки є найкращим.

Запити CoAP застосовуються до ресурсів і побудовані шляхом визначення методу, який застосовується до перезавантаженого ресурсу, його

ідентифікатора, корисної інформації, типу медіа та додаткових метаданих про запит. Після отримання та інтерпретації запиту сервер надсилає відповідь CoAP з кодом відповіді, який може вказувати на успіх, помилку клієнта чи помилку сервера. Відповідь відповідає запиту за допомогою токену, створеного клієнтом у запиті.

Кінцеві точки CoAP здатні активно надсилати дані, коли встановлюються відповідні зв'язки. Цей механізм є реалізацією шаблону проектування *спостерігач*. Клієнт, який зацікавлений у отриманні оновлень певного фрагмента даних, може зробити GET запит на відповідний ресурс, встановлюючи параметр спостереження до значення 0. Якщо можливо, сервер додасть клієнта до списку спостерігачів ресурсу. Встановлюючи параметр "Спостереження" в 1, клієнт може вимагати, щоб він був знятий з реєстрації. Коли змінюється ресурс, додаткові відповіді, які називаються сповіщеннями, будуть надсилатися клієнту з оновленими даними. Кожне сповіщення включає токен, який клієнт подав у запиті. Сповіщення та нормальна відповідь є іншим варіантом, з тією лише різницею, що присутній параметр спостереження.

Протокол XMPP: Extensible Messaging and Presence Protocol.

Протокол Extensible Messaging and Presence (XMPP) був розроблений для спілкування та обміну повідомленнями. Він був стандартизований IETF більше десяти років тому, тому вже є добре перевіреним протоколом, який широко використовувався по всьому Інтернету. Однак, будучи старим протоколом, він не забезпечує необхідні послуги для деяких нових виникаючих додатків. З цієї причини в 2017 році компанія Google припинила підтримку стандарту XMPP через відсутність підтримки в усьому світі. Проте останнім часом XMPP знову привернув увагу як протокол зв'язку, придатний для IP.

XMPP працює над TCP і забезпечує публікацію/підписку (асинхронно), а також повторного запиту/відповіді (синхронно) систем обміну повідомленнями. Він призначений для операцій зв'язку в режимі реального часу, і, таким чином, він підтримує невеликий розмір повідомлення і обмін повідомлень з низькою затримкою. Як видно з назви, XMPP розширюваний і

дозволяє специфікації XMPP Extension Protocols (XEP) збільшити його функціональність.

XMPP має TLS/SSL захист, побудований в ядрі специфікації. Однак він не надає параметрів QoS, які роблять його непрактичним для M2M-зв'язків. Тільки успадковані механізми TCP забезпечують надійність.

XMPP підтримує архітектуру публікації/підписки, яка більш підходить для IP, на відміну від підходу запиту/відповіді CoAP. Крім того, це вже встановлений протокол, який підтримується в усьому Інтернеті як плюс відносно нового MQTT. Однак XMPP використовує XML-повідомлення (extensible Markup Language), які створюють додаткові витрати через непотрібні теги, і вимагають розбору XML, що потребує додаткової обчислювальної здатності, що збільшує споживання енергії.

Платформи Інтернету речей

Технологія IP дозволяє легко підключати до мережі різні речі і розробляти програми для управління цими речами. Всі складності, пов'язані з підключенням сервісів і хмарою для цих пристроїв, є завданням IP платформ.

Платформа IP забезпечує безшовну інтеграцію з різними апаратними засобами, використовуючи ряд популярних протоколів зв'язку, застосовуючи різні типи топології (пряме підключення або шлюз) і використовуючи SDK, коли це необхідно.

Використовуючи інтерфейси для інтеграції надані платформою, ви також можете передавати зібрані дані IP в певні системи аналізу / візуалізації даних зберігання даних, а також передавати дані на підключені пристрої (конфігурація, повідомлення) або між ними (елементи управління, події) на використовуючи різні види призначених для користувача додатків.

Платформа IP також часто згадується як проміжне програмне забезпечення Інтернету речей, яке підкреслює його функціональну роль як медіатора між апаратним і прикладним рівнями. Кращі платформи IP здатні інтегруватися практично з будь-яким підключеним пристроєм і вбудовуватися в додатки, які використовуються пристроями. Ця незалежність від базового обладнання і програмного забезпечення дозволяє одній платформі IP

реалізувати ті ж функції IP на будь-якому підключеному пристрої одним і тим же способом.

1.3.1 Платформа Інтернету речей Thingspeak

ThingSpeak - це відкрита платформа даних Інтернету речей, яка базується на загальнодоступних хмарних технологіях. ThingSpeak дозволяє збирати, аналізувати дані та обробляти їх в режимі реального часу за допомогою Open API. Завдяки додаткам та плагінам стало можливим зберігання даних, візуалізація, моніторинг та інтеграція даних користувача з різними сторонніми платформами, включаючи провідні платформи IP, такі як ioBridge, Arduino, Twilio, Twitter, ThingHTTP, MATLAB. Дані датчика збираються у кожний канал, який містить вісім полів, які можуть містити будь-який тип даних, три поля з місця розташуванням та одне поле стану. Різноманітні програми, такі як TimeControl (автоматично виконують дії заздалегідь за допомогою програми ThingSpeak), TweetControl (слухають сигнали Twitterverse та реагують у режимі реального часу), React (реагує, коли дані каналу відповідають деяким умовам), TalkBack (команда черги для пристрою користувача) покращують реакційні вимірювання.

ThingSpeak API - це інтерфейс з відкритим вихідним кодом, який прослуховує вхідні дані, маркує їх і виводить як для користувачів (через візуальні графіки), так і для комп'ютерів (через легко аналізований код). Він може використовуватися з мікроконтролером Arduino, а також для зв'язку з графічними інтерфейсами операційних систем через скрипт Python. ThingSpeak особливо корисний для невеликих апаратних проектів, де потрібне підключення через Інтернет, але в яких обслуговування виділеного сервера недоцільно. Існують альтернативні сервіси IP, але вони вимагають оплати деяких функцій і, отже, не є відкритими.

Платформа Інтернету речей Google Cloud IoT

Google пропонує аналогічний сервіс хмарних обчислень у вигляді Google Cloud, який досить популярний в області хмарних обчислень, платформою як сервіс, маштабованою системою баз даних NoSQL, двигуном машинного навчання TensorFlow та документо-орієнтованою базою даних. Його

можливості Інтернету речей також збираються на платформі Google Cloud IoT. З точки зору вартості це зазвичай трохи дорожче, ніж Microsoft Azure, але все-таки дешевше, ніж Amazon Web Services. Google Cloud Platform володіє 12 відсотками на ринку хмарної реклами, що на 10 відсотків більше, ніж у попередньому році, та на 20 відсотків ніж два роки до цього, і є третьою на сьогодні платформою IP.

Платформа Інтернету речей Oracle IoT cloud

Oracle IoT являє собою комбінацію з чотирьох основних параметрів, таких як Open, що з'єднує будь-який тип пристрою починаючи з датчиків і закінчуючи шлюзами; Insight - збирає цінність даних IP з точки зору бізнесу; Secure - забезпечує нормалізовану безпеку для всіх типів пристроїв, даних та гетерогенних з'єднань; Accelerate - дуже швидко перетворює ідею у виконання з мінімальним ризиком. В принципі, Oracle виконує аналіз та інтеграцію отриманих даних із пов'язаних з ним речей. Аналізуючи, він обробляє вхідні потоки даних у реальному часі за допомогою фільтрації подій, кореляції та агрегації. Аналітика великих даних дає змогу Oracle все частіше реагувати на виявлення аномалій, а також керувати механізмами сповіщень. Запит та візуалізація великого обсягу даних прокладає нові ідеї інтелектуальних хмарних сервісів. Хмара Oracle IoT пропонує декілька рішень, включаючи Oracle Java SE / ME Embedded Suite, Java Card, Database та Event Processing, щоб відповідати вимогам до пристроїв з розміром пам'яті 11 MB або більше для Java. Платформа M2M ідеально підходить для тестування та розгортання пристроїв Інтернету речей.

Платформа Інтернету речей KAA

KAA - це багатофункціональна багатопрофільна платформа IP (Apache License 2.0) із відкритим кодом для побудови інтелектуальних кінцевих рішень IP. Вона полегшує обмін даними між приєднаними пристроями, аналітикою даних, візуалізацією та хмарними сервісами IP. Виділення технічних характеристик пристрою, підготовка до використання пристроїв, конфігурація, взаємодія між пристроями та оновлення розповсюдженого вбудованого програмного забезпечення - це основні послуги, що надаються KAA. Дана

платформа забезпечує кінцеві функціональні можливості для роботи великомасштабних рішень Інтернету речей, включаючи безпеку даних, узгодженість, сумісність та управління даними за допомогою SDK, який вбудовується в чип чи пристрій розробника. KAA SDK вимагає як мінімум 10 Кб оперативної пам'яті та 40 Кб ПЗП. SDK збирає ендпоінти, надає профілі конфігурації та забезпечує обмін повідомленнями між обраними ендпоінтами. Зберігання даних здійснюється за допомогою таких варіантів баз даних NoSQL, як Cassandra, Hadoop і MongoDB.

Платформа Інтернету речей Carriots

Carriots - це продукт Wairbut (www.wairbut.com), який надає допомогу будь- якому користувачеві у створенні додатку для IP максимально швидко, заощаджуючи час, витрати та зусилля. Хмарна модель платформи як сервісу (PasS) володіє цілою низкою ключових технічних характеристик, таких як віддалене керування пристроєм, протоколювання активності користувачів, запуск спеціальних засобів для нагадування та експорт даних. RESTful API дає змогу користувачеві працювати з даними використовуючи такі засоби, як Device, Asset, Group, Service, Project, Stream, Rule, Alarm, Listener, Trigger, Networking, Entity та ConfigTrigger. Email, SMS, Twitter, основні утиліти HTTP, для інформування користувача про поточний стан пристрою. Дані зберігаються в базі даних NoSQL Big Data Base, що розширює можливості застосування даних у відповідному сенсі.

Платформа Temboo

Temboo - платформа для генерації коду додатків на хмарній основі. Дана пропозиція зменшує накладні витрати на створення апаратного та програмного забезпечення, що призводить до скорочення часу на розробку та комерціалізацію продукту IP. Понад 90 вбудованих бібліотек під назвою "Choreos" надають користувачеві можливість користуватися цілою низкою сервісів, серед яких прогноз погоди від Yahoo, Amazon Qoud, покупки продуктів через Ebay, керування фотографіями з Flickr, API Facebook Graph, аналітика Google, мікроблоги Twitter, Twilio телефонія, сервіс оплати PayPal, транспортні послуги Uber, трансляція відео в YouTube та багато іншого. Labs -

це експериментальний каталог Choreos, який об'єднує багато засобів, створюючи потужні робочі процеси, які охоплюють більшість основних переваг необхідних для розробки широкого спектру додатків. Він допомагає розробнику візуально налаштовувати апаратне забезпечення для запуску та реагування на онлайн-процеси, зберігати вхідні дані для заощадження оперативної пам'яті, віддалено програмувати обладнання, фільтрувати дані відповідно до вказаних критеріїв.

Платформа ThingWorx

ThingWorx - це популярне рішення, розроблене для створення приватної хмарної платформи. ThingWrox надає інфраструктуру на базі M2M та інфраструктуру Інтернету речей як сервіс, в проектування моделі, об'єднується з SQUEAL (Search, Query, Analysis), щоб включити до неї можливість інтелектуального пошуку. Інтелектуальне середовище виконання є важливим компонентом сумісної робочої моделі ThingWrox. Засіб створення нульових кодів призначений для розробників, щоб скоротити час на розробку продукту відповідно до вимог ринку. Разом з тим мобільні інтерфейси відображаються за допомогою програм, які використовують подійно-орієнтований механізм. Крім того пропонується інноваційний 3D-накопичувач для полегшення роботи користувацьких пристроїв. Нормалізація даних, передача протоколів, організація входу та виходу пристрою, є важливими елементами ефективного керування даними, які вимагають від платформи.

Таблиця 1.1. Порівняння підтримуваних протоколів платформами Інтернету речей

Платформа	Підтримувані ППІ
Google Cloud IoT	MQTT bridge, HTTP bridge
AWS IoT Core	HTTP, MQTT, Websockets
IBM IoT	HTTP, MQTT
Oracle IoT cloud	REST API, MQTT bridge (requires setup)

Kaa	MQTT, HTTP
Carriots	MQTT, HTTP
Temboo	MQTT, CoAP, HTTP
ThingWorx	REST API, MQTT extention

Тема 2.2. Протокол Websockets. Протокол Websocket як частина проекту HTML 5 по розширенню каналів зв'язку через TCP.

Протокол Websockets

Протокол Websocket був розроблений як частина проекту HTML 5 по розширенню каналів зв'язку через TCP. Веб-сайт не є протоколом типу запит / відповідь, ні протоколом типу публікація / підпис. В Websocket клієнт ініціює рукостискання з сервером для створення сеансу. Саме рукостискання нагадує HTTP, тому веб-сервери можуть обробляти веб-сеанси і HTTP-з'єднання через один і той же порт. Однак, після встановлення сеансу, рукостискання не відповідає правилам HTTP. Фактично, під час сеансу HTTP-заголовки видаляються, а клієнти та сервери можуть обмінюватися повідомленнями в асинхронному повнодуплексному з'єднанні. Сеанс може бути зупинений, коли він більше не потрібен ні з сервера, ні з клієнтської сторони. Websocket був створений, щоб зменшити накладні витрати на зв'язок з Інтернетом, забезпечуючи при цьому повнодуплексний зв'язок в режимі реального часу. Існує також підпротокол Websocket, що називається протокол повідомлень для веб- додатків в (WAMP), який надає системні повідомлення / підписки.

Websocket працює над надійним TCP і самостійно не використовує механізми надійності. При необхідності сеанси можуть бути захищені за допомогою веб-сайту через TLS / SSL.

Під час сеанса повідомлень Websocket мають лише 2 байта службових даних. Як повідомляється в відповідних дослідженнях, HTTP-опрос (в REST) повторює інформацію заголовка, коли швидкість передачі даних збільшується, що збільшує затримку. За оцінками, Websocket забезпечує зниження затримок в три рази у порівнянні з половиною дуплексного HTTP-опросу. Websocket не призначений для пристроїв з обмеженими ресурсами, оскільки попередні протоколи та його клієнт-серверна архітектура не підходять для додатків IoT. Однак він призначений для обміну в реальному часі, він безпечний, він мінімізує поточні витрати і,

використовуючи WAMP, може забезпечити ефективні системи обміну повідомленнями. Таким чином, він може конкурувати з будь-яким іншим протоколом, що працює через TCP.

Протокол AMQP: Advanced Message Queuing Protocol

Advanced Message Queuing Protocol (AMQP) - це протокол, який виник у фінансовій галузі. Він може використовувати різні транспортні протоколи, але він передбачає надійний транспортний протокол, такий як TCP.

AMQP забезпечує асинхронну комунікацію за допомогою повідомлення публікація/підписка. Його головна перевага - це функція зберігання та передачі даних, яка забезпечує надійність навіть після збоїв в роботі мережі. Це забезпечує надійність з наступними гарантіями доставки повідомлень:

Тільки один раз: означає, що повідомлення відправляється один раз, не залежно воно доставлено чи ні.

1. *Принаймні один раз*: означає, що повідомлення точно буде доставлено один раз, можливо, і більше.
2. *Точно один раз*: означає, що повідомлення буде доставлено лише один раз.

Безпека досягається з використанням протоколів TLS/SSL на TCP.

Недавні дослідження показали, що AMQP має низький рівень успішності доставки при низькій пропускній здатності та збільшується при збільшенні пропускної здатності. Інше дослідження показує, що, порівнюючи AMQP з вищезгаданим REST, AMQP може відправляти більшу кількість повідомлень у секунду.

Крім того, повідомляється, що середовище AMQP з 2000 користувачами, що поширюються на п'яти континентах, може обробляти 300 мільйонів повідомлень на день. Крім того, JPMorgan, яка є американською компанією з банківських та фінансових послуг, використовує AMQP для надсилання 1 мільярда повідомлень на день.

Тема 2.3. RESTFUL Services.

The Representational State Transfer (REST) насправді не є протоколом, а архітектурним стилем. Він був вперше представлений Роєм Філдінгом в 2000 році, і він широко використовується з тих пір.

REST використовує HTTP методи такі як GET, POST, PUT і DELETE, щоб забезпечити ресурсоорієнтовану систему обміну повідомленнями, де всі дії можуть бути виконані за допомогою синхронних команд HTTP-запитів та відповідей. Тип тіла запиту варіюється між XML або JSON (JavaScript Object Notation) і залежить від HTTP-сервера та його конфігурації. REST вже є важливою частиною IP, оскільки підтримується багатьма комерційними хмарними платформами M2M (machine-to-machine). Крім того, він легко може бути реалізований на смартфонах і планшетах, оскільки вимагає тільки підтримку протоколу HTTP, котрий доступний для всіх дистрибутивах операційних систем (ОС). Функції HTTP можуть бути повністю використані в архітектурі REST, включаючи кешування, аутентифікацію та узгодження типу вмісту.

RESTful сервіси використовують безпечний і надійний протокол HTTP, який є перевіреним у мережі Інтернет. Безпека гарантується шляхом використання протоклів TLS/SSL. Проте сьогодні більшість комерційних платформ M2M не підтримують запити HTTPS. Замість цього вони забезпечують унікальні ключі аутентифікації, які повинні бути у заголовку кожного запиту для досягнення відповидного рівня безпеки.

Хоча REST вже широко використовується в комерційних платформах M2M, малоймовірно, що він стане домінуючим протоколом через неможливість забезпечення прозорої реалізації . Він використовує HTTP, що означає відсутність сумісності з пристроями з обмеженим зв'язком. Це дозволяє використовувати його для кінцевих додатків.

Враховуючи поточну тенденцію використання додатків, що працюють на смартфонах та планшетах, додаткові накладні витрати, пов'язані з протоколами

запитів/відповідей, використовують заряд акумулятора, оскільки вимагають виконувати безперервне опитування та спостереження за статистикою додатку. Вищезгадані накладні витрати є марними, особливо коли немає нових оновлень. Такої проблеми можна уникнути, якщо використати протоколи опублікування/підписки, такі як MQTT або XMPP. З іншого боку, CoAP є легкою версією REST та має такі ж недоліки, що стосуються надзусиль при надсиланні запиту/відповіді. Однак він призначений для роботи через UDP, що робить його здатним використовувати обмежені ресурсні пристрої, в противагу REST.

Telemetry Transport Queue Message (MQTT) - протокол, розроблений IBM, та призначений для забезпечення легких комунікацій M2M. Це асинхронний протокол публікації/підписки, який працює поверх стеку TCP. Спочатку він був розроблений Енді Стенфорд-Кларком (IBM) та Арленом Ніппером у 1999 році. Після внутрішнього використання у IBM він був опрелюднений у 2010 році. MQTT був розроблений простим у застосуванні на обмежених пристроях, що характеризується дефіцитом обчислювальної потужності та мережевими середовищами з обмеженою пропускною здатністю. Його характеристики роблять даний протокол гарним вибором для використання в обмежених середовищах та семантичних зв'язках типу "машина до машини" (M2M) чи "Інтернет-речей". Протоколи публікації/підписки краще відповідають вимогам IP, ніж запит/відповідь, оскільки клієнтам не потрібно вимагати оновлення. Таким чином, пропускна здатність мережі зменшується, а потреба у використанні обчислювальних ресурсів знижується.

У MQTT виробники даних можуть публікувати повідомлення на тему, а споживачі даних можуть підписатися на отримання повідомлень, опублікованих на цю тему. Ці взаємодії опосередковуються та відокремлюються через брокера, тому видавцям і абонентам потрібно лише знати про брокера, а не про інших виробників та споживачів. Теми визначаються за допомогою UTF-8 рядків, що мають структуру дерева, в якій рівні розділені за допомогою прямої косою рисою ('/').

Наприклад, сенсор світла в вітальні будинку можна було б моделювати за темою "home/firstfloor/livingRoom/lightSensor". Температурному датчику у тій же вітальній кімнаті може бути присвоєна тема "home/firstfloor/livingRoom/tempSensor". Додаток, зацікавлений в цих даних, може підписатися на ці теми. MQTT визначає підстановки для підписки на декілька тем у рівнях дерева тем. Багатоступенева підстановка "#" може бути використана для відповідності будь-якій кількості рівнів у межах теми, а шаблон "+" відповідає лише одному рівню теми. У нашому прикладі можливо підписатися на тему "house/firstfloor/#", щоб підписати всі теми, що включають піддерево "house/firstfloor". Це включатиме такі теми, як "house/firstfloor/kitchen/tempSensor", а також тему "house/firstfloor/livingRoom/wall/sensor". Для ілюстрації використання однорівневої підстановки розглянемо тему "house/firstfloor/+/tempSensor", яка відповідає темі "house/firstfloor/kitchen/tempSensor", але не темою з додатковими рівнями, як наприклад "house/firstfloor/livingRoom/northwall/motionSensor ".

MQTT визначає пакети керування, що складаються з фіксованого заголовку, необов'язкового змінюваного за розміром заголовка та корисного навантаження. У контрольних пакетах дані можуть бути представлені з використанням бітів, 16- бітових цілих чисел у великому розмірі або UDF-8-закодованих рядків. Корисні завантаження надсилаються у форматі байтів, а кодування та декодування виконуються програмою.

MQTT забезпечує надійність, забезпечуючи можливість вибору трьох рівнів QoS:

1. *Відправити та забути*: повідомлення відправляється один раз, і підтвердження не потрібно.
2. *Поставляється щонайменше один раз*: повідомлення надсилається принаймні один раз, а підтвердження потрібно.
3. *Доставлено рівно один раз*: Механізм чотирьохстороннього рукостискання використовується для забезпечення доставки

повідомлення рівно один раз.

Хоча MQTT працює на базі TCP, він розроблений для того, щоб мати низькі накладні витрати порівняно з іншими протоколами основаними на протоколі TCP. Крім того, архітектура опублікування/підписки, яку вона використовує, більше підходить для IP, ніж HTTP запит/відповідь, наприклад, тому що повідомлення мають отримати відповіді. Це призводить до нижчої пропускної здатності мережі та ускладнює обробку повідомлень, фактично збільшує термін служби пристроїв, що працюють на батареях.

Для забезпечення безпеки, брокери MQTT можуть вимагати перевірки аутентичності на ім'я користувача та пароль, які обробляються TLS/SSL (Secure Sockets Layer), тобто ті самі протоколи захисту, які забезпечують конфіденційність HTTP-транзакцій у всьому Інтернеті.

Тема 2.4. Напрямки практичного застосування та проблеми впровадження IoT.

За останні декілька років напрямок розвитку сучасних інформаційних технологій суттєво змінився у зв’язку з появою пристроїв IoT. Насамперед це викликає необхідність зміни архітектури обчислювальних пристроїв у напрямку спрощення, зниження собівартості та підвищення енергоефективності. Крім того, постійне збільшення обчислювальної потужності комп’ютерної техніки вже неможливе, і так званий закон Мура не працює в сучасних умовах ринку інформаційних технологій.

Використання пристроїв IoT значно збільшує можливості інформаційних систем сучасних організацій, дозволяючи запроваджувати нові функції, отримувати більш достовірні та актуальні дані при відносно невеликих витратах. При цьому, апаратне та програмне забезпечення дозволяє використовувати пристрої IoT без суттєвих змін існуючої архітектури інформаційної системи організації.

Пристрої IoT на сьогоднішній день в тій чи інший мірі використовуються майже всіма організаціями. Так, наприклад, у великих містах розпочато реалізацію програм Smart City, які направлені на підвищення рівня комфортності та якості послуг, а також безпеки для громадян. В той же час, використання пристроїв IoT негативно впливає на рівень інформаційної безпеки організації.

Як правило використання пристроїв IoT комерційними організаціями не афішується з метою захисту інформації про архітектуру внутрішньої інформаційної системи від зловмисників чи конкурентів. Це дозволяє підвищити рівень інформаційної безпеки організації у короткостроковій перспективі. Але при подібній закритій схемі функціонування інформаційної системи організації виникає проблема запізнення у розвитку. Якщо інформаційна система організації обмежена у взаємодії з навколишнім середовищем, вона буде розвиватися набагато повільніше. Використання нових апаратних та програмних елементів ускладнюється і не завжди вважається доцільним зі сторони регламенту безпеки інформаційної системи організації.

Основним функціональним напрямком розвитку IoT в Україні та у світі є передача телеметричних даних та зворотній зв'язок у вигляді сигналів керування. Але з точки зору інформаційної безпеки, пристрої IoT фактично можуть розглядатися як повноцінні вузли мережі Інтернет. Тобто зловмисники можуть використовувати будь-які пристрої, підключені до мережі Інтернет чи інформаційної системи організації, незалежно від того, для яких функцій вони використовуються кінцевим користувачем.

Сучасний етап розвитку мережі IoT обумовлений переходом на нові стандарти та протоколи передачі даних. Сьогодні як правило використовується обладнання з мінімальним споживанням електроенергії. Це дозволяє мінімізувати витрати на функціонування мережі IoT. Саме тому використання сучасних пристроїв IoT стало економічно доцільним. Використання обладнання на базі мікроконтролерів з низькою тактовою частотою у поєднанні зі

спеціальними протоколами передачі даних дозволяє створювати IoT з низькою собівартістю та високою енергоефективністю.

При розробці такого обладнання та запровадженні спеціальних протоколів передачі даних не завжди враховуються вимоги до інформаційної безпеки організації. Якщо для побутового рівня вбудованих засобів захисту може бути достатньо, використання IoT пристроїв для організації може стати причиною появи нових загроз інформаційної безпеки.

Так, наприклад, існуючі на ринку пристрої IoT для оснащення «розумного будинку» мають досить багато слабких місць у захисті програмного забезпечення та протоколах передачі даних. Приблизно 30% пристроїв IoT для побутового використання мають вразливості. Підключитися та внести зміни у роботу такого пристрою можуть зловмисники, навіть не маючи достатньо глибоких знань у галузі інформаційної безпеки.

Навіть якщо конкретний пристрій IoT чи інформаційна система організація загалом не цікавить зловмисників, вони будуть об’єктами кібератак. Згідно з дослідженнями OWASP (Open Web Application Security Project) кожен новий підключений до мережі Інтернет пристрій IoT вже через 5 хвилин піддається атаці. При цьому атаки проводяться у автоматичному режимі. Тобто зловмисники постійно сканують існуючі в мережі Інтернет пристрої та вузли на наявність уразливостей. Таким чином, якщо увімкнути пристрій IoT, який має відому зловмисникам уразливість, в середньому вже через 5 хвилин зловмисники можуть підключитися до нього та використовувати у своїх власних цілях, змінювати передані дані, зчитувати передані чи отримані повідомлення, виконувати DDOS атаки. Як правило, зловмисники роблять свої атаки непомітними для користувача, тому більшість таких несанкціонованих підключень не фіксуються. Крім того, пристрій IoT може працювати у штатному режимі і повноцінно виконувати свої основні функції, і при цьому одночасно використовуватися зловмисниками для певних злочинних дій.

Наявність такої великої кількості уразливостей обумовлена у першу чергу недосконалістю програмного та апаратного забезпечення, що напряму залежить

від низької собівартості та масового характеру виробництва. У другу чергу це пов’язано з тим, що сьогодні велика кількість ентузіастів на безоплатній основі проводить тестування пристроїв IoT на наявність типових уразливостей та розповсюджують результати своїх досліджень для широкої аудиторії у мережі Інтернет.

Що стосується комерційного використання IoT, можна зробити висновок про більш високий рівень захисту мережі пристроїв. Насамперед це пов’язано з тим, що вбудовані засоби захисту використовуються, а не ігноруються, як у випадку з побутовим використанням.

На рис. 1. представлено основні існуючі проблеми безпеки використання пристроїв IoT в українських організаціях. Відповідно до характеру проблеми поділено на законодавчі, організаційні та технічні.

Слід зауважити, що наведені проблеми є типовими не лише для українських, але і для інших організацій, які використовують пристрої IoT у складі своєї інформаційної системи.

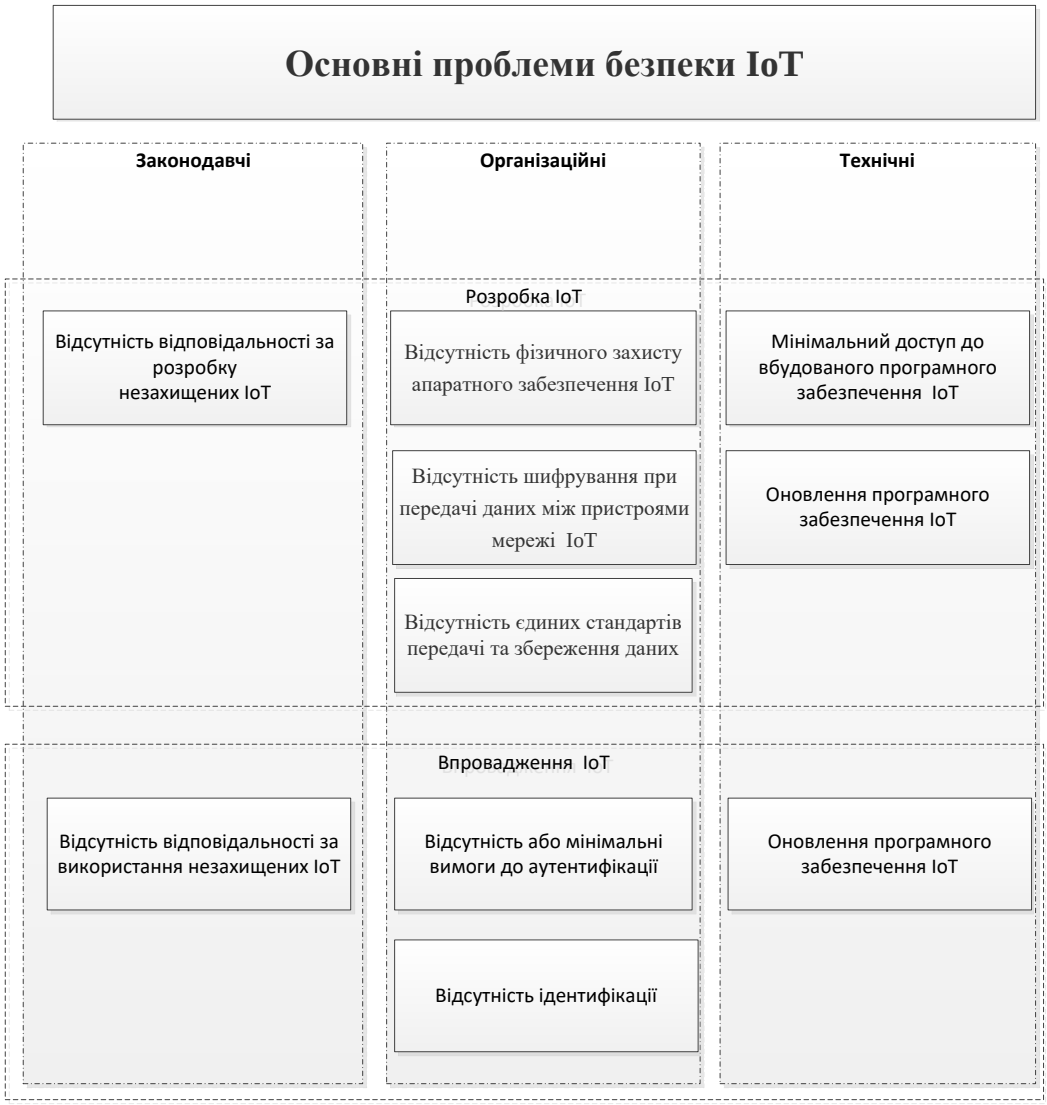

Рис. 1. Основні існуючі проблеми безпеки використання пристроїв IoT в українських організаціях

У зв’язку з жорсткими обмеженнями собівартості обладнання, IoT як правило, мають мінімальний фізичний захист від дій зловмисників або негативного впливу зовнішнього середовища. Найпростішою дією зловмисника може бути фізичне пошкодження чи виведення з ладу пристрою. Також часто зловмисники можуть зчитувати код чи mac-адресу пристрою, для того щоб підмінити його на інший та передавати некоректні дані. Пряме підключення до пристрою дозволяє змінювати його налаштування чи вбудоване програмне забезпечення.

Також скористатися відсутністю фізичного захисту IoT можуть недобросовісні співробітники організації або абоненти системи, які прагнуть уникнути виявлення злочинних дій чи порушень.

Одним з методів зниження впливу даної загрози є підвищення рівня захисту обладнання. Так, наприклад, виробництво апаратного забезпечення з урахуванням стандартів захисту IP-67 чи IP-68 дозволяє майже повністю уникнути впливу зовнішнього середовища. Для обмеження несанкціонованого доступу використовується антивандальне обладнання.

Крім того в деяких випадках достатньо лише зафіксувати факт несанкціонованого втручання в роботу апаратного забезпечення IoT. По аналогії з лічильниками електро, газо та водопостачання, на обладнання доцільно встановлювати індикатори несанкціонованого втручання в роботу пристрою IoT. В такому разі можна виявити факт втручання співробітників чи абонентів IoT.

Так, наприклад, транспортні засоби підприємств можуть оснащатися датчиками пересування та розходу палива, які постійно передають дані та підключені до електричної мережі транспортного засобу. Переривання в роботі датчиків через відсутність електропостачання неможливе, тому в такому разі можна робити висновок про несанкціоноване втручання в апаратну частину IoT. Встановлення антивандального захисту в даному випадку не має сенсу, тому що вартість транспортного засобу в сотні разів перевищує вартість IoT чи можливу користь від його виведення з ладу. Тобто лише оператор транспортного засобу має змогу і мотиви для виведення з ладу чи втручання в нормальну роботу IoT.

Спрощення програмної та апаратної частини IoT призвело також до ускладнення процесу шифрування передачі даних. Обчислювальна потужність найбільш розповсюджених пристроїв IoT не дозволяє використовувати перевірені захищені протоколи захисту даних, що автоматично робить їх уразливими для атак зловмисників. Враховуючи той факт, що більшість мереж, до яких підключаються пристрої IoT є бездротовими, дана загроза є однією з

найбільш актуальних. Прослуховування даних, що передаються між пристроями всередині бездротової мережі без шифрування – достатньо тривіальна задача для зловмисника. Одночасно з телеметричними даними третя особа може отримати аутентифікаційні дані про вузли та виконати підміну одного з пристроїв IoT мережі.

На сьогоднішній день загальносвітовою є проблема відсутності єдиних стандартів передачі та збереження даних у мережах IoT пристроїв. Тобто у даний момент кожна мережа IoT пристроїв відокремлена і не може ефективно співпрацювати з іншими мережами, або може співпрацювати у обмеженому режимі. Кожен виробник як програмного, так і апаратного забезпечення може розробляти і використовувати власну архітектуру та стандарти зв'язку.

У цьому випадку доцільними є розробка та реалізація стандартів програмного та апаратного забезпечення IoT на міжнародному рівні. Це дозволить підвищити взаємозамінність обладнання та спростити використання IoT на практиці. Крім того, уніфікація та стандартизація в галузі IoT дозволяє підвищити якість програмного та апаратного забезпечення.

Згідно з дослідженнями, лише дві третини IoT пристроїв мають пароль для підключення. До інших пристроїв може підключитися будь-який користувач, який має фізичний доступ. Це надає зловмисникам можливість підключатися до пристрою, вносити зміни в налаштування та програмне забезпечення, виконувати будь-які несанкціоновані дії. Також до пристроїв IoT можна підключатися за допомогою бездротового доступу.

Крім того, навіть наявність пароля для більшості IoT пристроїв не є гарантією захисту від зловмисників. Користувачі, як правило, використовують прості стандартні паролі, які не відповідають навіть мінімальним вимогам захисту.

Також дуже часто користувачі залишають встановлений за умовчанням пароль, який є стандартним для даного типу обладнання або програмного забезпечення. В такому разі зловмиснику достатньо буде лише знати тип

обладнання щоб підібрати необхідну комбінацію символів серед невеликого переліку стандартних паролів.

Пристрої IoT не завжди дозволяють проводити ідентифікацію підключеного до них обладнання. Так само користувач, коли здійснює підключення до обладнання IoT, не завжди проводить його ідентифікацію. Це дозволяє зловмисникам здійснювати атаки типу MITM. Також можлива підміна пристроїв на інші. В цьому разі можливе несанкціоноване отримання інформації, підміна інформації .

Реалізація процедури ідентифікації можлива шляхом перевірки ID обладнання IoT та обладнання користувача, шлюзу, чи сервера, які підключаються до пристрою IoT.

Більшість виробників програмного забезпечення IoT не підтримують можливість користувачів змінювати налаштування пов'язані з безпекою пристрою. Так, наприклад, неможливо встановлювати різні права доступу в залежності від категорії користувача: звичайний користувач, адміністратор, керівник і т.д.

Користувачі можуть використовувати такі пристрої без попередньої підготовки та налаштувань. З одного боку це зручно для рядових користувачів, які встановлюють IoT для побутових цілей і не мають жорстких обмежень безпеки. З іншого боку використання найпростіших пристроїв для IoT для підприємств стає причиною уразливостей в інформаційній системі.

Таким чином, доцільним буде формування та запровадження різних класів пристроїв – побутові IoT та промислові IoT. Для організацій з підвищеним рівнем потреб до безпеки, таких як електростанції та промислові підприємства, вкрай важливим є питання запровадження стандартів безпеки пристроїв та мереж IoT.

Рівень безпеки IoT напряму визначається сучасністю вбудованого програмного забезпечення та його відповідністю стандартам безпеки. Не зважаючи на це, не всі виробники обладнання та програмного забезпечення для IoT своєчасно випускають оновлення для своєї продукції. Дуже часто нове

обладнання має застаріле вбудоване програмне забезпечення, яке не відповідає навіть мінімальним вимогам інформаційної безпеки.

Крім того, користувачі IoT не завжди приділяють увагу оновленню програмного забезпечення. Навіть при наявності оновлень у виробника, користувачі можуть їх просто не використовувати. При цьому, архітектура більшості IoT пристроїв дозволяє оновляти програмне забезпечення дистанційно, через бездротове з'єднання.

Основною перевагою сучасних IoT пристроїв є простота та низька собівартість. Саме це дозволяє зробити їх масовим та доступним інструментом, який можна застосувати для вирішення широкого спектру задач. Але з метою зниження собівартості виробники найчастіше йдуть шляхом використання морально застарілого або неякісного обладнання. Якщо для побутових систем це майже не помітно, у сфері промислових IoT призводить до суттєвого зниження рівня інформаційної безпеки та стабільності роботи інформаційної системи організації.

Використання IoT пристроїв на підприємствах має відбуватися тільки при відповідному тестуванні та перевірці якості програмного та апаратного забезпечення на стабільну роботу та відповідність стандартам інформаційної безпеки.

Згідно з новим законом "Про основні засади забезпечення кібербезпеки України", який набув чинності 9.05.2018, IoT не регламентується та не відокремлюється ні як пристрої, ні як комплекс програмно-апаратного забезпечення. Таким чином, на законодавчому рівні під IoT слід розуміти будь який пристрій, підключений до мережі Інтернет. Тобто, якщо пристрій IoT було використано зловмисниками для здійснення протизаконних дій, до відповідальності може бути притягнута особа, яка володіє цим пристроєм.

Найбільш розповсюдженою загрозою для IoT є використання пристроїв у якості ботнетів. Власник пристрою найчастіше не підозрює про те, що до програмного чи апаратного забезпечення вже було внесено зміни

зловмисниками, і у визначений час цей пристрій буде використано для здійснення DDOS-атаки або розсилання спам-повідомлень.

Основною причиною даної загрози є вразливість програмного та апаратного забезпечення. Саме забезпечення захисту IoT на сьогоднішній день майже ніяк не регламентується на законодавчому рівні в Україні. Придбання та використання пристроїв, які підключаються до мережі Інтернет не регулюється до того моменту, поки злочин не було скоєно з використанням цього пристрою. В той же час кожен пристрій має свій унікальний код та mac-адресу, які можуть однозначно ідентифікувати обладнання. При цьому найпростіші мікроконтролери, з яких на 99% складається мережа IoT, не мають власної операційної системи, що ускладнює зміну mac-адреси зловмисниками.

Таким чином, програмне та апаратне забезпечення сучасних пристроїв IoT дозволяє мінімізувати втручання зловмисників у користувацькі чи промислові мережі. Але ці можливості захисту майже ніколи не використовуються. З одного боку немає законодавчих вимог до впровадження заходів з боку користувача. З іншого боку, сам користувач пристрою IoT не завжди усвідомлює наслідки можливих дій зловмисників і не володіє інструментами інформаційного захисту.

Список використаної літератури

1. Соммер У. Программирование микроконтроллерных плат Arduino/ Freeduino. – СПб.: БХВ-Петербург, 2012. – 256 с.
2. Евстифеев А.В. Микроконтроллеры AVR семейства Mega. Руководство пользователя. – М.: Издательский дом “Додэка-XXI”, 2007. – 592 с.: ил. (Серия “Программируемые системы”).
3. Michael Margolis. Arduino Cookbook. – O'Reilly Media, 2011.
4. – 662 с.
5. Evans B. Arduino programming notebook [Електронний ресурс] / Brian W. Evans // First edition. – 2007. – Режим доступу до ресурсу: https://playground.arduino.cc/uploads/Main/ arduino_notebook_v1-1.pdf.
6. Іванченко Г. Ф. Прикладні системи штучного інтелекту. Навч.посібник. -К.:КНЕУ, 2014.-630 с. http://posibniki.com.ua/catalog-prikladni-sistemi-shtuchnogo-intelektu
7. Іванченко Г. Ф. Системи штучного інтелекту Навч.посібник. -К.:КНЕУ, 2011.-382 с. http://programming.in.ua/programming/basisprogramming/330-ivanchenko-systems-of-artificial-intelligence.html
8. Гавриленко В.В.,Іванченко Г.Ф., Шевченко Г.Є. Теорія розпізнавання образів. Національний Транспортний Університет.,К. НТУ 2015.- 76с.
9. Бондарев В.Н., Аде Ф.Г. Искусственный интеллект. Учебное пособие для вузов. -Севастополь, Изд-во СевНТУ, 2002. -615 с.
10. Искусственный интеллект. Справочник. - В 3-х томах. - М.: Радио и связь, 1990
11. https://cyberleninka.ru
12. https://www.kpi.kharkov.ua
13. http://financial.lnu.edu.ua/wp-content/uploads/2015
14. http://www.zgia.zp.ua/index.php?page

ЗМІСТ

Навчальне видання

Густера О.М.

КОНСПЕКТ ЛЕКЦІЙ

з дисципліни «Інтернет речей»

за спеціальністю 051 «Економіка», освітньо-професійною програмою:
«Цифрова економіка»