https://doi.org/10.31891/2219-9365-2026-87-44
УДК 004.65:004.056.5:004.738.5

СТЕЦЮК Микола
Хмельницький національний університет
https://orcid.org/0000-0003-0312-2276
e-mail: mykola.stetsiuk@khmnu.edu.ua
СТЕЦЮК Юрій
Хмельницький національний університет
https://orcid.org/0000-0003-0312-2276
e-mail: yuriy.stetsuk@khmnu.edu.ua
СТЕЦЮК Василь
Хмельницький національний університет
https://orcid.org/0000-0001-9880-2666
e-mail: swmua@khmnu.edu.ua

РОБАСТНИЙ АДАПТИВНИЙ МЕТОД ВИЯВЛЕННЯ АНОМАЛЬНИХ ТРАНЗАКЦІЙ У БАЗАХ ДАНИХ ІОТ-СИСТЕМ

У роботі розглянуто виявлення аномалій безпосередньо в журналі аудиту системи керування базами даних, у якій накопичуються телеметрія, команди та службові події IoT-пристроїв. Запропоновано робастний адаптивний метод, що формує профіль нормальної поведінки окремого пристрою, шлюзу, облікового запису або прикладного сервісу за агрегованими характеристиками SQL-запитів і транзакцій. Інтегральну оцінку аномальності обчислено за регуляризованою робастною відстанню Махаланобіса з оцінюванням центра та коваріації методом Minimum Covariance Determinant. Порогове значення визначено на основі моделі Peaks Over Threshold із узагальненим розподілом Парето, а накопичення слабких тривалих відхилень реалізовано процедурою CUSUM. Експериментальний стенд охоплює детермінований генератор журналу, реляційне сховище агрегованих вікон, модуль виявлення та блок обчислення метрик. На синтетичному журналі з 60 суб'єктами та 28 800 п'ятихвилинними вікнами запропонований метод отримав Precision 86,48 %, Recall 72,32 % та F1 78,77 % за частоти хибних спрацьовувань 0,55 %. Для 16 сценаріїв повільного витоку даних подієва повнота становила 81,25 %, а медіанна затримка виявлення – шість часових вікон. Медіанний час побудови 60 профілів дорівнював 1,10 с; після формування ознак пакетне оцінювання 14 400 вікон тривало 19,90 мс, а 95-й процентиль потокової затримки одного вікна становив 23,3 мкс. Межі вимірювання явно відокремлюють час детектора від накладних витрат штатного аудиту конкретної промислової СУБД.

Ключові слова: база даних, СУБД, IoT, журнал аудиту, аномальна транзакція, робастна відстань Махаланобіса, Minimum Covariance Determinant, Peaks Over Threshold, CUSUM.

STETSIUK Mykola, STETSIUK Yuriy, STETSIUK Vasyl
Khmelnytskyi National University

ROBUST ADAPTIVE METHOD FOR DETECTING ANOMALOUS TRANSACTIONS IN DATABASES OF IOT SYSTEMS

This paper addresses anomaly detection directly in the audit log of a database management system that stores telemetry, commands, and service events generated by Internet of Things devices. The proposed robust adaptive method builds a normal-behavior profile for an individual device, gateway, database account, or application service from aggregated characteristics of SQL queries and transactions. An integral anomaly score is calculated using a regularized robust Mahalanobis distance whose location and covariance parameters are estimated by the Minimum Covariance Determinant method. The decision threshold is calibrated from the tail of the score distribution with a Peaks Over Threshold model and a generalized Pareto distribution, while weak persistent deviations are accumulated by a CUSUM procedure. The experimental stand includes a deterministic audit-log generator, a relational repository of aggregated windows, a detection module, and a metric-calculation module. A reproducible synthetic dataset contains 60 subjects and 28,800 five-minute windows, including five types of anomalous database activity. The complete method achieved a precision of 86.48%, a recall of 72.32%, an F1-score of 78.77%, and a false-alarm rate of 0.55%. For 16 slow data-exfiltration episodes, event-level recall was 81.25%, with a median detection delay of six windows. The median time required to build 60 profiles was 1.10 s; after feature aggregation, batch scoring of 14,400 windows took 19.90 ms, and the 95th percentile of streaming latency for one window was 23.3 µs. The reported timing covers the analytical detector and does not represent the overhead of native auditing in a production DBMS.

Keywords: database; DBMS; Internet of Things; audit log; anomalous transaction; robust Mahalanobis distance; Minimum Covariance Determinant; Peaks Over Threshold; CUSUM.

Стаття надійшла до редакції / Received 10.07.2026
Прийнята до друку / Accepted 26.08.2026
Опубліковано / Published 10.09.2026

ПОСТАНОВКА ПРОБЛЕМИ У ЗАГАЛЬНОМУ ВИГЛЯДІ ТА ЇЇ ЗВ'ЯЗОК ІЗ ВАЖЛИВИМИ НАУКОВИМИ ЧИ ПРАКТИЧНИМИ ЗАВДАННЯМИ

Інформаційні системи Інтернету речей створюють безперервний потік телеметрії, команд, діагностичних повідомлень і службових подій. Після проходження через шлюзи ці дані потрапляють до

реляційної, часової або гібридної бази даних. Саме СУБД у такій архітектурі стає точкою, де сходяться дії пристроїв, прикладних сервісів, операторів і фонових процесів. Розподіленість, неоднорідність і ресурсні обмеження IoT-пристроїв ускладнюють наскрізний контроль безпеки [22–24]. Тому атака або технічна несправність може проявлятися не лише як нетиповий мережевий трафік, а й як аномальна послідовність запитів, надмірне читання, незвична зміна рядків, серія відкатів, доступ до нетипової множини таблиць або використання легітимного облікового запису не за його звичайним профілем.

Механізми розмежування доступу СУБД перевіряють, чи має суб'єкт формальне право виконати операцію. Водночас вони не завжди здатні відрізнити штатне використання дозволу від зловживання ним. Прикладний сервіс може мати законний доступ до таблиці телеметрії, але виконати масове читання даних, яке не відповідає його типовій роботі. Скомпрометований IoT-шлюз може використовувати правильні облікові дані, однак раптово змінити співвідношення виконуваних операторів SELECT, INSERT, UPDATE і DELETE, перейшовши до нехарактерних для нього дій. Тому контроль функціонування IoT-систем має бути доповненим поведінковим аналізом журналу аудиту.

Методи виявлення аномалій, що базуються на незалежному пороговому контролі окремих ознак, характеризуються простотою реалізації, проте не враховують статистичні залежності між параметрами поведінки системи. Зокрема, збільшення кількості запитів може бути наслідком допустимого пікового навантаження і саме по собі не свідчити про аномальну активність. Водночас одночасне зростання кількості виконаних запитів, обсягу прочитаних даних, числа охоплених таблиць і тривалості виконання операцій може формувати нетиповий багатовимірний профіль, характерний для несанкціонованого доступу або масового збирання даних. Для врахування взаємозв'язків між такими ознаками доцільним є використання відстані Махаланобіса, яка оцінює відхилення поточного вектора спостережень з урахуванням їхньої коваріаційної структури. Однак класичне оцінювання середнього вектора та коваріаційної матриці є чутливим до викидів і аномальних спостережень у навчальній вибірці, що може призводити до спотворення еталонної моделі нормальної поведінки. Крім того, фіксоване порогове значення не враховує часової мінливості навантаження та поступових змін характеру функціонування IoT-системи. Тому для підвищення стійкості поведінкового аналізу доцільно застосовувати робастне оцінювання параметрів багатовимірного розподілу в поєднанні з адаптивним механізмом визначення порогу аномальності.

ФОРМУЛЮВАННЯ ЦІЛЕЙ СТАТТІ

Метою цієї роботи є розроблення цілісного методу виявлення аномальних транзакцій у базі даних IoT-системи, який:

- ✓ формує окремі профілі для пристроїв, ролей або прикладних сервісів;
- ✓ враховує спільну зміну кількох ознак журналу СУБД;
- ✓ є стійким до поодиноких забруднених спостережень у профілі нормальної поведінки;
- ✓ автоматично калібрує поріг за хвостом розподілу оцінки;
- ✓ виявляє як різкі події, так і слабкі тривалі відхилення;
- ✓ записує результат у нормалізовану структуру БД, придатну для подальшого розслідування.

Наукова новизна полягає в поєднанні робастного багатовимірного профілю доступу до БД, потокового порогу на основі теорії екстремальних значень і накопичувального контролю CUSUM в єдиній схемі, орієнтованій на журнали транзакцій IoT-систем. На відміну від початкової постановки, об'єктом аналізу є не абстрактний мережевий трафік, а формалізовані операції над даними, їхні виконавці, цільові об'єкти БД та результат виконання.

ВИКЛАД ОСНОВНОГО МАТЕРІАЛУ

Поведінковий контроль спирається на класичну модель виявлення вторгнень, у якій відхилення поточної активності порівнюється з профілем нормальної поведінки [1]. Загальні підходи до аномалій охоплюють статистичні, сусідські, кластерні, спектральні та класифікаційні моделі [9]. Для баз даних рання система DEMIDS формувала з журналу аудиту типові робочі області користувача [2], а рольовий підхід пов'язав поведінкові профілі з RBAC [3]. Kamra, Terzi та Bertino надалі запропонували профілювати нормальний доступ за SQL-запитами як для окремих користувачів, так і для ролей [4]. Це підтверджує, що журнал СУБД є самостійним джерелом ознак для виявлення вторгнень, а не лише додатком до мережевого моніторингу.

У системі DetAnom профіль прикладної програми пов'язує сигнатуру запиту з обмеженнями контексту його виконання. Відхилення між поточним запитом і профілем програми розглядається як ознака аномальної транзакції [6]. DBSAFE орієнтовано на захист від спроб ексфільтрації та використовує профілювання доступу прикладних програм до бази даних [7]. Архітектура збору подій також має значення: винесення сенсора й захищеного сховища аудиту за межі контрольованої СУБД дає змогу спостерігати дії звичайних і привілейованих користувачів [8]. Ці роботи показують практичну важливість аналізу дій формально авторизованих суб'єктів.

Синтаксичний профіль не завжди достатній. Mathew та співавтори показали доцільність профілювання саме множини даних, до яких звертається користувач, оскільки однакова форма SQL-запиту

може відповідати різним намірам і різному обсягу доступу [5]. Для IoT-систем це особливо важливо: типовий шаблон SELECT ... WHERE device_id = ? може повертати кілька останніх вимірювань або повну історію за тривалий період. Тому запропонований вектор ознак поєднує характеристики операцій, часу виконання та обсягу даних.

Для стійкого багатовимірного профілю використано оцінювач Minimum Covariance Determinant (MCD). Алгоритм FAST-MCD дає змогу обчислювати робастні оцінки центра та коваріації на підмножині спостережень із мінімальним визначником коваріаційної матриці [10]. Подальші узагальнення MCD охоплюють регуляризацію та застосування у задачах виявлення багатовимірних викидів [11]. Робастна відстань Махаланобіса на основі MCD є менш чутливою до спотворення центра та коваріації поодинокими викидами, ніж класичний варіант [12].

Порогування інтегральної оцінки становить окрему проблему. Модель Peaks Over Threshold спирається на асимптотичний опис перевищень високого рівня узагальненим розподілом Парето [13]. Метод Streaming Peaks Over Threshold (SPOT) переносить цей принцип у потік і дає змогу задавати хвостовий ризик замість ручного вибору абсолютного порога [14]. Для тривалих малих змін використано CUSUM, початкову послідовну схему якого запропоновано Page [15].

Класичними альтернативами для навчання без учителя є Isolation Forest, який ізолює рідкісні спостереження випадковими деревами [16], та Local Outlier Factor (LOF), що порівнює локальну щільність сусідніх спостережень [17]. Однокласові межі норми можуть будуватися методами One-Class SVM [18] і Support Vector Data Description [19], а автоенкодер оцінює аномальність за похибкою нелінійного відновлення [20]. Огляд глибоких і класичних підходів показує, що вибір детектора залежить від структури даних, обсягу навчальної вибірки та вимог до пояснюваності [21]. Isolation Forest є швидким і придатним до великих вибірок, однак його випадкова модель не дає безпосередньої статистичної інтерпретації порога. LOF здатний знаходити локальні відхилення, але потребує зберігання сусідів і залежить від вибору їх кількості. Ядерні однокласові моделі й автоенкодери описують нелінійні межі, проте потребують налаштування гіперпараметрів і більшої калібрувальної вибірки. Для цієї роботи обрано MCD + POT, оскільки для кожного суб'єкта доступно лише 240 калібрувальних вікон, потрібні відтворюваний поріг із заданим хвостовим ризиком, фіксована вартість оцінювання $O(p^2)$ та пояснення сигналу через ознаки журналу. Це обґрунтовує вибір методу, але не означає його універсальної переваги над зазначеними ML-підходами.

Відмічається, що сучасний цифровий світ оперує величезними обсягами потокових даних у різних областях, що стало повсюдністю. Однак значна їх частина не мають маркування, що ускладнює ідентифікацію подій, зокрема виявлення аномалій. Це завдання стає ще складнішим у нестаціонарних середовищах, де продуктивність моделі може з часом погіршуватися через дрейф концепцій. Для вирішення цих проблем у [22] пропонується метод VAE++ESDD, який використовує поступове навчання та дворівневе ансамблювання: ансамбль варіаційних автокодувальників (VAE) для прогнозування аномалій, а також ансамбль детекторів дрейфу концепцій. Оцінка ефективності методу VAE++ESDD показує його чутливість до виявлення аномалій.

В [23] пропонується алгоритм визначення динамічного порогу виявлення дрейфу, на відмінну від існуючих методів зосереджених на аналізі чутливої тестової статистики, які розглядають поріг виявлення як фіксований гіперпараметр. В ньому вдосконалено існуючі системи виявлення дрейфу новою фазою порівняння, щоб визначити, як слід коригувати поріг. Зростання масштабів застосування IoT-систем робить безумовними необхідності застосування потужних, масштабованих та адаптивних систем виявлення вторгнень [24].

Таким чином, приведений стан досліджень обґрунтовує окремі складові, однак для задачі IoT необхідна узгоджена комплексна модель, яка починається зі схеми журналу аудиту БД, формує віконні ознаки, виконує робастне багатовимірне оцінювання, адаптивно визначає поріг і зберігає пояснювальну подію аномалії.

Об'єктом дослідження є процес доступу IoT-пристроїв, шлюзів, сервісів і операторів до бази даних прикладної системи. Предметом дослідження є методи формування профілю нормальних SQL-транзакцій та виявлення відхилень за подіями аудиту СУБД. Вхідними даними є завершені події читання, запису, автентифікації, помилки й відкати, доповнені ідентифікатором суб'єкта, роллю, цільовим об'єктом, тривалістю, кількістю рядків та обсягом результату.

Потрібно для кожного завершеного часового вікна визначити, чи узгоджується спільна зміна ознак із профілем суб'єкта. Виходом є двійкове рішення про аномалію, неперервна оцінка небезпеки, ключова ознака відхилення та запис anomaly_event, пов'язаний із версією профілю й первинним журналом. Метод має працювати без міток атак під час побудови профілю, бути стійким до невеликої частки забруднених калібрувальних спостережень і виявляти як одиничні різкі дії, так і послідовності слабких відхилень.

Роль IoT-системи полягає у формуванні специфічних регулярностей: періодичності телеметрії, сталого співвідношення операцій читання й запису, обмеженої множини таблиць для класу пристрою та характерного обсягу результатів. Компрометація датчика, шлюзу або сервісного облікового запису змінює ці закономірності на рівні транзакцій БД. Отже, метод не аналізує радіосигнал чи мережевий пакет безпосередньо, а виявляє наслідки компрометації на рівні доступу до даних.

Дослідницька гіпотеза полягає в тому, що поєднання робастного багатовимірного профілю, адаптивного хвостового порога і CUSUM забезпечить низьку частоту хибних сигналів за нормальної зміни навантаження та збереже подієву повноту для повільної ексфільтрації. Перевірка має окремо оцінювати точність, подієву повноту, затримку першого сигналу, час побудови профілів, потокову затримку й обсяг стану.

ВИКЛАД ОСНОВНОГО МАТЕРІАЛУ

Модель події аудиту.

Нехай кожна завершена операція СУБД утворює подію аудиту

$$e_j = \langle id_j, t_j, s_j, r_j, o_j, d_j, \ell_j, n_j, b_j, c_j, tx_j \rangle, \quad (1)$$

де id_j – ідентифікатор події; t_j – час; s_j – суб'єкт (IoT-пристрій, шлюз, обліковий запис або сервіс); r_j – роль БД; o_j – тип операції; d_j – цільовий об'єкт БД; ℓ_j – тривалість виконання; n_j – кількість прочитаних або змінених рядків; b_j – обсяг повернутих даних; c_j – код завершення; tx_j – ідентифікатор транзакції.

Події групуються для кожного суб'єкта s у неперетинні часові вікна

$$W_t = [t\Delta, (t+1)\Delta), \quad (2)$$

де Δ – тривалість вікна. У проведеному експерименті $\Delta = 5$ хв.

Логічний контур оброблення та пов'язані сутності БД наведено на рисунку 1.

Рис. 1: Контур виявлення та логічні сутності бази даних

Рекомендована схема зберігання містить чотири основні сутності (таблиця 1).

Таблиця 1.

Логічні сутності зберігання журналу, профілів і сигналів

Сутність	Первинний ключ	Основні атрибути	Призначення
subject	subject_id	device_id, db_role, service_name, active	Ідентифікація пристрою, ролі або сервісу
audit_event	event_id	ts, subject_id, operation, object_id, duration_ms, rows_affected, bytes_returned, result_code, tx_id	Незмінний журнал операцій СУБД
profile	(subject_id, valid_from)	center, covariance, tail_parameters, threshold, profile_version	Параметри нормальної поведінки
anomaly_event	anomaly_id	subject_id, window_start, score, threshold, cusum, severity, key_feature, status	Результат виявлення та стан розслідування

Зовнішні ключі audit_event.subject_id і anomaly_event.subject_id посилаються на subject. Версіонування профілю дає змогу відтворити рішення, навіть якщо нормальна поведінка згодом змінилася. Таблиця anomaly_event не повинна замінювати первинний журнал: вона містить лише аналітичний результат і посилання на відповідний часовий інтервал.

Вектор ознак.

Для суб'єкта *s* та вікна *W_t* формується вектор

$$\mathbf{x}_{s,t} = \left[n_q, \rho_w, \rho_e, Q_{0.95}(\ell), \sum n, \sum b, H_{D,s,t}, n_a\right]^T, \tag{3}$$

компоненти якого наведено в таблиці 2.

Таблиця 2.

Ознаки віконного профілю доступу до БД

Позначення	Ознака	Інтерпретація
n_q	кількість запитів	Інтенсивність роботи суб'єкта
ρ_w	частка INSERT, UPDATE, DELETE	Зміна співвідношення читання і запису
ρ_e	частка помилок та відкатів	Збої, підбір параметрів або нештатні транзакції
Q_0.95(ℓ)	95-й процентиль затримки	Важкі запити, блокування або перевантаження
Σn	кількість опрацьованих рядків	Масове читання чи руйнівне оновлення
Σb	обсяг повернутих даних	Ознака ексфільтрації
H_D,s,t	нормована ентропія таблиць	Ширина охоплення схеми БД
n_a	невдалі автентифікації	Підбір облікових даних або помилки конфігурації

Нехай $\mathcal{D}_s = \{d_1, \dots, d_{m_s}\}$ – множина таблиць, дозволених профілем суб'єкта s, а $m_s = |\mathcal{D}_s|$. Для непорожнього вікна $p_{k,s,t} = n_{k,s,t}/n_{q,s,t}$, де $n_{k,s,t}$ – кількість звернень до таблиці d_k. Нормована ентропія цільових таблиць визначається з явним опрацюванням крайового випадку:

$$H_{D,s,t} = \begin{cases} -\dfrac{1}{\log m_s} \displaystyle\sum_{k=1}^{m_s} p_{k,s,t} \log p_{k,s,t}, & m_s > 1 \ \wedge \ n_{q,s,t} > 0, \\ 0, & m_s \le 1 \ \vee \ n_{q,s,t} = 0, \end{cases} \tag{4}$$

де прийнято $0\log 0 = 0$. Отже, якщо профіль містить лише одну таблицю або у вікні немає запитів, $H_{D,s,t} = 0$ без ділення на $\log 1 = 0$. Якщо звернення розподілені між кількома об'єктами, значення належить інтервалу $[0,1]$ і зростає зі збільшенням рівномірності охоплення схеми БД.

Ознаки мають різні масштаби та межі. Для додатних лічильників застосовано $\log(1+x)$, а для часток – перетворення logit. Щоб значення 0 і 1 не спричиняли появу $-\infty$ та $+\infty$, часткові ознаки явно обмежуються:

$$\hat{x}_j = \min\{\max(x_j, \varepsilon), 1-\varepsilon\}, \qquad \varepsilon = 10^{-4}, \qquad \operatorname{logit}(\hat{x}_j) = \log\frac{\hat{x}_j}{1-\hat{x}_j}.$$

Після цього вектор перетворених ознак має вигляд

$$\boldsymbol{\phi}(\mathbf{x}) = [\log(1+x_1), \operatorname{logit}(\hat{x}_2), \operatorname{logit}(\hat{x}_3), \log(1+x_4), \log(1+x_5), \log(1+x_6), \operatorname{logit}(\hat{x}_7), \log(1+x_8)]^T. \tag{5}$$

Робастний профіль.

Для кожного суб'єкта використовується калібрувальна множина вікон, які вважаються переважно нормальними. Нехай вона містить N перетворених векторів $\mathbf{z}_i = \boldsymbol{\phi}(\mathbf{x}_i)$, а $p = 8$ – кількість ознак. MCD знаходить підмножину H^* розміру h, для якої визначник коваріаційної матриці мінімальний:

$$H^* = \arg\min_{H:|H|=h} \det(\operatorname{Cov}\{\mathbf{z}_i : i \in H\}). \tag{6}$$

За цією підмножиною обчислюються робастний вектор центра $\tilde{\boldsymbol{\mu}}_s$ і коваріаційна матриця $\tilde{\boldsymbol{\Sigma}}_s$ профілю суб'єкта. Щоб уникнути нестійкого обернення за високої кореляції ознак, застосовується регуляризація:

$$\tilde{\boldsymbol{\Sigma}}_{s,\lambda} = (1-\lambda)\tilde{\boldsymbol{\Sigma}}_s + \lambda\frac{\operatorname{tr}(\tilde{\boldsymbol{\Sigma}}_s)}{p}\mathbf{I}, \quad 0 \le \lambda < 1. \tag{7}$$

Інтегральна оцінка аномальності дорівнює робастній квадратній відстані Махаланобіса:

$$S_{s,t} = (\mathbf{z}_{s,t} - \tilde{\boldsymbol{\mu}}_s)^T \tilde{\boldsymbol{\Sigma}}_{s,\lambda}^{-1} (\mathbf{z}_{s,t} - \tilde{\boldsymbol{\mu}}_s). \tag{8}$$

На відміну від суми незалежних Z-оцінок, вираз (8) враховує коваріацію. Наприклад, одночасне помірне зростання кількості запитів, обсягу читання й ентропії таблиць може мати високу спільну оцінку, навіть якщо жодна окрема ознака не перевищила жорсткий поріг.

Адаптивний поріг POT.

На калібрувальних оцінках *S* вибирається початковий високий рівень *u*, наприклад 97-й процентиль. Для перевищень $Y = S - u$, $S > u$, оцінюються параметри форми ξ та масштабу β узагальненого розподілу Парето:

$$P(Y \le y \mid Y > 0) = 1 - \left(1 + \xi \frac{y}{\beta}\right)^{-\frac{1}{\xi}}. \quad (9)$$

Нехай *N_u* – кількість перевищень серед *N* калібрувальних оцінок, а *q* – допустимий хвостовий ризик. Тоді поріг сигналу визначається:

$$\tau = u + \frac{\beta}{\xi}\left[\left(\frac{N_u}{Nq}\right)^{\xi} - 1\right], \quad \xi \neq 0. \quad (10)$$

Для $\xi \to 0$ використовується границя

$$\tau = u + \beta \log\left(\frac{N_u}{Nq}\right). \quad (11)$$

У потоковому режимі параметри хвоста періодично оновлюються лише за спостереженнями, які не були позначені як аномальні. Це запобігає швидкому «навчанню на атаці». Версія порога та параметри *(u, ξ, β, q)* зберігаються у таблиці profile.

Накопичення слабких відхилень.

Різкий сплеск зазвичай виконує умову $S_{s,t} > \tau_t$. Повільний витік даних може складатися з послідовності помірних відхилень, кожне з яких окремо залишається нижче порога. Для такого режиму оцінка стандартизується відносно робастних характеристик калібрувального розподілу:

$$Z_{s,t} = \frac{S_{s,t} - \text{median}(S_S^{train})}{1.4826\,\text{MAD}(S_S^{train}) + \varepsilon_S}, \qquad \varepsilon_S = 10^{-6}. \quad (12)$$

Одностороння накопичувальна статистика CUSUM:

$$G_{s,t} = \max\{0, G_{s,t-1} + Z_{s,t} - \kappa\}, \qquad G_{s,0} = 0, \quad (13)$$

де κ визначає мінімальне систематичне відхилення, яке має накопичуватися. Остаточне правило:

$$A_{s,t} = \begin{cases} 1, & S_{s,t} > \tau_t \ \vee \ G_{s,t} > h, \\ 0, & \text{інакше.} \end{cases} \quad (14)$$

Після сигналу *G_s,t* скидається, щоб одна тривала подія не створювала необмежену кількість дубльованих сповіщень. Рівень критичності визначається:

$$R_{s,t} = 1 - \exp\left[-\max\left(\frac{S_{s,t}}{\tau_t}, \frac{G_{s,t}}{h}\right)\right]. \quad (15)$$

Для пояснення сигналу виконується декомпозиція відхилення в декорельованому (вибіленому) просторі ознак. Якщо **L** є матрицею Холєцького регуляризованої коваріаційної матриці $\widetilde{\boldsymbol{\Sigma}}_{s,\lambda}$, то

$$\mathbf{c}_{s,t} = \mathbf{L}^{-1}\left(\mathbf{z}_{s,t} - \widetilde{\boldsymbol{\mu}}_s\right), \qquad j^* = \underset{j}{\text{argmax}}\left|c_{s,t,j}\right|. \quad (16)$$

Індекс j^* визначає ключову ознаку відхилення, яка записується у поле key_feature. Це не є повним причинним поясненням, але скорочує первинний аналіз події.

Алгоритм реалізації.

Метод працює у двох режимах. Перший формує версіонований профіль нормальної поведінки суб'єкта за калібрувальною частиною журналу (таблиця 3). Другий асинхронно перевіряє кожне завершене часове вікно, оновлює накопичувальну статистику та зберігає пояснюваний сигнал про аномалію (таблиця 4).

Таблиця 3.

Алгоритму формування робастного профілю нормальної поведінки

Кроки алгоритму 1
Require: $C_s = \{x_{s,i}\}_{i=1}^{N}$ – калібрувальні вікна; $p = 8$; λ, q, α – параметри регуляризації та хвоста
Ensure: $P_s = \{\tilde{\mu}_s, \tilde{\Sigma}_s, \lambda, u, \xi, \beta, \tau, G_s = 0\}$ – активний профіль суб'єкта
1: Begin
2: Зчитати завершені події аудиту з `audit_event` без зміни первинних записів
3: Згрупувати події за `subject_id` і часовими вікнами тривалості Δ
4: Для кожного вікна обчислити вектор ознак $x_{s,i}$ за (3)
5: Виконати перетворення $z_{s,i} = \varphi(x_{s,i})$ за (5)
6: Оцінити робастний центр $\tilde{\mu}_s$ та коваріацію $\tilde{\Sigma}_s$ методом MCD
7: Регуляризувати $\tilde{\Sigma}_s$ і обчислити калібрувальні оцінки $S_{s,i}$
8: Встановити початковий рівень u як квантиль α оцінок $S_{s,i}$
9: Апроксимувати перевищення $S_{s,i} - u$ розподілом GPD і оцінити ξ, β
10: Обчислити поріг τ для хвостового ризику q за (12)
11: Зберегти параметри, версію та інтервал дії профілю в `profile`
12: Return P_s
13: End

Таблиця 4.

Алгоритм потокового виявлення аномального вікна транзакцій

Кроки алгоритму 2
Require: $W_{s,t}$ – завершене вікно; активний профіль P_s; параметри κ, h, ε
Ensure: $A_{s,t}$ – рішення; $R_{s,t}$ – критичність; j^* – ключова ознака; запис `anomaly_event` за наявності сигналу
1: Begin
2: Обчислити $x_{s,t}$ з подій вікна $W_{s,t}$ та виконати $z_{s,t} = \varphi(x_{s,t})$
3: Завантажити версію P_s, чинну для моменту t
4: if профіль відсутній або прострочений **then** позначити `cold_start` і перейти до кроку 15
5: Обчислити робастну оцінку $S_{s,t}$ за (8)
6: Обчислити нормоване перевищення $Z_{s,t}$ та оновити $G_{s,t}$ за (13)
7: Встановити $A_{s,t} = \mathbb{I}\{S_{s,t} > \tau_t \vee G_{s,t} > h\}$
8: if $A_{s,t} = 1$ **then**
9: Обчислити критичність $R_{s,t}$ за (15)
10: Визначити ключову ознаку j^* за (16)
11: Записати `anomaly_event` і скинути $G_s \leftarrow 0$
12: else додати допустиму оцінку до буфера адаптації хвоста POT
13: Зберегти версію профілю, оцінку та час оброблення
14: Return $\{A_{s,t}, R_{s,t}, j^*\}$
15: End

Обчислювальна складність оцінювання одного вікна після побудови профілю визначається множенням матриці розміру $p \times p$ і становить $O(p^2)$. Для $p = 8$ це мала фіксована вартість. У пам'яті для одного профілю зберігаються центр, коваріаційна матриця, параметри хвоста та поточне значення CUSUM, тобто $O(p^2)$.

Практична реалізація має відокремлювати збір журналу від аналітичного контуру. Запис аудиту виконується в межах штатного механізму СУБД, а агрегація та оцінювання – асинхронним процесом або потоком змін даних. Це зменшує вплив аналізу на затримку транзакцій. Для захисту від модифікації журнал доцільно зробити додавальним, обмежити права UPDATE і DELETE та вести контроль цілісності.

ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ

Мета та перевірювані припущення.

Експеримент мав перевірити не лише здатність методу класифікувати аномальні вікна, а й придатність алгоритму до асинхронного контролю журналу СУБД. Сформульовано три перевірювані припущення. По-перше, багатовимірний профіль має зменшити кількість хибних сигналів порівняно з незалежними порогами для ознак. По-друге, CUSUM має підвищити подієву повноту для слабких тривалих атак, навіть якщо не кожне їх вікно перевищує поріг POT. По-третє, час оцінювання завершеного вікна має бути значно меншим за тривалість вікна $\Delta = 5$ хв, а обсяг стану профілю – лінійно залежати від кількості суб'єктів.

Одиницею класифікації є пара «суб'єкт – завершене часове вікно». Калібрувальна частина використовується тільки для побудови нормального профілю й порогів. Мітки тестової частини відкриваються лише на етапі обчислення метрик. Такий поділ виключає безпосереднє налаштування параметрів за тестовими аномаліями, хоча синтетична природа даних залишається обмеженням зовнішньої валідності.

Структура експериментального стенда.

Експеримент виконано у контейнеризованому середовищі x86-64. Стенд є програмним: він не імітує фізичний рівень датчиків, а відтворює ділянку після надходження SQL-подій до контуру аудиту. Детермінований генератор формує агреговані записи, реляційне сховище зберігає їх у таблиці audit_window, модуль детектора читає завершені вікна, а блок оцінювання зіставляє сигнали з еталонними мітками. Основні компоненти наведено в таблиці 5, а потік даних – на рисунку 2.

У сховищі створювалася таблиця audit_window з первинним ключем (subject_id, window_no) та індексом (split, subject_id, window_no). Один запис містив ідентифікатор суб'єкта, номер вікна, ознаку частини набору, еталонну мітку, тип сценарію та вісім ознак (3). Вимірювання швидкості сховища виконувалися окремо від оцінювання детектора, щоб не змішувати витрати SQL-вводу-виводу та математичного обчислення.

Таблиця 5.

Конфігурація програмного експериментального стенда

Компонент	Конфігурація	Призначення
Обчислювальне середовище	AMD EPYC 9V74; доступно 9 логічних процесорів; 15,93 ГіБ RAM	Генерація даних, побудова профілів, оцінювання
Операційна система	Linux 6.12.13, x86-64	Ізольований запуск повторних вимірювань
Програмне середовище	Python 3.12.13; NumPy 2.3.5; pandas 2.2.3; SciPy 1.17.0; scikit-learn 1.8.0	Моделювання, MCD, POT, базові методи й метрики
Реляційне сховище	SQLite 3.50.4; journal_mode=WAL; synchronous=NORMAL	Перевірка запису та вибірки агрегованих вікон
Відтворюваність	seed = 20260726; однопотокове навчання базових моделей	Однакова генерація профілів і атак між запусками

Рис. 2: Структура програмного експериментального стенда

Формування нормального навантаження.

Через відсутність у вихідній роботі первинних журналів або програмного коду числовий експеримент побудовано заново. Згенеровано профілі 60 суб'єктів, які представляють IoT-шлюзи, сервісні облікові записи або класи пристроїв. Для кожного суб'єкта створено 240 калібрувальних і 240 тестових п'ятихвилинних вікон. Отже, калібрувальна та тестова частини містять по 14 400 вікон, разом – 28 800.

Нормальні ознаки формувалися у перетвореному просторі як корельовані випадкові величини з індивідуальним центром для кожного суб'єкта. Матриця кореляцій задавала, зокрема, додатний зв'язок між частотою запитів, кількістю опрацьованих рядків і обсягом повернених даних, а також між часткою помилок і невдалими автентифікаціями. До кожного ряду додано синусоїдальну компоненту з періодом 48 вікон і слабкий лінійний дрейф. Так відтворено повторювану зміну навантаження в межах десятиденного фрагмента без різкого розриву між калібрувальною та тестовою частинами.

Після оберненого перетворення лічильники обмежувалися невід'ємними значеннями, а частки – інтервалом *[0, 1]*. Калібрувальні вікна не містили навмисно внесених атак. Передавання даних між етапами виконувалося у незмінному порядку subject_id, window_no, тому затримку сигналу для послідовної атаки можна було виміряти без відновлення часової шкали.

Сценарії аномальних транзакцій.

До тестової частини внесено 672 аномальні вікна. Точкові атаки розміщувалися без повторного використання одного вікна; тривалі епізоди займали неперервні послідовності та не перетиналися з точковими атаками. Зміни вносилися в перетвореному просторі, а їхня амплітуда мала випадкове відхилення до заданого профілю сценарію. Це запобігало появі повністю однакових аномалій. Зміст проведених тестів приведені в таблиці 6.

Сплеск запису моделює помилку пристрою або зловживання сервісним обліковим записом, за якого за короткий час змінюється значна кількість рядків. Масове читання відповідає спробі вивантаження телеметрії з кількох таблиць. Підбір облікових даних створює серію невдалих дій і помилок. Руйнівне оновлення поєднує високу частку модифікацій, велику область впливу та зростання затримки. Для повільного витоку амплітуда зростала від 0,80 до 1,15 від базового вектора зміни протягом 12 вікон; окреме вікно могло залишатися нижче τ, але послідовність мала накопичувати CUSUM.

Таблиця 6.

Склад і реалізація сценаріїв аномальної поведінки

Сценарій	Кількість	Тривалість	Основні змінені ознаки
Сплеск операцій запису	130	1 вікно	query_count, write_share, rows_touched, затримка
Масове читання	130	1 вікно	bytes_returned_kb, table_entropy, query_count, затримка
Підбір облікових даних	110	1 вікно	error_share, failed_auth_count, затримка
Руйнівне оновлення	110	1 вікно	write_share, rows_touched, error_share, затримка
Повільний витік даних	16 подій / 192 вікна	12 вікон	Помірне наростання bytes_returned_kb, query_count, rows_touched, table_entropy

Параметри та базові методи.

Для запропонованого методу використано h/N = 0,78 у MCD, регуляризацію λ = 0,08, початковий рівень POT на 97-му процентилі та хвостовий ризик q = 0,0015. За калібрувальною вибіркою отримано u = 19,35, ξ = 0,49, β = 5,05 і τ = 53,76. Для CUSUM встановлено κ = 1,1, h = 8.

Поряд з абляційними варіантами до порівняння додано два відомі методи машинного навчання без учителя. Їх не оптимізували за тестовими мітками: поріг для Isolation Forest і LOF визначався за 99,5-м процентилем їхніх оцінок на калібрувальній частині. Вибрані параметри наведено в таблиці 7.

Таблиця 7.

Параметри порівнюваних методів

Метод	Профіль і параметри	Правило формування сигналу
Modified Z-score	Медіана, MAD; поріг 4,0; нижня межа масштабу 0,10	Максимальний модуль одновимірної оцінки перевищує 4,0
Isolation Forest	200 дерев; max_samples=auto; окрема модель для суб'єкта	Оцінка вища за 99,5-й процентиль калібрувальних оцінок
LOF	20 сусідів; novelty=True; окрема модель для суб'єкта	Оцінка вища за 99,5-й процентиль калібрувальних оцінок
MCD + POT	*h/N = 0,78*; *λ = 0,08*; *q = 0,0015*	$S_{s,t} > \tau$
MCD + POT + CUSUM	Параметри MCD + POT; *κ = 1,1*; *h = 8*	Правило (14)

Порівняння MCD + POT із повною конфігурацією відокремлює внесок накопичення CUSUM. Modified Z-score показує наслідок незалежного контролю ознак. Isolation Forest і LOF дають зовнішній орієнтир відносно поширених ML-методів, але це порівняння не є повним пошуком гіперпараметрів. Автоенкодер не включено до числового експерименту, оскільки 240 нормальних вікон на суб'єкта недостатньо для обґрунтованого вибору архітектури та окремого навчання 60 нейронних моделей.

Протокол вимірювання ефективності.

Якість виявлення оцінювали за матрицею помилок на всіх 14 400 тестових вікнах. Для точкових і тривалих атак обчислювали Precision, Recall, F1 і частоту хибних спрацьовувань:

$$\text{Precision} = \frac{TP}{TP + FP}, \qquad \text{Recall} = \frac{TP}{TP + FN}, \tag{17}$$

$$F1 = 2\frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}, \qquad FAR = \frac{FP}{FP + TN}. \tag{18}$$

Для тривалої атаки додатково вимірювали частку подій, у межах яких з'явився хоча б один сигнал, і затримку першого сигналу:

$$R_{\text{event}} = \frac{N_{\text{detected events}}}{N_{\text{events}}}, \qquad D_{\text{med}} = \text{median}_e\left(t_{\text{alert}}^{(e)} - t_{\text{start}}^{(e)} + 1\right)\Delta. \tag{19}$$

Вимірювання обчислювальної ефективності проводили монотонним таймером високої роздільної здатності. Побудову 60 профілів повторювали 5 разів; пакетне оцінювання 14 400 вікон – 25 разів після прогрівання; потоковий прохід – 3 рази, тобто затримки обчислено за 43 200 спостереженнями. Для кожної серії використовували медіану, а для профілювання та потокової затримки – також 95-й або 99-й процентиль. Пропускну здатність і процентиль затримки визначали як

$$\Theta = \frac{N_{\text{windows}}}{T_{\text{processing}}}, \qquad L_\gamma = Q_\gamma(l_1, l_2, \ldots, l_N). \tag{20}$$

Масштабованість пакетного оброблення перевіряли на 1 800, 3 600, 7 200 і 14 400 вікнах; кожну точку повторювали 15 разів. Числовий обсяг стану одного профілю визначали за фактичним розміром масивів float64: вектор центра, матриця точності та три скаляри для порога й стану CUSUM:

$$M_{\text{profile}} = 8(p + p^2 + 3) \text{ байт.} \quad (21)$$

Для $p = 8$ це 600 байт на суб'єкта без накладних витрат об'єктів мови програмування. Швидкість запису та читання audit_window вимірювали п'ятьма незалежними створеннями файлової БД. У кожному повторі 28 800 записів вставлялися однією транзакцією, після чого читалися 14 400 тестових рядків із сортуванням за суб'єктом і часом.

Межа вимірювання є принциповою: час потокового детектора включає перетворення восьми вже агрегованих ознак, пошук профілю, обчислення квадратичної форми та оновлення CUSUM. Він не включає виконання прикладних SQL-транзакцій, штатний механізм аудиту промислової СУБД, передавання журналу мережею та побудову ознак із сирих подій. Тому наведені часові значення характеризують аналітичний модуль після агрегації, а не повні накладні витрати контролю БД.

Загальні результати та обговорення.

Результати для всіх тестових вікон наведено в таблиці 8.

Таблиця 8.

Порівняння методів на синтетичному журналі аудиту

Метод	TP	FP	FN	TN	Precision, %	Recall, %	F1, %	FAR, %
Modified Z-score	475	266	197	13 462	64,10	70,68	67,23	1,94
Isolation Forest	459	52	213	13 676	89,82	68,30	77,60	0,38
LOF	508	89	164	13 639	85,09	75,60	80,06	0,65
MCD + POT	410	23	262	13 705	94,69	61,01	74,21	0,17
MCD + POT + CUSUM	486	76	186	13 652	86,48	72,32	78,77	0,55

Одновимірний Modified Z-score має найвищу частоту хибних спрацьовувань, оскільки перевіряє кожну ознаку окремо й не враховує нормальні кореляції. MCD + POT є найбільш консервативним: Precision становить 94,69 %, а FAR – 0,17 %, проте Recall зменшується до 61,01 %. Додавання CUSUM підвищує кількість правильно виявлених аномальних вікон з 410 до 486, збільшує Recall на 11,31 процентного пункту та F1 на 4,56 пункту, але створює 53 додаткові хибні сигнали.

Серед фіксованих ML-конфігурацій LOF отримав найбільший F1 – 80,06 %, що на 1,29 пункту вище за повний запропонований метод. Водночас запропонований метод має дещо вищу Precision (86,48 проти 85,09 %) і нижчий FAR (0,55 проти 0,65 %), а його поріг має статистичну інтерпретацію через хвостовий ризик. Isolation Forest забезпечив F1 77,60 % за FAR 0,38 %. Отже, експеримент не підтверджує універсальної переваги MCD + POT + CUSUM над LOF; він показує конкурентний баланс якості, інтерпретованого порога та накопичення тривалих відхилень без навчання нейронної мережі.

Графічне порівняння наведено на рисунку 3.

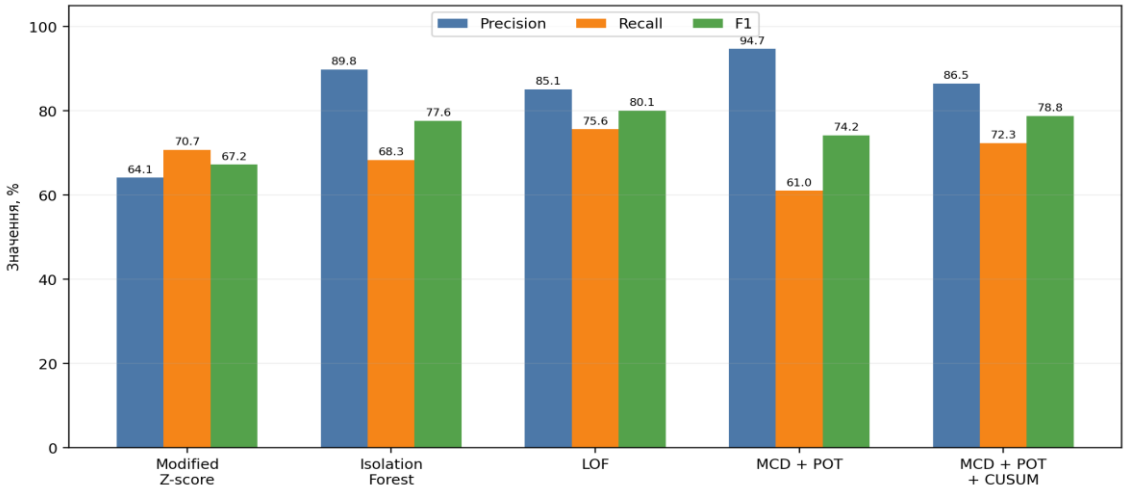

Рис. 3: Порівняння Precision, Recall і F1

Результати за типами аномалій.

Повнота запропонованого методу для окремих типів подій наведена в таблиці 9.

Таблиця 9.

Виявлення за типами аномальних дій у БД

Тип аномалії	**Аномальних вікон**	**Виявлено**	**Recall за вікнами, %**
Масове читання	130	128	98,46
Підбір облікових даних	110	110	100,00
Руйнівне оновлення	110	108	98,18
Сплеск запису	130	119	91,54
Повільний витік даних	192	21	10,94

Низька віконна повнота для повільного витоку даних потребує правильної інтерпретації. Після першого сигналу CUSUM скидається, тому метод не прагне позначити кожне наступне вікно тієї самої тривалої події. Для реагування достатньо хоча б одного сигналу в межах послідовності. Із 16 змодельованих подій повільного витоку даних виявлено 13, тобто подієва повнота становить 81,25 %. Медіанна затримка першого сигналу дорівнює шести вікнам, або 30 хв.

На рисунку 4 показано поведінку оцінки та CUSUM для одного суб'єкта. Різкі аномалії перетинають τ. У затіненому інтервалі значення $S_{s,t}$ залишаються нижчими за поріг, але послідовність помірних відхилень збільшує $G_{s,t}$ вище $h = 8$, що створює сигнал.

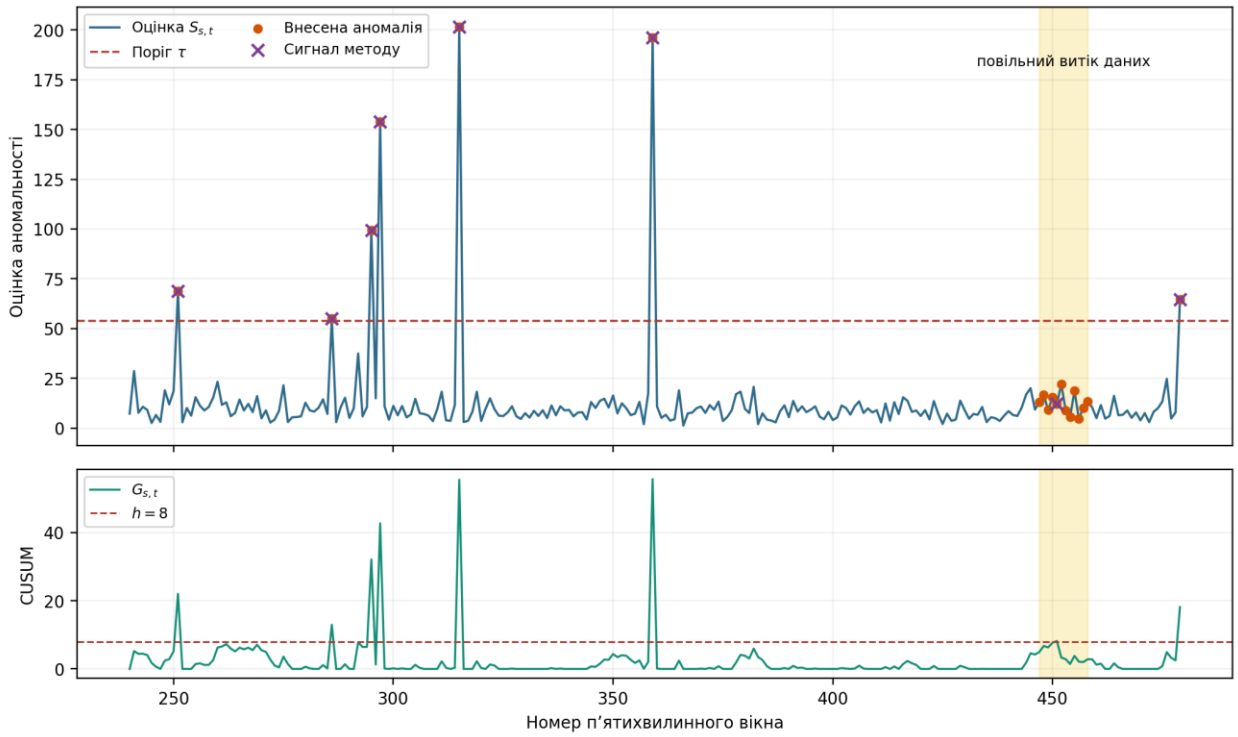

Рис. 4: Оцінка аномальності та CUSUM для суб'єкта з повільним витоком даних

Чутливість параметрів CUSUM.

Параметри к і h визначають компроміс між швидким накопиченням слабких відхилень і кількістю хибних сигналів. Щоб перевірити стійкість висновків, після фіксації оцінок $S_{s,t}$ і порога τ повторно виконано CUSUM для п'яти конфігурацій. Інші елементи контуру, тестова вибірка та порядок вікон не змінювалися. Результати наведено в таблиці 10.

Зменшення h до 6 підвищує подієву повноту повільного витоку до 93,75 %, але збільшує FAR до 0,84 %. За h = 10 хибних сигналів менше, проте виявляється лише 75 % тривалих подій. Зміна к дає аналогічний компроміс: мале значення швидше накопичує відхилення, а велике пригнічує слабкі зміни. Базову пару к = 1,1; h = 8 обрано як середній режим з FAR нижче 0,6 % і подієвою повнотою понад 80 %. Максимальний F1 у

цій локальній перевірці має к = 1,3; h = 8, але різниця з базовою конфігурацією становить лише 0,11 процентного пункту та супроводжується втратою однієї з шістнадцяти подій витоку.

Таблиця 10.

Аналіз чутливості параметрів CUSUM

Параметри	Precision, %	Recall, %	F1, %	FAR, %	Подієва повнота витоку, %
к = 1,1; h = 6	81,41	75,60	78,40	0,84	93,75
к = 1,1; h = 8	86,48	72,32	78,77	0,55	81,25
к = 1,1; h = 10	89,64	69,49	78,29	0,39	75,00
к = 0,9; h = 8	82,40	72,47	77,12	0,76	87,50
к = 1,3; h = 8	88,07	71,43	78,88	0,47	75,00

Медіанна затримка для всіх п’яти конфігурацій серед уже виявлених тривалих подій залишилася рівною шести вікнам. Цей результат не означає однакової поведінки: суворіші параметри пропускають більше епізодів, а медіана обчислюється тільки для подій, у яких сигнал з’явився. Тому затримку необхідно аналізувати разом із R_{event}.

Обчислювальна ефективність.

Результати вимірювання часу та обсягу стану наведено в таблиці 11. Значення округлено після обчислення медіани, тому пропускна здатність не повинна повторно обчислюватися з уже округленого часу.

Таблиця 11.

Результати вимірювання обчислювальної ефективності

Операція	Обсяг і повторення	Основний результат	Додатковий показник
Побудова профілів MCD	60 суб’єктів × 240 вікон; 5 повторів	медіана 1 100,80 мс	P95 1 158,57 мс
Пакетне оцінювання	14 400 вікон; 25 повторів	медіана 19,90 мс	723 673 вікон/с
Потокове оцінювання	3 проходи, 43 200 вимірів	P50 22,1 мкс; P95 23,3 мкс	P99 37,7 мкс; 43 801 вікон/с
Числовий стан профілів	*p = 8*; 60 суб’єктів	600 байт/профіль	35,16 КіБ разом
Запис audit_window	28 800 рядків; 5 повторів	медіана 34,99 мс	822 992 рядків/с
Читання тестової частини	14 400 рядків; 5 повторів	медіана 15,83 мс	909 505 рядків/с

Різниця між пакетною та потоковою пропускною здатністю пояснюється способом оброблення. У пакетному режимі перетворення та квадратичні форми обчислюються векторизовано для групи суб’єкта. Потоковий режим виконує пошук профілю та оновлення стану для кожного завершеного вікна окремо, тому його показник нижчий, але саме він ближчий до робочого контуру.

За P95 потокове оцінювання займає приблизно 0,0000078 % від п’ятихвилинного інтервалу. Це показує великий часовий резерв аналітичної частини для заданого p = 8. Водночас цей відсоток не характеризує вплив аудиту на прикладні транзакції, бо вимірювання починається після формування агрегованого вектора.

Медіанний пакетний час збільшився з 2,25 мс для 1 800 вікон до 4,14; 9,19 та 19,69 мс для 3 600, 7 200 і 14 400 вікон відповідно. Залежність є близькою до лінійної; невелике зменшення пропускної здатності на найбільшому пакеті може бути пов’язане з кешем і накладними витратами групування. Результати масштабування та розподіл потокової затримки показано на рисунку 5.

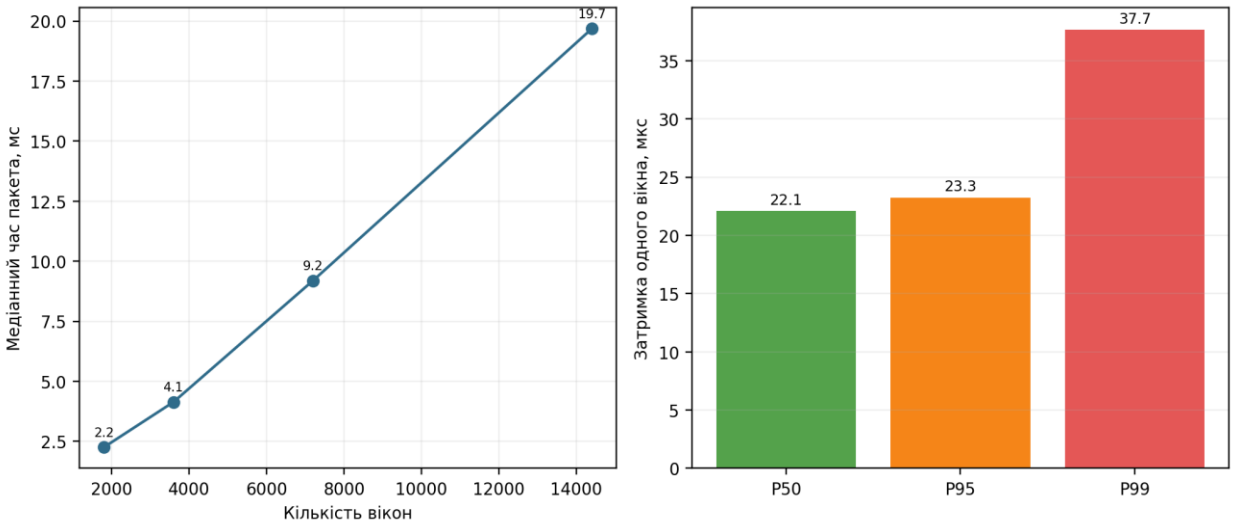

Рис. 5: Масштабованість пакетного оцінювання та потокова затримка

Вимірювання реляційного сховища підтверджує, що збереження 28 800 агрегованих записів не є вузьким місцем саме для цього стенда. Проте одна пакетна транзакція SQLite не відтворює конкурентне навантаження PostgreSQL, MySQL або MariaDB. Отримані 823 тис. записів/с слід трактувати як контроль працездатності експериментального контуру, а не як прогноз продуктивності виробничої БД.

Практична інтерпретація.

У контексті баз даних важливо розділяти технічний сигнал і рішення про інцидент. Висока оцінка може бути наслідком легітимного перенесення даних, резервного копіювання або зміни версії сервісу. Тому запис anomaly_event має містити статус розгляду, посилання на зміну конфігурації та, за можливості, ідентифікатор запланованої роботи. Підтверджені штатні зміни не слід одразу додавати до профілю; оновлення має виконуватися контрольовано та з новою версією.

Профіль за subject_id доцільний для стабільних IoT-шлюзів і сервісних облікових записів. Якщо кількість пристроїв велика або окремий пристрій створює мало подій, профіль можна будувати за роллю чи класом пристрою, а індивідуальне відхилення оцінювати додатковим рівнем. Поле db_role дає змогу узгодити поведінковий контроль із RBAC: право доступу визначає допустиму область, а профіль – типовий спосіб використання цього права.

Ключова ознака відхилення допомагає сформувати первинну категорію: bytes_returned_kb і table_entropy вказують на можливе масове читання; write_share і rows_touched – на нетипову модифікацію; error_share і failed_auth_count – на проблеми автентифікації або підбір параметрів. Остаточний висновок має враховувати SQL-текст, роль, цільову таблицю, часовий контекст і пов’язані мережеві події.

Обмеження валідності.

Експеримент є синтетичним. Він перевіряє внутрішню узгодженість математичного апарату та порівняння методів на контрольованих сценаріях, але не доводить досягнення таких самих метрик у конкретній виробничій СУБД. Генератор не моделює повну семантику SQL, блокування, плани запитів, конкурентні транзакції, кешування, реплікацію та всі типи дрейфу навантаження.

Не вимірювався накладний час штатного аудиту конкретної СУБД. Запропонована складність $O(p^2)$ і наведені мікросекундні затримки стосуються лише оцінювання вже агрегованого вікна. Витрати на збір журналу, нормалізацію SQL і визначення фактично прочитаних байтів залежать від обраної платформи.

Вимірювання часу виконано в одному контейнеризованому середовищі без фіксування частоти процесора та ізоляції від усіх сусідніх процесів. Повтори й процентилі зменшують вплив випадкових коливань, але для апаратно незалежного висновку потрібні випробування на окремому сервері та малоресурсному шлюзі. Фіксовані конфігурації Isolation Forest і LOF не проходили повного пошуку гіперпараметрів; тому їх результати є орієнтиром, а не остаточним рейтингом алгоритмів.

Для зовнішньої перевірки потрібні реальні журнали PostgreSQL, MySQL, MariaDB або іншої СУБД з анонімізованими ідентифікаторами, відомими періодами штатної роботи та підтвердженими інцидентами. Параметри λ, q, κ, h і Δ слід калібрувати на окремій валідаційній частині. За тривалої експлуатації необхідно окремо контролювати дрейф концепту й оновлювати профілі лише після перевірки зміни режиму. Числові результати цієї роботи не слід подавати як вимірювання реальної інфраструктури.

ВИСНОВКИ З ДАНОГО ДОСЛІДЖЕННЯ І ПЕРСПЕКТИВИ ПОДАЛЬШИХ РОЗВІДОК У ДАНОМУ НАПРЯМІ

Статтю перебудовано навколо задачі захисту баз даних IoT-систем. Об'єктом аналізу визначено журнал аудиту СУБД, а одиницею спостереження – агрегований профіль запитів і транзакцій окремого пристрою, ролі або сервісу. Запропонований вектор ознак охоплює інтенсивність запитів, співвідношення читання та запису, помилки, затримку, кількість опрацьованих рядків, обсяг даних, множину таблиць і невдалі автентифікації.

Математичний апарат складається з робастного оцінювання MCD, регуляризованої відстані Махаланобіса, адаптивного порога POT та CUSUM. Така комбінація усуває дублювання попередніх формул, враховує кореляції між ознаками, не використовує довільний універсальний поріг і дає окремий механізм для слабких тривалих атак.

На синтетичному журналі повний метод отримав Precision 86,48 %, Recall 72,32 % і F1 78,77 % при FAR 0,55 %. Для повільного витоку даних подієва повнота становила 81,25 % із медіанною затримкою 30 хв. Додавання CUSUM збільшило F1 на 4,56 процентного пункту відносно MCD + POT. Водночас фіксована конфігурація LOF отримала F1 80,06 %, тому результат слід трактувати як конкурентність запропонованого методу та перевагу його інтерпретованого порога й накопичувального механізму, а не як універсальне домінування над ML-аналогами.

Медіанний час побудови 60 профілів становив 1,10 с, пакетне оцінювання 14 400 завершених вікон – 19,90 мс, а P95 потокової затримки – 23,3 мкс. Числовий стан 60 профілів займав 35,16 КіБ. Ці показники демонструють низьку вартість аналітичного модуля після агрегації, але не включають накладні витрати штатного аудиту та формування ознак.

Подальша робота має охопити експеримент з реальною СУБД, наскрізне вимірювання впливу аудиту на латентність і пропускну здатність прикладних SQL-транзакцій, профілювання за ролями RBAC, оброблення зміни концепту, а також поєднання статистичної оцінки з ознаками синтаксичного дерева SQL і графом доступу до таблиць.

References

1. Yang Z., Liu X., Li T., Wu D., Wang J., Zhao Y., Han H. A systematic literature review of methods and datasets for anomaly-based network intrusion detection. Computers & Security. 2022. Vol. 116. Article 102675. https://doi.org/10.1016/j.cose.2022.102675
2. Sgueglia A., Di Sorbo A., Visaggio C. A., Canfora G. A systematic literature review of IoT time series anomaly detection solutions. Future Generation Computer Systems. 2022. Vol. 134. P. 170–186. https://doi.org/10.1016/j.future.2022.04.005
3. Jain M., Kaur G., Saxena V. A K-Means clustering and SVM based hybrid concept drift detection technique for network anomaly detection. Expert Systems with Applications. 2022. Vol. 193. Article 116510. https://doi.org/10.1016/j.eswa.2022.116510
4. Soltani N., Rahmani A. M. Artificial intelligence empowered threat detection in the Internet of Things: A systematic review. Concurrency and Computation: Practice and Experience. 2022. Vol. 34, no. 22. e6894. https://doi.org/10.1002/cpe.6894
5. Al-Bakaa A., Al-Musawi B. A new intrusion detection system based on using non-linear statistical analysis and features selection techniques. Computers & Security. 2022. Vol. 122. Article 102906. https://doi.org/10.1016/j.cose.2022.102906
6. 31. DeMedeiros K., Hendawi A., Alvarez M. A survey of AI-based anomaly detection in IoT and sensor networks. Sensors. 2023. Vol. 23, no. 3. Article 1352. https://doi.org/10.3390/s23031352
7. Bovenzi G., Aceto G., Ciuonzo D., Montieri A., Persico V., Pescapé A. Network anomaly detection methods in IoT environments via deep learning: A fair comparison of performance and robustness. Computers & Security. 2023. Vol. 128. Article 103167. https://doi.org/10.1016/j.cose.2023.103167
8. Catillo M., Pecchia A., Villano U. CPS-GUARD: Intrusion detection for cyber-physical systems and IoT devices using outlier-aware deep autoencoders. Computers & Security. 2023. Vol. 129. Article 103210. https://doi.org/10.1016/j.cose.2023.103210
9. Landauer M., Onder S., Skopik F., Wurzenberger M. Deep learning for anomaly detection in log data: A survey. Machine Learning with Applications. 2023. Vol. 12. Article 100470. https://doi.org/10.1016/j.mlwa.2023.100470
10. Li Z., Zhu Y., Van Leeuwen M. A survey on explainable anomaly detection. ACM Transactions on Knowledge Discovery from Data. 2023. Vol. 18, no. 1. Article 23. P. 1–54. https://doi.org/10.1145/3609333
11. Manoharan P., Yin J., Wang H., Zhang Y., Ye W. Insider threat detection using supervised machine learning algorithms. Telecommunication Systems. 2024. Vol. 87, no. 4. P. 899–915. https://doi.org/10.1007/s11235-023-01085-3
12. Jaiswal A., Dwivedi P., Dewang R. K. Handling imbalance dataset issue in insider threat detection using machine learning methods. Computers and Electrical Engineering. 2024. Vol. 120. Article 109726. https://doi.org/10.1016/j.compeleceng.2024.109726
13. Shyaa M. A., Ibrahim N. F., Zainol Z., Abdullah R., Anbar M., Alzubaidi L. Evolving cybersecurity frontiers: A comprehensive survey on concept drift and feature dynamics aware machine and deep learning in intrusion detection systems. Engineering Applications of Artificial Intelligence. 2024. Vol. 137. Article 109143. https://doi.org/10.1016/j.engappai.2024.109143
14. Li B., Gupta S., Müller E. State-transition-aware anomaly detection under concept drifts. Data & Knowledge Engineering. 2024. Vol. 154. Article 102365. https://doi.org/10.1016/j.datak.2024.102365
15. Mašková M., Zorek M., Pevný T., Šmídl V. Deep anomaly detection on set data: Survey and comparison. Pattern Recognition. 2024. Vol. 151. Article 110381. https://doi.org/10.1016/j.patcog.2024.110381
16. Catillo M., Pecchia A., Villano U. MultiCIDS: Anomaly-based collective intrusion detection by deep learning on IoT/CPS multivariate time series. Internet of Things. 2025. Vol. 30. Article 101519. https://doi.org/10.1016/j.iot.2025.101519
17. Rheey J., Park H. Robust hierarchical anomaly detection using feature impact in IoT networks. ICT Express. 2025. Vol. 11, no. 2. P. 358–363. https://doi.org/10.1016/j.icte.2025.02.009
18. Hendaoui F., Meddeb R., Trabelsi L., Ferchichi A., Ahmed R. FLADEN: Federated Learning for Anomaly DEtection in IoT Networks. Computers & Security. 2025. Vol. 155. Article 104446. https://doi.org/10.1016/j.cose.2025.104446
19. Moriano P., Hespeler S. C., Li M., Mahbub M. Adaptive anomaly detection for identifying attacks in cyber-physical systems: A systematic literature review. Artificial Intelligence Review. 2025. Vol. 58, no. 9. Article 283. https://doi.org/10.1007/s10462-025-11292-w
20. Zhou P. A survey of streaming data anomaly detection in network security. PeerJ Computer Science. 2025. Vol. 11. e3066. https://doi.org/10.7717/peerj-cs.3066

21. Yang L., Yao Y., Yang D., Yang W., Hao Y. A generalizable anomaly detection framework with dynamic concept drift suppression for non-stationary time series. Knowledge-Based Systems. 2026. Vol. 337. Article 115380. https://doi.org/10.1016/j.knosys.2026.115380

22. Li J., Malialis K., Panayiotou C. G., Polycarpou M. M. Drift-aware variational autoencoder-based anomaly detection with two-level ensembling. Neurocomputing. 2026. Vol. 676. Article 133061. https://doi.org/10.1016/j.neucom.2026.133061

23. Lu P., Lu J., Liu A., Yu E., Zhang G. Autonomous concept drift threshold determination. Proceedings of the AAAI Conference on Artificial Intelligence. 2026. Vol. 40, no. 29. P. 24079–24087. https://doi.org/10.1609/aaai.v40i29.39586

24. Komal A., Li S. Intrusion detection in the Internet of Things: A comprehensive review of techniques, architectures, datasets, and emerging trends. Sensors. 2026. Vol. 26, no. 11. Article 3405. https://doi.org/10.3390/s26113405