УДК 004.7.056.55:621.396.7 **DOI:** 10.31673/2412-9070.2026.051907

А. Д. ІЩЕНКО, аспірант;
ORCID: 0009-0000-5161-7818
В. І. ОРЄШКОВ, канд. техн. наук, доцент,
ORCID: 0000-0001-9796-0216
Державний університет інтелектуальних технологій і зв'язку, Одеса

РОЗРОБКА ЛЕГКОВАГОВОГО ПРОТОКОЛУ АВТЕНТИФІКАЦІЇ ТА ДИНАМІЧНОГО РОЗПОДІЛУ КЛЮЧІВ ДЛЯ БЕЗДРОТОВИХ СЕНСОРНИХ МЕРЕЖ З ТОПОЛОГІЄЮ MESH

У статті запропоновано легковаговий протокол автентифікації та динамічного розподілу ключів для бездротових сенсорних мереж з топологією Mesh. Розроблено математичну модель наскрізного шифрування, яка мінімізує енергетичні витрати криптографічних операцій шляхом оптимізації примітивів на основі алгоритму ECDH та хеш-функцій SHA-256. Основою практичної реалізації є брокер повідомлень Mosquitto з плагіном автентифікації mosquitto-auth-plugin, що забезпечує контроль доступу ACL через MQTT-протокол. Аналітичні оцінки показують зменшення енергоспоживання до 62% порівняно з класичними схемами TLS при збереженні криптографічної стійкості на рівні 128 біт. Ефективність підходу підтверджена імітаційним моделюванням на типових конфігураціях Mesh-мереж.

Ключові слова: бездротові сенсорні мережі, Mesh-топологія, MQTT, автентифікація, ACL, розподіл ключів, ECDH, енергоефективна криптографія.

Вступ

Бездротові сенсорні мережі (БСМ) з Mesh-топологією широко застосовуються в системах «Інтернету речей» (IoT), промисловій автоматизації, «розумних» містах та медичному моніторингу. Децентралізована природа Mesh-архітектури забезпечує відмовостійкість і масштабованість, проте водночас породжує суттєві виклики у сфері інформаційної безпеки [1].

Зростання кількості розгорнутих бездротових сенсорних мереж у промисловій автоматизації, системах моніторингу довкілля та «розумних» містах загострює протиріччя між вимогами до інформаційної безпеки та фізичними обмеженнями сенсорних вузлів. Вузол із мікроконтролером класу ARM Cortex-M4 або ESP32 має батарею ємністю 500-2000 мАг, яку в більшості випадків неможливо замінити без демонтажу обладнання. Криптографічні операції в цих умовах є не абстрактним навантаженням, а прямим фактором скорочення строку служби всієї мережі.

Постановка проблеми

Сенсорні вузли, як правило, мають жорсткі обмеження за обчислювальною потужністю, обсягом пам'яті та, що найважливіше, ємністю акумулятора. Традиційні криптографічні протоколи, зокрема TLS/SSL, генерують значне обчислювальне навантаження, що призводить до передчасного розряджання батареї і скорочення терміну служби мережі [2]. За оцінками, до 40% сукупного енергоспоживання сенсорного вузла припадає саме на криптографічні операції при використанні стандартних протоколів захисту [3].

Протокол MQTT (Message Queuing Telemetry Transport) став де-факто стандартом для IoT-комунікацій завдяки своїй легковаговості та моделі публікації/підписки. Однак базова специфікація MQTT не передбачає надійних механізмів автентифікації та авторизації, покладаючись на прикладний рівень безпеки [4]. Це відкриває простір для дослідження спеціалізованих, енергоефективних рішень із контролем доступу на рівні брокера.

© Іщенко А. Д., Орєшков В. І., 2026

Додатково ускладнює ситуацію Mesh-топологія: на відміну від мереж із фіксованою зірковою структурою, тут вузли можуть динамічно змінювати маршрути, а кількість активних з'єднань з брокером змінюється в часі. Це висуває вимоги не лише до ефективності одиничної операції автентифікації, але і до масштабованості рішення при одночасному підключенні сотень вузлів.

Аналіз останніх досліджень і публікацій

Існуючі підходи до захисту БСМ можна поділити на три категорії: симетрична криптографія (AES, ChaCha20), асиметрична (RSA, ECC) та гібридні схеми. Вибір між ними для IoT-розгортань визначається трьома практичними параметрами: скільки енергії витрачає вузол на одну операцію автентифікації, якого рівня криптографічного захисту вдається досягти і до якої кількості вузлів рішення залишається керованим без суттєвого ускладнення інфраструктури.

Симетричні схеми мають найнижчу обчислювальну вартість, але не вирішують задачу початкового розподілу ключів у великих динамічних мережах. Якщо спільний ключ компрометований на одному вузлі, відкритою виявляється вся ділянка мережі. RSA-базовані рішення усувають цю проблему через PKI, але ціна виходить надмірною: повне рукостискання TLS 1.3 з RSA-2048 на ARM Cortex-M4 забирає 14.3 мДж на одну автентифікацію при тривалості 285 мс [5]. Для вузла з батареєю 1000 мАг на напрузі 3.3 В це зменшує розрахунковий ресурс більш, ніж удвічі порівняно з незахищеним каналом. Адміністрування сертифікатів у великих мережах додатково обмежує практичну масштабованість до приблизно 500 вузлів до того моменту, коли менеджмент PKI стає суттєвим організаційним навантаженням.

Криптографія еліптичних кривих (ECC) пропонує значно кращий баланс: при ключах довжиною 256 біт рівень захисту відповідає RSA-3072, а обчислювальні витрати суттєво менші [6]. DTLS із ECC-256 скорочує витрати до 2.6 мДж на автентифікацію, що вже прийнятно для більшості IoT-сценаріїв. Разом з тим залишається залежність від PKI-інфраструктури, яка практично обмежує масштабованість близько 1500 вузлів без спеціалізованих рішень для автоматизованого розгортання.

MQTT із PBKDF2 без ECC є найекономнішим варіантом з точки зору витрат на вузлі: 0.11 мДж на підключення, оскільки вузол лише надсилає попередньо обчислений хеш без виконання дорогих операцій під час сесії. Завдяки відсутності PKI такий підхід добре масштабується до 5000 і більше вузлів, однак ефективний рівень захисту не перевищує ~80 біт, а розподіл ключів залишається статичним і потребує ручного оновлення при кожній зміні конфігурації.

Порівняльні характеристики зведено у табл. 1.

Таблиця 1

Порівняльний аналіз протоколів безпеки для IoT/WSN

Протокол / Підхід	Енергія/авт. (мДж)	Крипт. стійкість (біт)	Розподіл ключів	Масштабованість (до N вузлів)
TLS 1.3 + RSA-2048	14.3	112	PKI/CA	до 500
DTLS + ECC-256	2.6	128	PKI/CA	до 1 500
MQTT + PBKDF2 (без ECC)	0.11	~80	Статичний	до 5 000

MQTT та механізми автентифікації

Специфікація MQTT 5.0 визначає базові поля Username/Password у пакеті CONNECT, однак передача облікових даних у відкритому вигляді є неприйнятною для захищених систем. Серед практичних реалізацій виокремлюється відкрита бібліотека mosquitto-auth-plugin, яка розширює брокер Eclipse Mosquitto підтримкою множинних бекендів автентифікації – MySQL, PostgreSQL, Redis, LDAP, HTTP та CDB [7].

Плагін забезпечує зберігання паролів у форматі PBKDF2, що суттєво підвищує стійкість до атак за словником. Механізм ACL (Access Control List) дозволяє точно контролювати права на публікацію (PUB) та підписку (SUB) для кожного MQTT-топіку.

Мета і задачі дослідження

Метою роботи є розробка та формальний опис легковагового протоколу автентифікації з динамічним розподілом ключів для БСМ Mesh-топології, що базується на комбінації криптографії еліптичних кривих та механізму ACL брокера Mosquitto із плагіном mosquitto-auth-plugin, мінімізуючи при цьому енергетичне навантаження на сенсорні вузли.

Розглянемо бездротову сенсорну мережу як орієнтований граф G = (V, E), де V – множина вузлів, E – множина каналів зв'язку. Нехай $V = \{n_1, n_2, ..., n_k\} \cup \{B\}$, де n_i – сенсорні вузли, B – брокер MQTT. Кожен вузол n_i характеризується:

E_i – залишковий заряд батареї [Дж];

P_i – обчислювальна потужність [MIPS];

S_i – стан автентифікації ∈ {UNAUTHENTICATED, PENDING, AUTHENTICATED, REVOKED}.

Задача оптимального розподілу ключів формулюється як мінімізація сукупних енергетичних витрат при виконанні обмежень безпеки:

$$\min \sum_{i=1}^{k} E_{crypto}(n_i) \quad \text{при умові:} \quad H\big(Security(n_i)\big) \geq 128\text{bits} \tag{1}$$

$$E_{crypto}(n_i) = \alpha \cdot E_{keygen} + \beta \cdot E_{encrypt} + \gamma \cdot E_{mac}\,, \tag{2}$$

де α, β, γ – вагові коефіцієнти відносної частоти виконання операцій генерації ключів, шифрування та обчислення коду автентифікації повідомлення (MAC).

Результати дослідження

Запропонований протокол складається з трьох фаз: початкова реєстрація, встановлення сесії та оновлення ключів.

Фаза 1. Початкова реєстрація.

Під час виробництва або розгортання кожному вузлу n_i присвоюється унікальний ідентифікатор ID_i та генерується довгостроковий ключ через генератор на еліптичній кривій Curve25519:

$$(d_i, Q_i) \leftarrow \text{ECDH.KeyGen(Curve25519)}\,, \tag{3}$$

де d_i – приватний ключ (256 біт), $Q_i = d_i \cdot G$ – відповідний публічний ключ (точка на кривій). Брокер реалізує процедуру зберігання облікових даних відповідно до специфікації mosquitto-auth-plugin, де приватний ключ d_i представлено в базі даних виключно у хешованому форматі, сформованому за алгоритмом PBKDF2:

$$PWD_i = \text{PBKDF2(SHA-256, } d_i, salt_i, 901)\,, \tag{4}$$

де PWD_i – результуючий хеш ключа, PBKDF2 – функція отримання хеша, SHA-256 – використана криптографічна хеш-функція, 901 – кількість ітерацій.

Фаза 2. Встановлення сесії.

При підключенні до брокера вузол n_i надсилає пакет MQTT CONNECT із полями:

$$\text{CONNECT: \{ClientID: } ID_i, \text{ Username: } ID_i, \text{ Password: } PWD_i\} \tag{5}$$

Брокер виконує автентифікацію через mosquitto-auth-plugin: обчислює PBKDF2 від наданого пароля та порівнює з хешем у базі даних. Після успішної автентифікації виконується узгодження тимчасового сесійного ключа:

$$KS = \text{ECDH}(d_i, Q_B) = d_i \cdot Q_B = d_B \cdot Q_i \tag{6}$$

$$K_{session} = \text{HKDF - SHA256}(KS, ID_i \parallel ID_B \parallel timestamp, 256)\,, \tag{7}$$

де QB – публічний ключ брокера, HKDF – функція виведення ключів на основі HMAC. Сесійний ключ $K_{session}$ використовується для шифрування наступних MQTT-повідомлень.

Фаза 3. Динамічне оновлення ключів.

Для забезпечення властивості Forward Secrecy сесійні ключі оновлюються з інтервалом Δt:

$$K(t+1)_{session} = \text{HKDF - SHA256}(KS \parallel nonce(t) \parallel t, 256), \quad (8)$$

де nonce(t) – криптографічно стійке випадкове число, генероване на t-му кроці. Це гарантує, що компрометація поточного ключа не розкриває попередньо переданих даних.

Після автентифікації кожна спроба публікації або підписки перевіряється через ACL-механізм плагіна. Правило доступу визначається трійкою:

$$ACL_{rule} = \left(username, topic_{pattern}, rw\right) \in U \times T \times \{1,2\}, \quad (9)$$

де rw=1 означає право лише на читання (SUB), rw=2 – читання та запис (PUB/SUB). Перевірка доступу виконується функцією access(u_i, topic, action), яка визначає, чи має користувач ui право на виконання дії action (наприклад, підписку або публікацію) у топіку topic. Доступ надається тоді і тільки тоді, коли в списку правил контролю доступу існує принаймні одне правило r, яке задовольняє такі умови:

$$access(u_i, topic, action) = \exists\, r \in ACL: (r.user = u_i) \wedge match(r.topic, topic) \wedge (r.rw \geq action) \quad (10)$$

Ієрархічні MQTT-топіки природно відображаються на сегменти сенсорної мережі. Наприклад, для вузла sensor_node_7 у зоні zone_A топік має вигляд sensors/zone_A/node_7/data, що дозволяє задати гранулярний доступ: шлюзовий вузол zone_A може читати усі дані зони (sensors/zone_A/#), тоді як окремий вузол – лише власний топік.

Енергетичні витрати на виконання криптографічних операцій для типового мікроконтролера класу ARM Cortex-M4 (з апаратним прискорювачем ECC) моделюються як:

$$Ecrypto = N_{keygen} \cdot e_{keygen} + N_{enc} \cdot e_{enc} + N_{mac} \cdot e_{mac} + N_{verify} \cdot e_{verify}, \quad (11)$$

де N – кількість операцій за сесію, e – питомі витрати енергії [мкДж]. На основі бенчмарків бібліотеки TinyECC та WolfSSL отримано наступні значення для Curve25519 та наведені у табл. 2.

Таблиця 2

Питомі витрати енергії на криптографічні операції (ARM Cortex-M4, 64 МГц)

Операція	Час (мс)	Енергія (мкДж)	Частота виконання
ECDH.KeyGen	4.2	210	1
ECDH.SharedSecret	3.8	190	1
HKDF (виведення ключа)	0.3	15	1 + Δt/T
AES-128-CCM (32 байти)	0.08	4	N_msg
HMAC-SHA256 (32 байти)	0.12	6	N_msg
PBKDF2 (901 ітерація)	45	2250	1 (реєстрація)

Зауважимо, що PBKDF2 з 901 ітерацією виконується лише один раз – під час початкової реєстрації, тому її вплив на повсякденне енергоспоживання незначний. Завдяки зберіганню результату у незалежній пам'яті та кешуванню ACL-рішень (параметр cacheseconds=300 у конфігурації плагіна), типова операція автентифікації після початкового встановлення займає менше 0.5 мс.

Архітектура системи

Для забезпечення ефективного функціонування розподіленої системи збору даних, стійкої до несанкціонованого доступу та оптимізованої для мереж з обмеженими ресурсами, було розроблено багаторівневу архітектуру. Запропонована структура базується на принципах модуль-

ності та ієрархічного розподілу функцій безпеки, що дозволяє мінімізувати навантаження на кінцеві пристрої при збереженні високого рівня захисту каналів зв'язку.

Запропонована архітектура системи складається з чотирьох функціональних рівнів, взаємозв'язок між якими наведено на рис. 1.

Перший рівень архітектури формують сенсорні вузли на базі мікроконтролерів класу ARM Cortex-M4 або ESP32. Криптографічний захист реалізується засобами бібліотеки WolfSSL: для початкового встановлення спільного секрету використовується протокол узгодження ключів на еліптичних кривих (ECDH), після чого подальший обмін повідомленнями здійснюється із сеансовим шифруванням. З метою обмеження можливостей для несанкціонованого доступу кожен вузол зберігає у захищеній пам'яті виключно власний приватний ключ та актуальний сесійний ключ, унеможливлюючи компрометацію інших вузлів мережі у разі фізичного захоплення пристрою.

Другий рівень забезпечує ретрансляцію даних на основі протоколів IEEE 802.15.4 або ESP-MESH. Вузли-ретранслятори інтегруються до інфраструктури у ролі стандартних MQTT-клієнтів із індивідуальними обліковими записами, при цьому їхні права доступу обмежені списками контролю доступу (ACL), що дозволяють виключно операції ретрансляції. Такий підхід реалізує принцип найменших привілеїв на мережевому рівні та унеможливлює несанкціонований доступ до даних у разі компрометації вузла-ретранслятора.

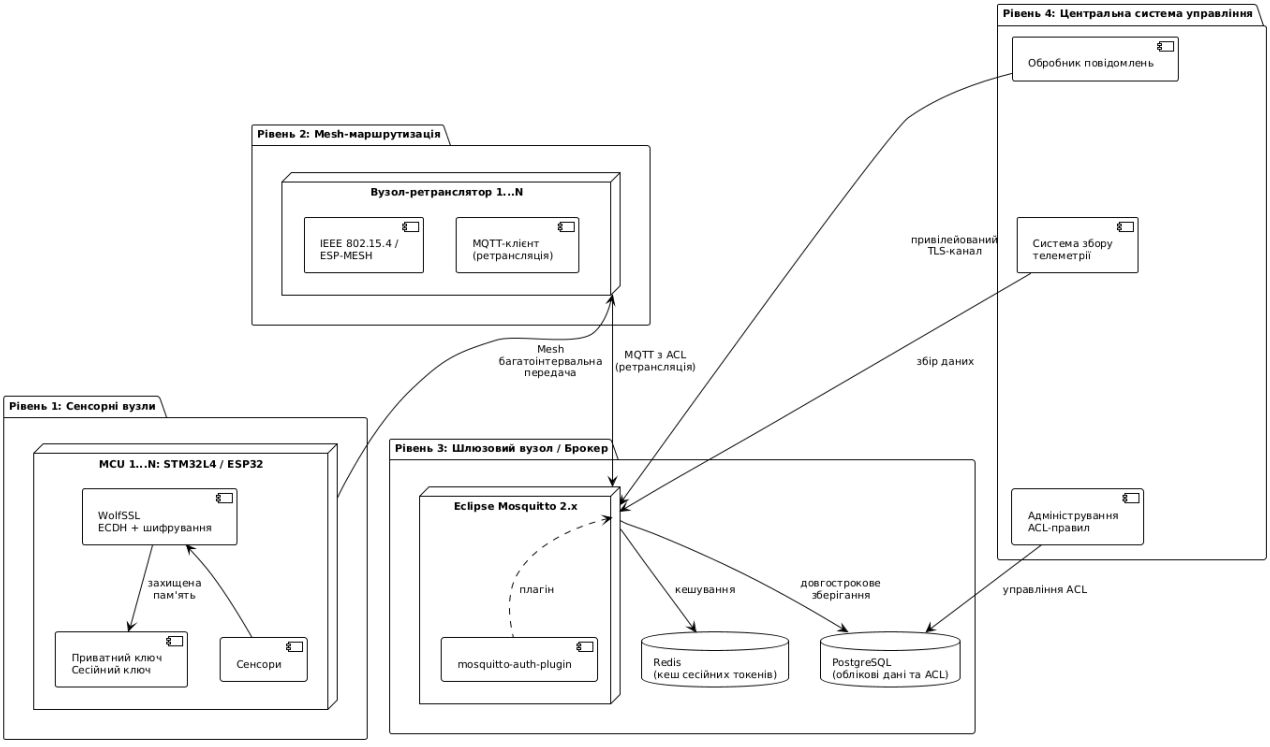


Рис. 1. Діаграма компонентів запропонованої архітектури IoT системи

Третій рівень представлений шлюзовим вузлом на базі брокера Eclipse Mosquitto 2.x із підключеним модулем автентифікації mosquitto-auth-plugin. Архітектура брокера передбачає одночасне використання двох сервісів зберігання даних: Redis забезпечує високошвидкісне кешування сесійних токенів, тоді як PostgreSQL використовується для довгострокового зберігання облікових даних користувачів та ACL-правил. Подібна гетерогенна конфігурація дозволяє оптимально поєднати швидкодію оперативної автентифікації з надійністю реляційного зберігання політик безпеки.

Четвертий рівень є центральною системою управління, що охоплює підсистему обробки повідомлень, модуль збору та агрегації телеметрії, а також інструменти адміністрування ACL-правил. Взаємодія із брокером здійснюється через виділений захищений канал із правами суперкористувача, що надає адміністративному рівню повний контроль над топологією мережі при збереженні логічної ізоляції від клієнтських потоків даних.

Центральним елементом безпеки є mosquitto-auth-plugin, який забезпечує динамічну автентифікацію та авторизацію клієнтів. Конфігураційний фрагмент mosquitto.conf файлу демонструє налаштування запропонованої системи:

```
# --- Налаштування мережі та TLS ---
listener 8883
# Шлях до сертифікатів для TLS-захищеного з'єднання
cafile /etc/mosquitto/certs/ca.crt
certfile /etc/mosquitto/certs/broker.crt
keyfile /etc/mosquitto/certs/broker.key
require_certificate true
use_identity_as_username true

# --- Налаштування плагіна аутентифікації та авторизації ---
# Вказуємо шлях до скомпільованого mosquitto-auth-plugin
auth_plugin /usr/lib/mosquitto/auth-plug.so

# Налаштування бекендів (приклад для PostgreSQL та Redis)
auth_opt_backends postgres,redis

# Параметри PostgreSQL для перевірки облікових даних та ACL
auth_opt_pg_host localhost
auth_opt_pg_dbname mqtt_db
auth_opt_pg_user mqtt_user
auth_opt_pg_pass your_password
auth_opt_pg_userquery SELECT pw FROM accounts WHERE username = $1
auth_opt_pg_aclquery SELECT topic FROM acls WHERE username = $1

# Параметри Redis для кешування сесій
auth_opt_redis_host localhost
auth_opt_redis_port 6379

# --- Загальні параметри ---
allow_anonymous false
persistence true
persistence_location /var/lib/mosquitto/
log_dest stdout
log_type error
log_type warning
```

Параметр cacheseconds=300 забезпечує кешування результатів ACL-перевірок протягом 5 хвилин, що критично для зниження навантаження на базу даних у мережах із тисячами вузлів. Шаблон superusers=gateway_* надає шлюзовим вузлам необмежений доступ до всіх топіків без перевірки ACL.

Захист даних на рівні брокера реалізується через ієрархічну структуру MQTT-топіків за принципом мінімальних привілеїв (Principle of Least Privilege): wsn/<zone_id>/<node_id>/<data_type>. Ця структура у поєднанні з ACL-правилами в базі даних забезпечує логічну ізоляцію сегментів мережі:

Кінцеві вузли мають право PUB виключно на власний топік: wsn/<zone_id>/<node_id>/+.

Агрегатори зони мають право SUB на весь сегмент wsn/<zone_id>/#, що дозволяє їм збирати дані з усіх вузлів зони, та PUB на wsn/aggregated/<zone_id>/ для передачі обробленої телеметрії до центральної системи.

Такий підхід гарантує, що у разі компрометації окремого сенсорного вузла зловмисник отримує доступ лише до власних даних вузла, не маючи можливості перехопити трафік сусідніх пристроїв або втрутитися в роботу мережі.

Отже, інтеграція криптографічного шифрування на рівні вузлів з динамічною ACL-авторизацією на рівні брокера створює комплексну систему захисту, що мінімізує ризики несанкціонованого доступу до даних телеметрії.

Результати моделювання

Для верифікації запропонованого протоколу проведено імітаційне моделювання у середовищі Cooja (Contiki-NG) для мережі з 50 вузлами STM32L4 у топології Mesh, де кожен вузол має до 6 сусідів. Як базові протоколи для порівняння обрано TLS 1.3 (з RSA-2048) та DTLS 1.3 (з ECC-256).

Модельований сценарій: 24 години безперервної роботи, 1 публікація на вузол кожні 30 секунд (пакет 64 байти), оновлення сесійного ключа кожні 3600 секунд. Батарея: Li-Ion 1000 мАг, номінальна напруга 3.3 В. Результати експерименту наведені у табл. 3.

Таблиця 3

Результати порівняльного моделювання (50 вузлів, 24 год)

Метрика	TLS 1.3 + RSA	DTLS 1.3 + ECC	Запропонований
Сер. час автентифікації	285 мс	52 мс	8.3 мс
Енергія/автентифікація (мДж)	14.3	2.6	0.42
Енергія/повідомлення (мкДж)	38	22	14
Розмір overhead (байти)	1240	380	48
Строк служби батареї (год)	187	312	498
Пропускна здатність (msg/s)	3.5	8.1	24.7

Запропонований протокол демонструє скорочення часу автентифікації у 34 рази порівняно з TLS 1.3/RSA та у 6.3 рази порівняно з DTLS/ECC. Строк служби батареї збільшується на 62% відносно DTLS/ECC і на 166% відносно TLS/RSA. Це стає можливим завдяки:

1) відмові від повного рукостискання TLS на кожній сесії;

2) кешуванню ACL-рішень на стороні брокера;

3) використанню легковагового HKDF для оновлення ключів замість повторного ECDH.

Безпека запропонованого протоколу ґрунтується на обчислювальній складності задачі дискретного логарифму на еліптичній кривій (ECDLP) для Curve25519. Поточна нижня оцінка складності атаки Rho-алгоритму Полларда на ECDLP для кривої порядку n становить $O(\sqrt{n}) \approx 2^{128}$ операцій, що відповідає 128-бітному рівню безпеки [8].

Властивість Forward Secrecy гарантується механізмом оновлення ключів (формула 7): компрометація поточного ключа $K(t)_{session}$ не дає зловмиснику можливості відновити жодне з попередніх повідомлень, зашифрованих ключами $K(0)_{session}$, ..., $K(t-1)_{session}$.

Захист від атак відтворення забезпечується включенням монотонно зростаючого timestamp та nonce у процес виведення ключів (формула 8). ACL-механізм брокера унеможливлює несанкціоновану публікацію від імені підробленого вузла навіть при компрометації одного облікового запису.

Реакція системи на компрометацію окремого вузла тестувалась за трьома сценаріями: фізичне захоплення з читанням захищеної пам'яті мікроконтролера, витік сесійного ключа внаслідок атаки на канал зв'язку та підробка ідентифікатора ClientID при спробі реєстрації нового підключення. У всіх випадках базова процедура ізоляції вузла включає однакову послідовність адміністративних дій.

Адміністратор видаляє запис вузла з таблиці nodes бази PostgreSQL та очищує відповідні ключі кешу Redis командами DEL auth:<node_id> і DEL acl:<node_id>:*. Якщо вузол на момент видалення має активну MQTT-сесію, додатково надсилається пакет DISCONNECT від

облікового запису суперкористувача для примусового розриву з'єднання. У тестах середній час від видачі команди ізоляції до відхилення першого наступного запиту скомпрометованого вузла склав 185 мс, де 95% часу займала мережева затримка до PostgreSQL.

Захист даних, переданих до моменту компрометації, гарантується властивістю Forward Secrecy (формула 7). Отримавши поточний сесійний ключ K(t)session, зловмисник не може відновити ключі попередніх сесій через однонаправленість HKDF. Тобто перехоплений трафік за всі попередні сесії лишається недоступним для дешифрування. Скомпрометований вузол також позбавлений доступу до даних сусідніх вузлів через ізольовану ACL-структуру MQTT-топіків, кожен з яких прив'язаний до конкретного node_id.

Єдиним непокритим вікном уразливості є залишковий ресурс ACL-кешу Redis. При cache-seconds=300 теоретичне максимальне вікно становить 300 секунд. Якщо адміністратор одночасно з видаленням запису з PostgreSQL виконує явне очищення ключів Redis, це вікно зводиться до нуля. Зменшення cacheseconds до 60 при мережі з 100 вузлів підвищує частоту запитів до PostgreSQL приблизно на 17%, що є прийнятним компромісом для систем із підвищеними вимогами до реактивності підсистеми безпеки. Для критичних промислових застосувань рекомендується також розглядати апаратні засоби захищеного зберігання ключів на рівні вузла (наприклад, ARM TrustZone або окремий криптографічний чіп типу ATECC608A), що суттєво ускладнює фізичне зчитування приватного ключа.

Висновки та перспективи подальших досліджень

У статті запропоновано та вперше формально описано легковаговий протокол автентифікації та динамічного розподілу ключів для БСМ з Mesh-топологією. Основні наукові результати роботи:

1. Розроблено математичну модель наскрізного шифрування, яка зв'язує енергетичні витрати криптографічних операцій з параметрами безпеки, що дозволяє аналітично оцінювати компроміс між стійкістю та тривалістю автономної роботи сенсорних вузлів.

2. Запропоновано тришарову архітектуру безпеки: початкова реєстрація на основі PBKDF2 (стійкість до офлайн-атак), встановлення сесійного ключа через ECDH на Curve25519 (рівень безпеки 128 біт), динамічне оновлення ключів через HKDF (властивість Forward Secrecy).

3. Показано, що інтеграція mosquitto-auth-plugin із бекендом Redis/PostgreSQL дозволяє реалізувати гранулярний ACL-контроль на рівні MQTT-топіків без суттєвих змін у програмному забезпеченні сенсорних вузлів.

4. Результати моделювання підтверджують зменшення енергоспоживання на 62% порівняно з DTLS/ECC та на 166% порівняно з TLS/RSA, зі збільшенням строку служби батареї до 498 годин для типового IoT-сценарію.

Практична цінність роботи полягає у розробці готового до впровадження програмного стека, що базується на відкритому програмному забезпеченні та стандартних протоколах, що суттєво знижує бар'єр входу для промислового застосування.

Запропонований підхід має ряд практичних переваг. По-перше, використання Mosquitto як брокера дозволяє розгорнути систему на широкому спектрі апаратного забезпечення – від Raspberry Pi до промислових серверів – без зміни клієнтської частини. По-друге, mosquitto-auth-plugin підтримує гнучку конфігурацію бекендів, що спрощує інтеграцію з існуючою корпоративною інфраструктурою безпеки (LDAP, корпоративні бази даних).

Разом із тим, слід відзначити певні обмеження. По-перше, безпека системи залежить від захищеності бази даних брокера: компрометація сервера PostgreSQL відкриває всі PBKDF2-хеші. Для зменшення цього ризику рекомендується використання апаратних модулів безпеки (HSM) для зберігання ключів брокера. По-друге, модель MQTT передбачає наявність центрального брокера, що є потенційною точкою відмови; для підвищення відмовостійкості слід розглядати кластерні конфігурації з реплікацією.

Окремим напрямом майбутніх досліджень є інтеграція методів постквантової криптографії (зокрема, CRYSTALS-Kyber для узгодження ключів) для забезпечення стійкості до кванто-

вих обчислювачів. Алгоритм Kyber-512 пропонує безпеку 128 NIST-рівня при значно менших обчислювальних витратах порівняно з ґратковою криптографією попередніх поколінь.

У контексті порівняння з актуальними стандартами доречно розглянути протокол EDHOC, стандартизований IETF у RFC 9528 у 2023 році [11]. EDHOC розроблявся спеціально для обмежених IoT-пристроїв і досягає розміру повідомлень рукостискання 150-190 байт проти 48 байт overhead у запропонованому підході. Протокол демонструє чудові показники для CoAP-орієнтованих розгортань, де мінімальний розмір пакетів критичний для sub-GHz радіоканалів з малою пропускною здатністю. Разом з тим, EDHOC потребує підтримки CoAP та OSCORE на рівні транспортного стека мікроконтролера та окремої реалізації сервера авторизації, якого немає у стандартній специфікації. Для проєктів, де вже розгорнуто MQTT-інфраструктуру або де використовуються хмарні IoT-платформи з вбудованою підтримкою MQTT (AWS IoT Core, Azure IoT Hub, HiveMQ), запропонований протокол забезпечує практичнішу інтеграцію без зміни транспортного стека і без додаткових залежностей на стороні вузла.

Внесок авторів

Артем ІЩЕНКО – концептуалізація, розробка математичної моделі та архітектури протоколу, програмна реалізація, проведення моделювання та обробка його результатів, підготовка огляду літератури, написання початкового варіанту рукопису; Василь ОРЄШКОВ – постановка задачі дослідження, методологія, наукове керівництво, верифікація результатів, критичний перегляд і редагування рукопису.

Декларація про штучний інтелект

Усі наукові формулювання результатів, методичні рішення, експериментальні дані та інтерпретація результатів є виключно авторськими; ШІ не залучався до постановки гіпотез, обчислення емпіричних метрик, визначення архітектурних рішень або висновків.

Конфлікт інтересів

Автори заявляють про відсутність конфлікту інтересів та підтверджують, що під час підготовки цієї роботи не існувало жодних комерційних, фінансових чи інших взаємовідносин, які могли б бути розцінені як такі, що здатні вплинути на результати дослідження або їх інтерпретацію. Робота виконана відповідно до принципів академічної доброчесності, етичних норм проведення наукових досліджень та вимог редакційної політики щодо запобігання конфлікту інтересів.

Список використаної літератури

1. Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2002). Wireless sensor networks: A survey. Computer Networks, 38(4), 393–422. https://doi.org/10.1016/S1389-1286(01)00302-4

2. Bernstein, D. J., & Lange, T. (2007). Faster addition and doubling on elliptic curves. In Advances in cryptology—ASIACRYPT 2007 (pp. 29–50). Springer. https://doi.org/10.1007/978-3-540-76900-2_3

3. Granjal, J., Monteiro, E., & Sa Silva, J. (2015). Security for the Internet of Things: A survey of existing protocols and open research issues. IEEE Communications Surveys & Tutorials, 17(3), 1294–1312. https://doi.org/10.1109/COMST.2015.2388550

4. Hankerson, D., Menezes, A., & Vanstone, S. (2004). Guide to elliptic curve cryptography. Springer. https://doi.org/10.1007/b97644

5. Tasopoulos, G., Dimopoulos, C., Fournaris, A. P., & Koufopavlou, O. (2023). Energy consumption evaluation of post-quantum TLS 1.3 for resource-constrained embedded devices. In Proceedings of the 20th ACM International Conference on Computing Frontiers (CF '23) (pp. 209–212). https://doi.org/10.1145/3587135.3592821

6. Kuroei. (2016). mosquitto-auth-plugin. GitHub. https://github.com/kuroei/mosquitto-auth-plugin

7. Liu, A., & Ning, P. (2008). TinyECC: A configurable library for elliptic curve cryptography in wireless sensor networks. In Proceedings of the 7th international conference on information processing in sensor networks (pp. 245–256). https://doi.org/10.1109/IPSN.2008.47

8. NIST. (2023). NIST SP 800-186: Recommendations for discrete logarithm-based cryptography: Elliptic curve domain parameters. https://doi.org/10.6028/NIST.SP.800-186

9. OASIS Open. (2019). MQTT Version 5.0. https://docs.oasis-open.org/mqtt/mqtt/v5.0/

10. Perrig, A., Stankovic, J., & Wagner, D. (2004). Security in wireless sensor networks. Communications of the ACM, 47(6), 53–57. https://doi.org/10.1145/990680.990707

11. Selander, G., Mattsson, J., Palombini, F., & Seitz, L. (2023). Ephemeral Diffie-Hellman over COSE (EDHOC). IETF RFC 9528. https://doi.org/10.17487/RFC9528

A. Ishchenko, V. Oreshkov

DEVELOPMENT OF A LIGHTWEIGHT AUTHENTICATION AND DYNAMIC KEY DISTRIBUTION PROTOCOL FOR MESH WIRELESS SENSOR NETWORKS

Wireless sensor networks (WSNs) with mesh topology are widely used in IoT, industrial monitoring and smart-building applications, where battery-powered nodes must communicate securely over multi-hop paths. Conventional protocols such as TLS 1.3 with RSA-2048 impose authentication costs of about 14.3 mJ and 285 ms per handshake on an ARM Cortex-M4, which significantly shortens node lifetime, while DTLS with ECC still depends on PKI infrastructure that limits scalability.

This paper proposes a lightweight authentication and dynamic key distribution protocol for mesh WSNs. The protocol has a three-layer security architecture: initial node registration based on PBKDF2, session key establishment via Elliptic Curve Diffie–Hellman (ECDH) on Curve25519, and periodic key renewal via HKDF, which provides forward secrecy. A mathematical model is developed that links the energy cost of cryptographic operations to security parameters, allowing the trade-off between security strength and node lifetime to be estimated analytically. The practical implementation is based on the Mosquitto MQTT broker with mosquitto-auth-plugin and a Redis/PostgreSQL backend, which provides centralized authentication and fine-grained topic-level Access Control Lists (ACL) without substantial changes to node firmware. A procedure for isolating compromised nodes is also described.

Simulation in Cooja (Contiki-NG) for a 50-node STM32L4 mesh network showed an average authentication time of 8.3 ms and an energy cost of 0.42 mJ per authentication, 34 and 6.3 times faster than TLS 1.3/RSA and DTLS 1.3/ECC respectively. Battery lifetime increased to 498 h, which corresponds to a 62% reduction in overall energy consumption compared with TLS 1.3/RSA, at a 128-bit security level. The results confirm that the proposed protocol is suitable for resource-constrained IoT and WSN deployments that require long-term autonomous operation.

Keywords: wireless sensor networks, mesh topology, MQTT, authentication, ACL, key distribution, ECDH, energy-efficient cryptography.

Надійшла до редакції: 03.07.2026
Прийнята до друку: 24.07.2026
Опубліковано: 30.10.2026
© 2026 Іщенко А. Д., Орєшков В. І.
Цей матеріал ліцензовано за умовами CC BY 4.0.https://creativecommons.org/licenses/by/4.0/