DOI: 10.18372/2225-5036.31.20639

ВРАЗЛИВОСТІ АРХІТЕКТУРИ IOT-МЕРЕЖ: КЛАСИФІКАЦІЯ ТА РЕАЛЬНІ ІНЦИДЕНТИ

Даніель Пастущак

Державний університет «Київський авіаційний інститут»

ПАСТУЩАК Даніель Васильович
Рік та місце народження: 1998 р., м. Бурштин, Україна
Освіта: Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», магістр з кібербезпеки, 2020
Посада: аспірант кафедри кібербезпеки Державного університету «Київський авіаційний інститут» (з 2024 р.)
Наукові інтереси: кібербезпека, аналіз великих даних, виявлення аномалій в IoT-мережах, потокова обробка даних
E-mail: danikpastushchak90@gmail.com
ORCID: 0009-0008-9120-8291

Анотація. *Стрімке поширення Інтернету речей (IoT) призводить до зростання кількості пристроїв, інтегрованих у критичну інфраструктуру, промисловість та повсякденне життя. Водночас низькі обчислювальні можливості, різнорідність протоколів і відсутність належних механізмів оновлення роблять IoT-екосистеми вразливими до широкого спектра атак. У статті проведено систематизацію основних категорій вразливостей IoT, включаючи технічні обмеження пристроїв, протокольні недоліки, використання стандартних налаштувань, фізичний доступ до обладнання та організаційні чинники. Окрему увагу приділено аналізу слабких місць у комунікаційних протоколах (MQTT, HTTP, CoAP), а також описано найпоширеніші інциденти, зокрема ботнет Mirai та атаки на промислові системи безпеки Triton і CrashOverride. Результати дослідження демонструють, що вразливості існують на всіх рівнях архітектури IoT-мереж і навіть одна слабка ланка може призвести до масштабних наслідків. Представлена класифікація та приклади реальних атак можуть бути використані для формування ефективних стратегій захисту IoT-систем та подальшого розвитку засобів кібербезпеки.*

Ключові слова*: IoT, вразливості, протоколи зв'язку, кібербезпека, Mirai, промислові атаки, критична інфраструктура.*

Постановка проблеми

Розвиток технологій Інтернету речей (IoT) сприяє масштабному впровадженню інтелектуальних сенсорних пристроїв у промисловість, енергетику, транспорт і побут. Разом із перевагами підвищення ефективності та автоматизації виникають нові ризики для кібербезпеки. Обмежені обчислювальні ресурси вузлів, відсутність уніфікованих стандартів захисту, використання небезпечних протоколів зв'язку та дефолтних налаштувань створюють умови для зловмисного використання IoT-систем. Наявність фізичного доступу до пристроїв дозволяє здійснювати апаратні атаки, а слабкий рівень організаційних заходів безпеки підвищує ймовірність експлуатації відомих вразливостей протягом тривалого часу. Відомі інциденти, зокрема ботнет Mirai та атаки Triton і CrashOverride, підтверджують, що навіть одна скомпрометована ланка в IoT-екосистемі може призвести до масштабних наслідків для критичної інфраструктури. Це визначає необхідність систематизації та аналізу вразливостей IoT з метою формування ефективних стратегій їхнього виявлення та усунення.

Аналіз останніх досліджень і публікацій

Дослідження з безпеки IoT зосереджені на розвитку систем виявлення вторгнень, які використовують машинне та глибинне навчання. Зокрема, інтеграція методів для протоколу MQTT підвищує точність детекції [1], ансамблеві моделі забезпечують стійкість у гетерогенних мережах [2], а few-shot learning дозволяє працювати за браку даних [3]. Для сценаріїв із підвищеними вимогами до конфіденційності застосовують федеративне навчання та fog-архітектури [4], тоді як енергоефективні механізми поєднують криптографію з оптимізацією ресурсів [5].

Суттєвим викликом залишаються масові атаки на кшталт Mirai, що експлуатують дефолтні паролі й слабке шифрування [6], а також wormhole-атаки у сенсорних мережах [7]. Огляди з безпеки та приватності підкреслюють, що уразливості HTTP, MQTT, Zigbee, LoRaWAN, BLE та протоколів нульової конфігурації залишаються поширеним джерелом ризику [11]–[13].

Водночас актуальними є фізичні атаки: доступ через UART/JTAG, побічні канали чи ін'єкція помилок дозволяють обійти захист на рівні прошивки [8]. Практичні кейси показують, що поєднання дефолтних налаштувань, протокольних помилок і фізичної доступності робить IoT-мережі вразливими до серйозних інцидентів [9]. Таким чином, останні публікації вказують на необхідність комплексної стратегії, що поєднує інтелектуальні IDS, фог-обчислення, енергоефективну криптографію та врахування апаратних векторів атак [1]–[13].

Сучасні IoT-мережі дедалі частіше інтегруються у критичні сфери — енергетику, транспорт, медицину та виробництво. Зростання масштабів їхнього використання супроводжується появою нових кіберзагроз, які пов'язані як із технічними

обмеженнями пристроїв, так і з уразливостями протоколів та організаційними недоліками. Це зумовлює необхідність не лише локальних рішень, а й системного аналізу архітектури IoT з метою виявлення ключових вразливостей та узагальнення практичного досвіду атак.

Мета та постановка завдання

Метою даної роботи є систематизація вразливостей архітектури IoT-мереж, їхня класифікація за рівнями та визначення наслідків для безпеки. У роботі також розглядаються приклади реальних інцидентів, які демонструють практичну значущість проблеми, та пропонуються напрями майбутніх досліджень, що мають сприяти зниженню ризиків для споживчих і промислових систем.

Для досягнення цієї мети необхідно розв'язати низку завдань. По-перше, потрібно дослідити технічні обмеження IoT-пристроїв, включаючи недостатні обчислювальні ресурси, обмежений обсяг пам'яті, труднощі із впровадженням сучасної криптографії та проблеми з оновленням прошивки. По-друге, варто проаналізувати недоліки комунікаційних протоколів: використання незашифрованого HTTP, відсутність TLS у MQTT, слабкі механізми автентифікації в Zigbee, LoRaWAN та BLE, а також ризики від протоколів нульової конфігурації, які відкривають додаткові вектори атак.

Третім завданням є узагальнення загроз, пов'язаних із дефолтними налаштуваннями, що досі залишаються поширеними: використання заводських паролів, вбудованих сервісних акаунтів у прошивці чи відсутність базового шифрування. Саме ці чинники стали причиною формування ботнетів на кшталт Mirai, які призвели до глобальних DDoS-інцидентів. По-четверте, слід розглянути фізичні та апаратні вектори атак: доступ через UART та JTAG, побічні канали та методи ін'єкції помилок (glitching), які дозволяють обходити перевірку підписів та отримувати контроль над прошивкою пристроїв.

Для підтвердження серйозності проблеми важливо проаналізувати приклади реальних інцидентів, серед яких Mirai, Triton та CrashOverride. Ці атаки показали, що поєднання технічних, протокольних і організаційних недоліків може призвести до масштабних наслідків як у споживчому сегменті (IoT-камери, «розумні» замки), так і у промисловості, де під загрозою опинилися енергосистеми та безпекові контролери.

Таким чином, завдання дослідження полягає у створенні систематизованої картини вразливостей IoT, яка дозволяє не лише виявити ключові проблеми архітектури, але й сформувати підґрунтя для майбутніх рішень. До таких рішень належать інтеграція інтелектуальних IDS на основі машинного навчання, використання фог-обчислень для локальної обробки даних, застосування енергоефективної криптографії та врахування апаратних ризиків у процесі розробки. У комплексі це має сприяти створенню більш захищених і стійких IoT-середовищ, здатних витримати як масові, так і таргетовані атаки.

Виклад основного матеріалу дослідження

1. Технічні обмеження IoT-пристроїв (ресурси, прошивка)

Розгортання IoT часто базується на пристроях, які є компактними, енергоефективними та недорогими. Досягнення цих апаратних та бюджетних цілей неминуче обмежує потужність ЦП, обсяг пам'яті та інші ресурси, що призводить до спрощених або неповних засобів безпеки. Окрім обмежень ресурсів, сама прошивка може стати значущою слабкістю, якщо вона не розробляється або не оновлюється відповідно до передових стандартів безпеки. Нижче детально описано, як ці технічні "вузькі місця" створюють і часом посилюють вразливості IoT [7][4].

1.1. Обмеження ЦП/пам'яті та виклики шифрування

Легкі апаратні профілі. Багато IoT-пристроїв використовують компоненти *System-on-Chip (SoC)*, які працюють на низьких тактових частотах (наприклад, десятки або сотні МГц) з декількома десятками або сотнями кілобайт оперативної пам'яті. Хоча такі рішення є економічно вигідними та енергоефективними, мінімальні апаратні можливості не сприяють використанню потужного шифрування (наприклад, RSA на 2048 біт або ECC-256), оскільки пристрій може мати труднощі з обчисленнями під час встановлення з'єднання. В результаті виробники часто обирають слабші алгоритми або скорочують довжину ключів, що робить пристрій вразливим до криптоаналізу або атак методом "грубої сили" [8].

Компроміс заради продуктивності. Навіть за наявності апаратного прискорення для криптографії, багато старих або нижчого класу IoT-плат може його не мати. Внаслідок цього спроби виконання сучасних алгоритмів — таких як AES-256-GCM або алгоритм обміну ключами на основі кривих Еліптичних (ECDH) — призводять до помітних затримок, що робить реальне збори даних з сенсорів або їхнє керування повільним [5]. Розробники іноді вимикають цілі шари безпеки або зменшують частоту ротації ключів. У контексті АСУ ТП вимоги до реального часу можуть зробити навіть односекундну затримку неприйнятною, тому виправлення для шифрування на рівні протоколів можуть відкладатися на невизначений час [4].

Обмеження пам'яті. Обмежена кількість оперативної пам'яті може заважати реалізації надійної логіки виявлення вторгнень або локального журналювання. Якщо пристрій не може зберегти повний аудит подій або підтримувати достатню кількість станів сесії, аномалії можуть залишатися незареєстрованими [3]. Ці ж проблеми з пам'яттю можуть ускладнювати процес TLS-переговорів, який часто вимагає зберігання сертифікатів, тимчасових

ключів та станів встановлення з'єднання. Через це деякі IoT-системи продовжують працювати на незашифрованому HTTP або покладаються на попередньо встановлені спільні ключі (PSK), які рідко змінюються [4].

1.2. Розробка прошивки та механізми оновлення

Монолітна проти модульної прошивки. Прошивка для IoT часто представлена як один єдиний монолітний бінарний образ, записаний у флеш-пам'ять. Такий підхід спрощує розробку, але ускладнює оновлення – навіть невеликі зміни вимагають перепрошивки всього образу [8]. Виробники часто вбудовують сторонні бібліотеки (наприклад, мережеві стеки, драйвери сенсорів) з відомими вразливостями, які можуть залишатися невиправленими протягом багатьох років. Зловмисники, виявивши вразливості в цих бібліотеках, можуть використовувати їх послідовно у багатьох продуктових лініях [5].

OTA (безпровідові) оновлення. Безпечні та зручні OTA-оновлення вважаються критично важливими для захисту IoT-пристроїв. Проте багато реальних IoT-продуктів не підтримують OTA, і оператори змушені фізично підключати пристрій через спеціалізований серійний або USB-порт для перепрошивки. Якщо власники не оновлюють ці пристрої, особливо у великих географічно розподілених мережах, відомі вразливості залишаються невиправленими. Навіть коли OTA підтримується, деякі реалізації пропускають перевірку підпису або використовують один і той же жорстко закодований ключ для всіх продуктів. Компрометація такого ключа дає змогу здійснювати масштабну дистрибуцію шкідливої прошивки [8].

Підписана прошивка та відкат оновлень. Перевірка підпису забезпечує запуск на пристрої лише тієї прошивки, яку затверджено виробником, що зменшує ризик маніпуляцій . Однак, якщо цей крок є необов'язковим або його можна легко обійти через відладочний інтерфейс, зловмисник може понизити версію пристрою до менш безпечної. Крім того, деякі системи впроваджують механізм відкату, який дозволяє повернутися до попередньої, перевіреної версії прошивки, якщо оновлення виявилося несправним. Проте ця логіка відкату може бути використана зловмисниками для повернення до вразливої версії, якщо вона не захищена належним чином [12].

Механізми апаратного блокування. Деякі мікроконтролери підтримують біти захисту пам'яті, що дозволяють заблокувати область флеш-пам'яті після початкового програмування [3]. Ці механізми можуть ускладнити відладку або судову експертизу, але значно підвищують рівень безпеки, ускладнюючи зловмисникам витягування прошивки або впровадження шкідливого коду. Якщо інтегратори не налаштовують ці біти або існують методи їх обходу (наприклад, за допомогою voltage glitching), прошивка пристрою залишається вразливою для викрадення та маніпуляцій [8].

1.3. Фізична безпека: відлагоджувальні порти, атаки за допомогою побічних каналів, ін'єкція помилок

Доступні відлагоджувальні інтерфейси. Більшість IoT-плат має відладочні порти, такі як UART, JTAG або SWD, що полегшують тестування під час розробки. У фінальному виробництві ці порти повинні бути вимкнені або захищені, але часто вони залишаються відкритими [8]. Зловмисники, отримавши фізичний доступ до пристрою, можуть підключити стандартний USB-to-UART кабель, видати root-команди або зчитати пам'ять для отримання секретів (наприклад, ключів шифрування або збережених облікових даних) [10]. Крім того, зловмисники можуть перепрошити пристрій, встановлюючи власну прошивку, яка ігнорує засоби безпеки.

Атаки за допомогою побічних каналів. Навіть якщо код пристрою захищено, побічні сигнали — такі як електромагнітні випромінювання або коливання струму — можуть розкрити інформацію про криптографічні операції або секретні ключі. Інструменти, такі як диференціальний аналіз споживання потужності (DPA), вимірюють незначні коливання струму під час шифрування, що дозволяє досвідченим зловмисникам реконструювати ключ. Деякі IoT-SoC надають апаратні засоби протидії (рандомізований час виконання, маскування S-box), але багато бюджетних пристроїв обходять ці засоби, [6].

Ін'єкція помилок (Voltage/Clock Glitching). Метод ін'єкції помилок полягає у навмисному створенні збоїв у подачі напруги або годинникового сигналу пристрою з метою пропустити певну інструкцію або змінити логіку виконання, що дозволяє обійти критичні перевірки безпеки. Ретельно сплановані voltage glitch можуть, наприклад, обійти перевірку підпису завантажувача або пропустити перевірку пароля в прошивці. Колись для цього потрібне було дороге лабораторне обладнання, але завдяки відкритим проектам методи стали доступнішими навіть для зловмисників з обмеженим бюджетом [8].

Таблиця 1

Технічні обмеження та апаратні вразливості IoT.

Категорія	Короткий опис
Обмеження ЦП/пам'яті	Слабкі SoC, обмежена пам'ять, неможливість використання сильного шифрування
Компроміс продуктивність/безпека	Вимкнення криптографії через затримки, повільна ротація ключів
Прошивка та оновлення	Монолітні образи, відсутність OTA, ризики rollback-атак
Відлагоджувальні порти	UART/JTAG відкриті у серійних пристроях, легкий фізичний доступ
Побічні канали	Аналіз споживання чи EM-випромінювання дозволяє викрасти ключі
Ін'єкція помилок	Voltage/clock glitching обходить перевірку підписів та автентифікацію

2. Недоліки протоколів зв'язку (HTTP, MQTT тощо)

Велика частина функціональності IoT базується на мережевій комунікації, яка забезпечує обмін даними між сенсорами, актуаторами, проміжними шлюзами та, в кінцевому рахунку, хмарою [7]. Проте багато з найпоширеніших IoT-протоколів — як загального призначення (наприклад, HTTP), так і спеціалізованих для обмежених пристроїв (наприклад, MQTT) — були спроектовані більше з огляду на простоту використання та низькі накладні витрати, аніж з урахуванням надійної безпеки "з коробки" [2], [5]. В результаті типовий трафік IoT може бути схильним до таких проблем, як перехоплення, повторне відтворення (replay), помилки налаштування брокера чи використання паролів за замовчуванням, що відкриває можливості для масштабових атак.

2.1. Незашифрований HTTP, атаки повторного відтворення та викрадення сесії

HTTP проти HTTPS. Оскільки багато IoT-пристроїв мають обмежені ресурси ЦП та пам'яті, або виробники вважають, що "ніхто не зверне уваги на цей трафік", значна частина вбудованих рішень досі використовує незашифрований HTTP для передачі даних або оновлень. Це означає, що облікові дані користувачів, токени сесії та дані сенсорів передаються мережею у відкритому вигляді. Атакуючий може з легкістю перехопити ці пакети (за допомогою Wireshark або tcpdump) у будь-якій спільній або локальній мережі, щоб отримати облікові дані або змінити дані [4]. У більш просунутих ICS-середовищах, якщо пристрій використовує застарілі криптографічні алгоритми або невірно перевірені самопідписані сертифікати, ефект буде майже таким же, як і відсутність шифрування.

Атаки повторного відтворення та викрадення сесії. Навіть якщо платформа використовує якийсь тип аутентифікації на основі токенів (наприклад, cookie-сесії або заголовок Basic Auth) через незашифрований HTTP, атакуючий, який перехоплює трафік, може захопити цей токен і використати його в наступних запитах. Наприклад, якщо IoT-датчик температури періодично викликає REST-ендоінт для отримання нових порогів, атакуючий, отримавши цей токен, може змусити систему встановити небезпечні пороги, що може призвести до саботажу — наприклад, до відключення системи охолодження на виробничій лінії [8]. Якщо на сервері не впроваджено додаткових перевірок (на основі часу або nonce), такі атаки стають цілком здійсненними.

HTTP/2 та нові підходи. Хоча HTTP/2 теоретично може прискорити передачу даних, проблеми безпеки залишаються, якщо базове з'єднання не зашифроване належним чином (HTTP/2 через TLS або "h2"). Деякі IoT-рішення впроваджують часткове шифрування, наприклад, застосовують TLS лише для певних ендпоінтів або лише для обміну обліковими даними. Але неповне покриття або ігнорування перевірки сертифікатів (наприклад, невірна перевірка Common Name або використання закріплених публічних ключів) знову відкриває двері для атак "людина посередині" [7].

2.2. MQTT: Помилки налаштування брокера та слабкі облікові дані

Модель публікації/підписки. MQTT набув великої популярності в IoT завдяки своїм мінімальним накладним витратам і можливості гнучкого налаштування схем публікації та підписки. Пристрої (публікатори) надсилають дані (наприклад, значення сенсорів) за певними темами, тоді як інші (підписники) отримують повідомлення, що відповідають цим темам. Центральною ланкою моделі є брокер MQTT, який координує всі повідомлення. Атаки зазвичай зосереджені на проникненні в брокер або імітації клієнтів [4].

Налаштування та передача у відкритому вигляді. Багато реальних впроваджень працюють на стандартному порту MQTT 1883 без використання TLS і без автентифікації, що практично означає довіру до локальної мережі. Атакуючі, перебуваючи в тій же мережі, можуть підписатися на будь-яку тему, отримувати дані з усієї мережі або надсилати команди, видаючи себе за легітимний пристрій (наприклад, "pump/stop" або "valve/open"). Навіть якщо використовується базова автентифікація (ім'я користувача/пароль), спільний ключ може бути вбудований у прошивку всіх пристроїв, і його викрадення з одного пристрою дозволяє імітувати інші [5].

Слабкі облікові дані. Поширеною проблемою є використання за замовчуванням або легко вгадуваних логінів для брокера MQTT. Деякі системи зберігають облікові дані у конфігураційних файлах шлюзу, або взагалі не ведуть журналів спроб злому, що дозволяє зловмисникам за допомогою скриптів проводити атаки методом словника, систематично знаходячи валідних користувачів або вводячи команди, що порушують логіку управління. Оскільки MQTT зазвичай не має вбудованого контролю доступу, після проникнення атакуючий може читати або писати дані до будь-якої теми [7].

2.2.1 Захист MQTT

Рекомендується використовувати MQTT з TLS (на порту 8883) разом з унікальними обліковими даними для кожного пристрою, можливо, навіть з клієнтськими сертифікатами TLS [9]. Проте обмеження пам'яті малих сенсорів ускладнюють надійне зберігання сертифікатів, тому багато виробників повертаються до простіших попередньо встановлених ключів або взагалі відмовляються від шифрування [3]. Деякі просунуті брокери впроваджують контроль доступу на основі ролей (RBAC), обмежуючи, які пристрої можуть публікувати або підписуватися на певні теми, але ці

рішення можуть бути важкими в налаштуванні у великих, гетерогенних мережах [11].

2.3. Спеціалізовані радіопротоколи: Zigbee, LoRa, BLE та їхні слабкі місця

Zigbee широко використовується для низькоенергетичних мереж персональної зони (PAN), наприклад, для керування розумним освітленням або в промислових сенсорних мережах [2], [5]. Хоча Zigbee підтримує шифрування на рівні каналу та опціональні засоби безпеки, багато впроваджень покладаються на один заводський ключ мережі. Атакуючий, перехопивши цей ключ з одного пристрою або під час процесу сполучення, може приєднатися до мережі або порушити її роботу. Також можливо повторне відтворення застарілих пакетів, якщо механізми захисту від replay не впроваджені.

LoRaWAN призначений для передачі невеликої кількості даних на великі відстані (до десятків кілометрів), що робить його особливо корисним для розумного сільського господарства, міських мереж і віддалених промислових трубопроводів [6]. Він використовує сесійні ключі для шифрування даних на рівні пристроїв (AppSKey, NwkSKey), але залежить від надійного розподілу ключів на заводі чи під час доставки. Якщо пам'ять пристрою доступна для читання, атакуючий може отримати ці ключі і перехоплювати або вводити дані через будь-який шлюз LoRa, що використовує ті ж мережеві облікові дані [6]. Недостатня випадковість або часте повторне використання nonce для генерації сесійних ключів послаблюють криптографічний захист.

Bluetooth Low Energy (BLE) забезпечує короткодіапазонні з'єднання, використовувані у носимих пристроях для охорони здоров'я або маяках у системах автоматизації будівель. Існує декілька режимів сполучення, але популярним залишається режим "Just Works", який не вимагає використання паролів. Атакуючі в межах зони дії BLE можуть перехоплювати процес сполучення або обманом змусити пристрій встановити нове з'єднання, фактично захоплюючи канал передачі даних . Якщо характеристики GATT пристрою не обмежені, зловмисник може безпосередньо зчитувати або змінювати критично важливі параметри (наприклад, швидкість двигуна або пороги сенсора), обходячи логіку програмного забезпечення [8].

Всі ці радіопротоколи також вразливі до атак типу **jamming** (коли зловмисники засмічують частоту шумом) або **replay** (захоплення дійсного пакету та повторне його транслювання), якщо не впроваджено надійних перевірок на основі nonce або сесійних ключів [3]. У децентралізованих mesh-мережах повторна трансляція старих повідомлень може спричинити конфліктні стани або навіть створити мережевий шторм. Без апаратних засобів протидії jamming або лічильників повідомлень, атаки на рівні радіозв'язку можуть стати досить простими.

2.4. Протоколи нульової конфігурації (UPnP, mDNS)

Universal Plug and Play (UPnP) було розроблено для спрощення виявлення пристроїв у локальних мережах, дозволяючи новим гаджетам (IP-камери, принтери) автоматично відкривати порти на міжмережевих екранах або оголошувати свої сервіси. У споживчому IoT багато маршрутизаторів через помилки налаштувань або випадковості відкривають UPnP для зовнішнього доступу, що дозволяє зловмисникам створювати NAT-мапінги та фактично обходити захист мережі [2], [4]. Зловмисники також можуть перенаправляти пристрої на підроблені сервери оновлень або змінювати налаштування DNS віддалено.

Multicast DNS (mDNS) та DNS Service Discovery (DNS-SD, часто відомий як Bonjour від Apple) використовуються для зрозумілого людьми найменування пристроїв та оголошення їхніх сервісів у локальних підмережах [9]. Атакуючі можуть транслювати шкідливі оголошення, змушуючи пристрої з'єднуватися з підробленими сервісами або підробляти записи про пристрої, що ускладнює контроль користувача. Зламаний пристрій також може відповідати на запити mDNS, обманюючи інші хости та змушуючи їх відправляти облікові дані на IP-адресу зловмисника. Оскільки ці протоколи зазвичай працюють через широкомовну або мультикастову передачу з мінімальною автентифікацією, їх підробка або захоплення стає досить простою.

Протоколи нульової конфігурації, хоч і зручні, дозволяють новим або заміненим пристроям автоматично інтегруватися в локальну мережу. Якщо зловмиснику вдасться вставити підробний пристрій або просто ввести фальшиві оголошення, він може порушити роботу всієї локальної IoT-мережі. Відсутність "підписаних оголошень" або надійної перевірки власності означає, що мережа автоматично довіряє будь-якому пристрою, який з'являється. З часом, різні виробники впроваджують власні або часткові версії UPnP/mDNS, що ускладнює застосування єдиних заходів захисту або оновлень безпеки [4].

Таблиця 2

Недоліки протоколів зв'язку в IoT

Протокол	**Основні вразливості**
HTTP	Передача у відкритому вигляді; легке перехоплення даних; вразливість до replay-атак і крадіжки сесій
MQTT	Відсутність TLS; брокери без автентифікації; дефолтні логіни/паролі; можливість підпису на будь-яку тему
Zigbee	Використання заводського ключа; відсутність захисту від replay; компрометація під час сполучення
LoRaWAN	Слабкий розподіл ключів; повторне використання nonce; доступ до ключів у пам'яті пристрою
BLE	Режим «Just Works» без PIN; можливість перехоплення або підробки з'єднання
UPnP/mDNS	Автоматичне відкриття портів; підробка оголошень; можливість редиректу на шкідливі сервіси

3. Реальні інциденти

Ботнет Mirai є найвідомішим прикладом того, як зловмисники використовують стандартні облікові дані для захоплення безлічі IP-камер і маршрутизаторів [6]. Програма систематично сканувала діапазони IP-адрес і спробувала відомі пари "логін:пароль" — наприклад, "admin:admin" або "root:1234" — через сервіси Telnet або SSH. Потрапивши в пристрій, вона змушувала його приєднатися до розподіленої мережі для проведення DDoS-атак. Хоча багато постачальників виправили цю проблему, рандомізуючи облікові дані або відключаючи Telnet, клоновані варіанти та їх наступники (наприклад, Gafgyt, Bashlite) все ще націлюються на нові або незабезпечені системи [5]. Приклад з ботнетом Mirai ілюструє, наскільки легко можна залучити десятки тисяч IoT-пристроїв, просто скануючи мережу на предмет сервісів Telnet з типовими заводськими налаштуваннями [6]. Після компрометації ці пристрої використовувалися для запуску масштабних DDoS-атак, що призвели до збоїв у роботі основних інтернет-сервісів. Основна вразливість — незмінені заводські паролі — залишається поширеною, що свідчить про мінімальні покращення в деяких секторах IoT-екосистеми [7].

У контексті АСУ ТП або промислового IoT цільові кампанії, такі як Triton/Trisis або CrashOverride, демонструють, як зловмисники експлуатують незахищені сервіси пристроїв або залишки відлагоджувальних облікових записів для маніпуляції системами безпеки (SIS), що може призвести до катастрофічних пошкоджень. Навіть за наявності часткового шифрування або автентифікації, один незахищений пристрій або стандартний пароль може стати відправною точкою для подальшого ескалації контролю. Деякі зловмисники отримують початковий доступ, скануючи наявні вразливості у інструментарії постачальників АСУ ТП, а потім знаходять стандартні облікові дані, що відкривають подальші точки контролю [4].

У споживчій сфері численні випадки зламу дитячих моніторів або "розумних" замків підкреслюють ризики, пов'язані з байдужістю користувачів до зміни заводських налаштувань. Зловмисники можуть перехоплювати трафік або відкривати двері, використовуючи широко розповсюджені стандартні коди або мінімальні облікові дані, передані через незашифровані Wi-Fi-з'єднання. Це не лише створює ризик викрадення або домагань, але й підірває довіру громадськості до можливостей IoT [12].

Висновки

Проведений аналіз показав, що архітектура IoT-мереж має системні вразливості, які виникають на всіх рівнях — від апаратних ресурсів до протоколів і організаційних практик. Обмежена потужність процесорів і мала кількість пам'яті змушують виробників жертвувати криптографічною стійкістю, що робить пристрої вразливими до атак грубої сили та криптоаналізу. Монолітні прошивки без механізмів OTA та використання спільних ключів спричиняють тривалу експозицію відомих вразливостей і створюють умови для масового поширення шкідливих образів.

Протоколи зв'язку, що лежать в основі IoT, у більшості випадків проєктувалися з орієнтацією на простоту й ефективність, а не на безпеку. Використання незашифрованого HTTP, відкритих MQTT-брокерів, заводських ключів у Zigbee або режиму «Just Works» у BLE формують широкий спектр можливостей для атак типу перехоплення, підміни чи повторного надсилання. Протоколи нульової конфігурації, зручні для інтеграції, водночас дозволяють легко підключати підроблені пристрої, ускладнюючи контроль цілісності мережі.

Не менш значущими є загрози, пов'язані з дефолтними налаштуваннями. Використання заводських паролів і вбудованих акаунтів залишається поширеною практикою, що вже призвело до масштабних атак, як у випадку з ботнетом Mirai. Ці фактори, у поєднанні з недоліками оновлень і протоколів, створюють сприятливе середовище для формування нових шкідливих кампаній.

Фізичний доступ і апаратні атаки доповнюють картину: відкриті UART/JTAG-порти, побічні канали та методи ін'єкції помилок дозволяють зловмисникам обходити програмні засоби захисту, зчитувати прошивку і ключі, модифікувати логіку пристроїв.

Розгляд реальних інцидентів — Mirai, Triton, CrashOverride — підтвердив, що уразливості IoT мають не теоретичний, а практичний вимір і можуть призводити до масштабних наслідків як у споживчому сегменті, так і в критичній інфраструктурі.

Таким чином, IoT-мережі потребують комплексного підходу до захисту, що включає інтеграцію інтелектуальних систем виявлення вторгнень, використання фог-обчислень для локальної обробки даних, впровадження енергоефективної криптографії та системне врахування апаратних ризиків на етапі проєктування. Лише багаторівнева стратегія, що поєднує технічні, протокольні та організаційні рішення, здатна забезпечити належний рівень захищеності та стійкості IoT у сучасних умовах.

Список літератури

[1] Michelena Á., García-Ordás M.T., Aveleira-Mata J., Yereguí Marcos D., Timiraos Díaz M., Zayas-Gato F., Jove E., Casteleiro-Roca J.L., Quintián H., Alaiz-Moretón H., Calvo-Rolle J.L., Beta Hebbian Learning for intrusion detection in networks with MQTT protocols for IoT devices, *Journal of Logic and Computation*, Oxford University Press, 2024, Vol. 32, No. 2, pp. 352–374.

[2] Jeffrey N., Tan Q., Villar J.R., Using Ensemble Learning for Anomaly Detection in Cyber–Physical Systems, *Electronics*, MDPI, 2024, Vol. 13, No. 7, Article 1391.

[3] Althiyabi T., Ahmad I., Alassafi M.O., Enhancing IoT Security: A Few-Shot Learning Approach for Intrusion Detection, *Mathematics*, MDPI, 2024, Vol. 12, No. 7, Article 1055.

[4] Wakili A., Bakkali S., Privacy-preserving security of IoT networks: A comparative analysis of methods and applications, *Cyber Security and Applications*, KeAi Publishing, 2025, Vol. 3, Article 100084.

[5] Iqbal F., Ahmed S., Tariq M.A.B., Waqas H.A., Al-Ammar E.A., Wabaidur S.M., A Survey on Energy-Aware Security Mechanisms for the Internet of Things, *Applied Sciences*, MDPI, 2024, Vol. 14, No. 1, Article 499.

[6] Lefoane M., Ghafir I., Kabir S., Awan I.-U., Internet of Things botnets: A survey on artificial intelligence based detection techniques, *Journal of Network and Computer Applications*, Elsevier, 2025, Vol. 236, Article 104110.

[7] Abdullah A., Albaihani A.N.A., Osman B., Omar Y., Detecting Wormhole Attack in Environmental Monitoring System for Agriculture using Deep Learning, *Journal of Advanced Research in Applied Sciences and Engineering Technology*, 2025, Vol. 51, No. 2, pp. 153–176.

[8] Van Woudenberg J., O'Flynn C., *The Hardware Hacking Handbook: Breaking Embedded Security with Hardware Attacks*, No Starch Press, San Francisco, 2022, 486 p.

[9] Chantzis F., Stais I., Calderón P., Deirmentzoglou E., Woods B., *Practical IoT Hacking: The Definitive Guide to Attacking the Internet of Things*, No Starch Press, San Francisco, 2021, 464 p.

[10] Alrawais A., Alhothaily A., Hu C., Cheng X., Fog Computing for the Internet of Things: Security and Privacy Issues, *IEEE Internet Computing*, IEEE, 2017, Vol. 21, No. 2, pp. 34–42.

[11] Sicari S., Rizzardi A., Grieco L.A., Coen-Porisini A., Security, Privacy and Trust in Internet of Things: The Road Ahead, *Computer Networks*, Elsevier, 2015, Vol. 76, pp. 146–164.

[12] Lin J., Yu W., Zhang N., Yang X., Zhang H., Zhao W., A Survey on Internet of Things: Architecture, Enabling Technologies, Security and Privacy, and Applications, *IEEE Internet of Things Journal*, IEEE, 2017, Vol. 4, No. 5, pp. 1125–1142.

[13] Humayed A., Lin J., Li F., Luo B., Cyber–Physical Systems Security: A Survey, *IEEE Internet of Things Journal*, IEEE, 2017, Vol. 4, No. 6, pp. 1802–1831.

УДК 004.056.5(045)

Pastushchak D. Vulnerabilities of IoT Network Architectures: Classification and Real Incidents

Abstract. The rapid expansion of the Internet of Things (IoT) has resulted in a growing number of devices integrated into critical infrastructure, industry, and everyday life. At the same time, limited computational resources, protocol heterogeneity, and the lack of proper update mechanisms make IoT ecosystems vulnerable to a wide range of attacks. This article systematizes the main categories of IoT vulnerabilities, including device limitations, protocol weaknesses, default configurations, physical access, and organizational factors. Special attention is paid to the analysis of communication protocol flaws (MQTT, HTTP, CoAP) and the description of common incidents, such as the Mirai botnet and industrial safety system attacks Triton and CrashOverride. The results show that vulnerabilities exist at all levels of IoT network architecture, and even a single weakness can lead to large-scale consequences. The presented classification and real-world attack cases can be applied to the development of effective IoT protection strategies and further advancement of cybersecurity solutions.

Keywords: IoT, vulnerabilities, communication protocols, cybersecurity, Mirai, industrial attacks, critical infrastructure.

Пастущак Даніель Васильович, аспірант кафедри кібербезпеки Державний університет: «Київський авіаційний інститут».
Pastushchak Daniel, Ph.D Student at the Department of Cybersecurity State University «Kyiv Aviation Institute».