КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

БУДІВНИЦТВА І АРХІТЕКТУРИ

автоматизації і інформаційних технологій

(факультет)

кібербезпеки та комп'ютерної інженерії

(назва випускової кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

ЗДОБУВАЧА СТУПЕНЯ ВИЩОЇ ОСВІТИ БАКАЛАВР

на тему:

« **Проєктування алгоритму моніторингу мережевої поведінки, спрямованого на оперативне виявлення зловмисної активності в масштабованих IoT-середовищах»**

Муренко Данііл В'ячеславович

(прізвище, ім'я та по батькові здобувача повністю)

Київ 2026 р.

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БУДІВНИЦТВА І АРХІТЕКТУРИ

автоматизації і інформаційних технологій

(факультет)

кібербезпеки та комп'ютерної інженерії

(назва кафедри)

ЗАТВЕРДЖУЮ

Завідувач кафедри

Делембовський М.М.

„___" _____20___року

КВАЛІФІКАЦІЙНА РОБОТА

ЗДОБУВАЧА СТУПЕНЯ ВИЩОЇ ОСВІТИ БАКАЛАВР

«Проєктування алгоритму моніторингу мережевої поведінки, спрямованого на оперативне виявлення зловмисної активності в масштабованих IoT-середовищах»

(назва)

Я як здобувач вищої освіти КНУБА розумію і підтримую політику закладу з академічної доброчесності. Я не надавав(-ла) і не одержував(-ла) недозволену допомогу під час підготовки цієї роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

Здобувач Муренко Данііл В'ячеславович

(прізвище, ім'я та по батькові повністю)

125 «Кібербезпека»

(спеціальність)

Безпека інформаційних та комунікаційних систем

(освітня програма)

Група БІКС-22

Керівник Шабала Є.Є..

(прізвище та ініціали)

доцент кафедри кібербезпеки та комп'ютерної інженерії, кандидат технічних наук

(вчене звання, науковий ступінь)

Рецензент

(прізвище та ініціали)

Ідентичність підтверджу

Київ 2026 рік

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БУДІВНИЦТВА І АРХІТЕКТУРИ

Факультет: автоматизації і інформаційних технологій
Випускова кафедра: кібербезпеки та комп'ютерної інженерії
Ступінь вищої освіти: бакалавр
Спеціальність: 125 «Кібербезпека»
Освітня програма: Безпека інформаційних і комунікаційних систем

ЗАТВЕРДЖУЮ
Завідувач кафедри
Делембовський М.М.
„___" _____20___року

З А В Д А Н Н Я
ДО ВИКОНАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ ЗДОБУВАЧА СТУПЕНЯ ВИЩОЇ ОСВІТИ БАКАЛАВР

Муренко Данііл В'ячеславович
(прізвище, ім'я та по батькові здобувача)

1. Тема роботи «Проєктування алгоритму моніторингу мережевої поведінки, спрямованого на оперативне виявлення зловмисної активності в масштабованих IoT-середовищах»

затверджена наказом ректора КНУБА № 576/52-15/13.2/26 від «14» травня 2026 року

2. Керівник роботи

доцент кафедри кібербезпеки та комп'ютерної інженерії, кандидат технічних наук Шабала Євгенія Євгенівна
(прізвище, ім'я та по батькові, науковий ступінь, вчене звання)

3. Термін подання здобувачем роботи до захисту 30 травня 2026

4. Зміст пояснювальної записки за розділами:

Р. 1. Теоретичні основи та аналіз сучасних підходів до моніторингу iot – середовищ

Р. 2. Постановка задачі та моделювання iot-середовища кіберфізичної системи насосної станції з елементами очищення води

Р. 3. Проєктування алгоритму моніторингу мережевої поведінки та виявлення аномалій аномалій вразливостей

Р. 4. Експериментальне оцінювання ефективності алгоритму

5. Календарний план виконання роботи:

Види робіт та їх зміст	Дата виконання
Розділ 1	04.05.2026
Розділ 2	13.05.2026
Розділ 3	25.05.2026
Остаточне оформлення роботи	01.06.2026
Направлення роботи для перевірки на плагіат	08.06.2026
Попередній захист роботи	08.06.2026
Направлення роботи на рецензування	08.06.2026

6. Дата видачі завдання 10.02.2026

Керівник _____ _____

Здобувач _____ _____

РЕЗЮМЕ (SUMMARY) *до кваліфікаційної випускової роботи здобувача*	ПІБ *Муренко Даніїл В'ячеславович* *Murenko Daniil Vyacheslavovich*		
ЗВО	Київський національний університет будівництва і архітектури		
Тема *(українською та англійською)*	«Проєктування алгоритму моніторингу мережевої поведінки, спрямованого на оперативне виявлення зловмисної активності в масштабованих IoT-середовищах» «Designing a network behavior monitoring algorithm aimed at quickly detecting malicious activity in scalable IoT environments »		
Освітній ступінь	Бакалавр		
Факультет	Автоматизації і інформаціних технологій		
Випускова кафедра	Кібербезпеки та комп'ютерної інженерії		
Спеціальність	125 «Кібербезпека»		
Освітня програма	Безпека інформаційних і комунікаційних систем		
Керівник	Шабала Євгенія Євгенівна		
Обсяг роботи:	*Поснювальна записка, стор.*	*Розділів*	*Презентація, кількість слайдів*
	103	4	12
Розділ 1	Теоретичні основи та аналіз сучасних підходів до моніторингу iot – середовищ		
Розділ 2	Постановка задачі та моделювання iot-середовища кіберфізичної системи насосної станції з елементами очищення води		
Розділ 3	Проєктування алгоритму моніторингу мережевої поведінки та виявлення аномалій аномалій вразливостей		
Розділ 4	Експериментальне оцінювання ефективності алгоритму		

Висновки по роботі	Розроблено алгоритм моніторингу мережевої поведінки та виявлення аномалій у кіберполігоні IoT-насосної станції, що моделює двонасосну гідравлічну систему з 37 телеметричними каналами. Спроєктовано шестикомпонентний конвеєр обробки даних на основі гібридного двошарового детектора (z-score + Isolation Forest). На тестовій вибірці з 184 зразків (FDI, Replay, DoS, фізичні несправності) комбінований конвеєр досяг: Recall = 0,991, Precision = 0,950, F1 = 0,970, AUC = 0,977. Обчислювальна ефективність: 7,3 мс/цикл, пікове споживання пам'яті 5,8 МБ — придатний для вбудованих IoT-систем.
Ключові слова:	безпека IoT, насосна станція, виявлення аномалій, Isolation Forest, z-score, MQTT, гібридний детектор, кіберфізична система, кіберполігон, Python.
Keywords:	IoT security, pump station, anomaly detection, Isolation Forest, z-score, MQTT, hybrid detector, cyber-physical system, cyber range, Python.

Здобувач Муренко Д. В. _____ / _____

Керівник Шабала Є. Є. / _____

АНОТАЦІЯ

У роботі розроблено алгоритм моніторингу мережевої поведінки та виявлення аномалій у кіберполігоні насосної станції на базі IoT, що моделює двонасосну гідравлічну систему з 37 телеметричними каналами протоколу MQTT. Спроєктовано двошаровий гібридний детектор: статистичний шар z-score з подвійним порогом та модель Isolation Forest із каліброваним порогом. Введено фізичну ознаку — залишок за характеристикою насоса εH — для виявлення Replay-атак. Експериментальне оцінювання на 184 зразках показало: Precision = 0,950, Recall = 0,991, F1 = 0,970, AUC = 0,997.

Ключові слова: безпека IoT, насосна станція, виявлення аномалій, Isolation Forest, z-score, MQTT, гібридний детектор, Python.

ABSTRACT

The work develops an anomaly detection algorithm for an IoT pump station cyber range simulating a two-pump hydraulic system with 37 MQTT telemetry channels. A two-layer hybrid detector is designed: a z-score statistical layer with dual-threshold logic and an Isolation Forest model with a calibrated threshold. A physics-based feature — the pump head residual εH — enables Replay attack detection. Experimental evaluation on 184 samples yielded Precision = 0.950, Recall = 0.991, F1 = 0.970, AUC = 0.997.

Keywords: IoT security, pump station, anomaly detection, Isolation Forest, z-score, MQTT, hybrid detector, Python.

ЗМІСТ

ВСТУП

Зі стрімким розвитком інформаційних технологій та зростанням кількості пристроїв Інтернету речей (IoT) у критичних та напівкритичних інфраструктурах, таких як системи водопостачання, енергетики та міського господарства, суттєво підвищується ризик виникнення кіберзагроз. Особливістю IoT-середовищ є велика кількість взаємопов'язаних пристроїв, обмежені обчислювальні ресурси вузлів, а також необхідність забезпечення безперервного функціонування систем у режимі, близькому до реального часу. Це створює додаткові виклики для забезпечення інформаційної безпеки таких систем.

У зв'язку з цим актуальною є задача розробки ефективних методів і алгоритмів моніторингу мережевої поведінки, які здатні оперативно виявляти аномалії та зловмисну активність у масштабованих IoT-системах.

Метою роботи є розробка алгоритму моніторингу мережевої поведінки IoT-пристроїв, який забезпечує оперативне виявлення аномальної та зловмисної активності в умовах масштабованого середовища.

Для досягнення поставленої мети необхідно вирішити такі задачі:
– провести аналіз особливостей IoT-середовищ та їх вразливостей;
– дослідити сучасні підходи до виявлення зловмисної активності, зокрема системи IDS/IPS, а також аномалійні та машинно-навчальні методи;
– сформувати модель мережевої поведінки IoT-пристроїв у нормальному режимі;
– розробити спрощену модель кіберфізичної системи насосної станції з елементами очищення води;
– згенерувати сценарії нормальної роботи, аномалій та атак;
– спроєктувати алгоритм моніторингу мережевої поведінки;
– реалізувати прототип (за можливості) та провести експериментальну оцінку ефективності алгоритму.

Об'єктом дослідження є мережеві взаємодії в IoT-середовищах кіберфізичних систем.

Предметом дослідження є методи та алгоритми виявлення аномальної і зловмисної мережевої активності.

Практичне значення отриманих результатів полягає у можливості їх використання для підвищення рівня безпеки IoT-систем у критичних та промислових середовищах, зокрема у сфері водопостачання.

РОЗДІЛ 1
ТЕОРЕТИЧНІ ОСНОВИ ТА АНАЛІЗ СУЧАСНИХ ПІДХОДІВ ДО МОНІТОРИНГУ IoT – СЕРЕДОВИЩ

1.1 Особливості та архітектура IoT – середовищ

1.1.1 Поняття та структура Інтернету речей

Інтернет речей (Internet of Things, IoT) є сучасною концепцією розвитку інформаційних технологій, що передбачає об'єднання фізичних об'єктів у єдину мережу для збору, передачі та обробки даних без безпосередньої участі людини [1]. До таких об'єктів належать різноманітні пристрої, оснащені датчиками, виконавчими механізмами та засобами зв'язку, що забезпечують їх взаємодію між собою та з зовнішніми інформаційними системами.

Ідея IoT базується на інтеграції фізичного світу з цифровим середовищем, що дозволяє автоматизувати технологічні процеси, підвищити ефективність управління та забезпечити оперативний моніторинг стану систем [2]. Сьогодні IoT широко застосовується у різних сферах, зокрема у промисловості (Industrial IoT), транспорті, енергетиці, медицині, а також у системах водопостачання та водоочищення.

Структура Інтернету речей включає кілька основних компонентів, серед яких можна виділити фізичні пристрої, мережеву інфраструктуру та системи обробки даних. Фізичні пристрої представлені датчиками та виконавчими механізмами, які здійснюють збір інформації про стан навколишнього середовища або технологічного процесу та виконують відповідні дії. Мережевий компонент забезпечує передачу даних між пристроями за допомогою різних протоколів зв'язку. Рівень обробки даних включає сервери або хмарні платформи, де здійснюється зберігання, аналіз та інтерпретація отриманої інформації [3].

Важливою особливістю IoT-систем є їх масштабованість та розподілений характер, що передбачає функціонування великої кількості взаємопов'язаних вузлів. Це створює додаткові вимоги до надійності, безпеки та ефективності

обробки даних, особливо у випадку використання таких систем у критичній інфраструктурі [4].

1.1.2 Рівні архітектури IoT (пристрої, мережа, обробка даних)

Архітектура Інтернету речей зазвичай розглядається як багаторівнева система, що забезпечує ефективну взаємодію між фізичними пристроями, мережевою інфраструктурою та засобами обробки даних. Найбільш поширеною є трирівнева модель, яка включає рівень пристроїв, мережевий рівень та рівень обробки даних [3].

Рівень пристроїв (perception layer) є базовим рівнем IoT-системи і включає фізичні об’єкти, такі як датчики та виконавчі механізми. Датчики здійснюють збір інформації про стан навколишнього середовища або технологічного процесу (наприклад, температура, тиск, рівень води), тоді як виконавчі пристрої виконують керуючі дії відповідно до отриманих команд. Особливістю цього рівня є обмежені обчислювальні ресурси та енергоспоживання пристроїв [5].

Мережевий рівень (network layer) відповідає за передачу даних між пристроями та системами обробки. Він включає різноманітні технології зв’язку, такі як дротові та бездротові мережі, а також протоколи передачі даних. У IoT-середовищах широко використовуються спеціалізовані протоколи, зокрема MQTT, CoAP та HTTP, які оптимізовані для роботи з обмеженими ресурсами та забезпечують ефективну передачу даних [6].

Рівень обробки даних (application або processing layer) призначений для зберігання, аналізу та інтерпретації отриманої інформації. На цьому рівні можуть використовуватися сервери, хмарні платформи або edge-обчислення, що дозволяють здійснювати обробку даних у режимі реального часу. Результати обробки використовуються для прийняття рішень, формування керуючих команд та візуалізації інформації для користувачів [7].

Взаємодія між зазначеними рівнями забезпечує цілісність функціонування IoT-системи, однак одночасно створює додаткові виклики, пов’язані з безпекою, затримками передачі даних та масштабованістю системи. Це обумовлює

необхідність розробки ефективних методів моніторингу та аналізу мережевої поведінки у таких середовищах.

1.1.3 Характеристики мережевої взаємодії IoT-пристроїв

Мережева взаємодія в IoT-середовищах має ряд специфічних характеристик, що відрізняють її від традиційних комп'ютерних мереж. Це обумовлено великою кількістю підключених пристроїв, їх обмеженими ресурсами та необхідністю забезпечення передачі даних у режимі, близькому до реального часу [3].

Однією з ключових особливостей є висока кількість вузлів у мережі, які можуть одночасно передавати дані. У більшості випадків IoT-пристрої здійснюють періодичну передачу телеметричної інформації, що призводить до формування великої кількості невеликих за обсягом пакетів даних [5]. Такий тип трафіку характеризується низькою інтенсивністю для окремого пристрою, але високою сумарною навантаженістю мережі.

Ще однією важливою характеристикою є використання легковагових протоколів передачі даних, спеціально розроблених для IoT-середовищ. Найбільш поширеними серед них є MQTT та CoAP, які забезпечують ефективну передачу даних з мінімальними витратами ресурсів [6], [7]. Протокол MQTT базується на моделі «видавець-підписник» і широко використовується для передачі телеметрії, тоді як CoAP орієнтований на роботу у мережах з обмеженими ресурсами.

Для IoT-мереж також характерні змінні затримки передачі даних та можливі втрати пакетів, що пов'язано з використанням бездротових каналів зв'язку та нестабільністю мережевого середовища Це створює додаткові складнощі для забезпечення надійності передачі інформації та вимагає використання спеціальних механізмів обробки помилок.

Крім того, мережева взаємодія IoT-пристроїв часто має асиметричний характер: значна частина трафіку спрямована від пристроїв до серверів збору даних, тоді як обсяг зворотного трафіку (керуючих команд) є значно меншим. Така особливість враховується при проєктуванні алгоритмів моніторингу мережевої

поведінки, оскільки дозволяє виявляти аномалії, пов’язані зі зміною типових патернів трафіку.

Таким чином, специфіка мережевої взаємодії в IoT-середовищах визначає необхідність застосування спеціалізованих підходів до аналізу трафіку та виявлення аномалій, що є важливим аспектом забезпечення безпеки таких систем.

1.1.4 Обмеження IoT-систем (ресурси, енергоспоживання, масштабованість)

Незважаючи на широкі можливості застосування Інтернету речей, IoT-системи мають ряд суттєвих обмежень, які впливають на їх функціонування, продуктивність та безпеку. Основними з них є обмежені обчислювальні ресурси пристроїв, енергоспоживання та проблеми масштабованості [8].

Більшість IoT-пристроїв характеризуються низькою обчислювальною потужністю, обмеженим обсягом оперативної пам’яті та сховища даних. Це пов’язано з необхідністю зменшення вартості пристроїв та забезпечення їх компактності. У зв’язку з цим складні алгоритми обробки даних або криптографічного захисту не завжди можуть бути реалізовані безпосередньо на пристрої [9]. Це створює додаткові виклики для забезпечення безпеки IoT-середовищ.

Окрім цього, обмежені ресурси пристроїв впливають на можливості їх оновлення та підтримки. Багато IoT-пристроїв не мають достатніх можливостей для регулярного оновлення програмного забезпечення, що призводить до накопичення вразливостей та підвищення ризику кібератак. Відсутність механізмів автоматичного оновлення є однією з причин поширення шкідливого програмного забезпечення у IoT-середовищах [9].

Ще одним важливим обмеженням є енергоспоживання. Значна частина IoT-пристроїв працює від автономних джерел живлення, таких як батареї, що обмежує тривалість їх безперервної роботи. Тому при проєктуванні IoT-систем особлива увага приділяється енергоефективності, зокрема оптимізації передачі даних та використанню енергозберігаючих протоколів [10].

З метою зменшення енергоспоживання часто застосовуються різні режими енергозбереження, такі як перехід пристроїв у сплячий режим або зменшення частоти передачі даних. Проте це може негативно впливати на швидкість реагування системи та точність моніторингу, що є критичним для кіберфізичних систем, зокрема у сфері водопостачання або енергетики [10].

Проблема масштабованості також є критичною для IoT-середовищ. Зі збільшенням кількості підключених пристроїв зростає навантаження на мережеву інфраструктуру та системи обробки даних. Це може призводити до затримок, втрат пакетів та зниження загальної продуктивності системи [11]. Крім того, управління великою кількістю пристроїв ускладнює процеси адміністрування та забезпечення безпеки.

Особливої уваги потребує проблема адресації та ідентифікації великої кількості пристроїв. У масштабних IoT-системах необхідно забезпечити унікальність кожного вузла та можливість ефективного керування ним. Це вимагає використання спеціалізованих протоколів та механізмів, що ускладнює архітектуру системи [11].

Додатковим обмеженням є гетерогенність IoT-середовищ, що полягає у використанні різних типів пристроїв, протоколів та платформ. Це ускладнює інтеграцію компонентів системи та створює проблеми сумісності між ними [12]. У результаті виникає необхідність розробки універсальних підходів до управління та моніторингу таких систем.

Гетерогенність також ускладнює процес забезпечення безпеки, оскільки різні пристрої можуть мати різні механізми захисту або взагалі їх не підтримувати. Це створює слабкі місця в системі, які можуть бути використані зловмисниками для здійснення атак.

Таким чином, обмеження IoT-систем зумовлюють необхідність використання спеціалізованих методів обробки даних, оптимізації ресурсів та розробки ефективних алгоритмів моніторингу, здатних працювати в умовах обмежених ресурсів та великої кількості пристроїв. Врахування цих обмежень є важливим етапом при проєктуванні безпечних та надійних IoT-рішень.

1.2 Безпека IoT та кіберфізичних систем

1.2.1 Особливості безпеки в IoT та CPS

Забезпечення безпеки в IoT-середовищах та кіберфізичних системах (Cyber-Physical Systems, CPS) є складним завданням, що обумовлено специфічними характеристиками таких систем, зокрема їх розподіленістю, гетерогенністю та обмеженими ресурсами пристроїв [8].

Кіберфізичні системи поєднують фізичні процеси з обчислювальними та мережевими компонентами, що забезпечують моніторинг та управління у реальному часі. У таких системах порушення інформаційної безпеки може призвести не лише до втрати даних, але й до фізичних наслідків, таких як пошкодження обладнання або порушення технологічних процесів [14].

Однією з ключових особливостей безпеки IoT є велика кількість потенційних точок входу для зловмисників. Кожен підключений пристрій може виступати як вразливий елемент системи, особливо якщо він має обмежені механізми автентифікації та шифрування [15]. Це значно ускладнює забезпечення комплексного захисту всієї системи.

Важливою проблемою є також обмежені можливості реалізації традиційних засобів захисту. Через низьку обчислювальну потужність IoT-пристроїв складні криптографічні алгоритми та системи виявлення вторгнень не завжди можуть бути застосовані безпосередньо на рівні пристрою [16]. У зв’язку з цим часто використовуються спрощені або розподілені підходи до забезпечення безпеки.

Ще однією особливістю є необхідність забезпечення безпеки в умовах реального часу. У багатьох CPS-системах затримки в обробці або передачі даних можуть призвести до критичних наслідків, тому механізми безпеки повинні бути не тільки ефективними, але й швидкодіючими [14].

Крім того, для IoT-середовищ характерна проблема оновлення програмного забезпечення та управління пристроями. Велика кількість розподілених вузлів ускладнює централізоване адміністрування, що може призводити до використання застарілих версій програмного забезпечення та підвищення рівня вразливості системи [15].

Особливу роль у забезпеченні безпеки відіграє моніторинг мережевої поведінки, який дозволяє виявляти аномалії та потенційні атаки. Оскільки IoT-трафік має характерні патерни, їх аналіз може бути використаний для раннього виявлення загроз [16].

Таким чином, безпека IoT та кіберфізичних систем потребує комплексного підходу, що враховує обмежені ресурси пристроїв, специфіку мережевої взаємодії та критичність фізичних процесів, які контролюються цими системами.

1.2.2 Основні загрози та вразливості IoT-систем

IoT-системи характеризуються підвищеним рівнем вразливості до кіберзагроз, що обумовлено їх архітектурними особливостями, великою кількістю підключених пристроїв та обмеженими можливостями забезпечення захисту. Основні загрози пов’язані як з технічними недоліками пристроїв, так і з помилками у проєктуванні та експлуатації систем [8].

Однією з ключових проблем є використання слабких або стандартних механізмів автентифікації. Багато IoT-пристроїв постачаються з типовими обліковими даними або взагалі не мають належного механізму ідентифікації користувача, що робить їх легкою мішенню для зловмисників [17]. Це дозволяє здійснювати несанкціонований доступ до пристроїв та мережі.

Ще однією важливою вразливістю є відсутність або недостатній рівень шифрування даних. У деяких випадках передача інформації між пристроями здійснюється у відкритому вигляді, що створює можливість перехоплення та модифікації даних під час передачі [8]. Це особливо небезпечно для кіберфізичних систем, де зміна даних може вплинути на фізичні процеси.

Проблема оновлення програмного забезпечення також є критичною. Багато IoT-пристроїв не підтримують автоматичне оновлення або оновлюються рідко, що призводить до накопичення відомих вразливостей [16]. У результаті такі пристрої можуть бути використані як точки входу для атак на всю систему.

Додатковою загрозою є недостатній рівень захисту мережевих протоколів. Легковагові протоколи, такі як MQTT або CoAP, часто не мають вбудованих

механізмів безпеки або використовуються без додаткового шифрування, що підвищує ризик атак типу "людина посередині" (Man-in-the-Middle) [18].

Важливою особливістю IoT-середовищ є їх розподілений характер, що ускладнює централізований контроль та моніторинг безпеки. Велика кількість вузлів створює додаткові труднощі у виявленні компрометації окремих елементів системи [8].

Крім того, фізична доступність IoT-пристроїв також є джерелом загроз. У багатьох випадках пристрої розміщуються у відкритому середовищі, що дозволяє зловмисникам отримати до них фізичний доступ та здійснити модифікацію або підміну обладнання [17].

Таким чином, сукупність технічних, організаційних та архітектурних факторів формує широкий спектр загроз і вразливостей IoT-систем, що вимагає застосування комплексних підходів до забезпечення їх безпеки.

1.2.3 Типові сценарії атак (DDoS, spoofing, botnet)

IoT-середовища є однією з найбільш вразливих категорій сучасних інформаційних систем, що обумовлено великою кількістю підключених пристроїв, їх обмеженими ресурсами та відсутністю належного рівня захисту. Серед найбільш поширених сценаріїв атак виділяють розподілені атаки відмови в обслуговуванні (DDoS), атаки підміни (spoofing), а також формування ботнетів на основі IoT-пристроїв [17], [22].

DDoS-атаки (Distributed Denial of Service) є одними з найбільш небезпечних загроз для IoT-систем. Вони полягають у перевантаженні серверів або мережевих ресурсів великою кількістю запитів, що генеруються з багатьох джерел одночасно. У контексті IoT такі атаки реалізуються через використання великої кількості скомпрометованих пристроїв, які утворюють розподілену мережу. Особливу небезпеку становлять ботнети, здатні генерувати значні обсяги трафіку та виводити з ладу навіть великі інфраструктури [17].

Одним із найвідоміших прикладів є ботнет Mirai, який використовував IoT-пристрої з типовими обліковими даними для організації масштабних DDoS-атак. Дослідження показують, що подібні ботнети можуть швидко масштабуватися та

адаптуватися до змін у мережі, що значно ускладнює їх виявлення та нейтралізацію [18].

Атаки типу spoofing (підміна) полягають у підробці ідентифікаційних параметрів пристрою або мережевого вузла. У IoT-середовищах це може включати підміну IP-адрес, MAC-адрес або ідентифікаторів пристроїв. Завдяки цьому зловмисник може видавати себе за легітимний пристрій, отримувати доступ до системи або впливати на її роботу шляхом передачі неправдивих даних [19]. У кіберфізичних системах це може призводити до серйозних наслідків, включаючи порушення технологічних процесів.

Формування ботнетів є ще одним поширеним сценарієм атак у IoT. Зловмисники використовують вразливості пристроїв для їх зараження та подальшого об’єднання у керовану мережу. Такі ботнети можуть застосовуватися для проведення DDoS-атак, розповсюдження шкідливого програмного забезпечення або збору конфіденційної інформації [20]. Важливою особливістю IoT-ботнетів є їх масовість та складність виявлення через розподілений характер.

Крім зазначених атак, у IoT-середовищах також поширені атаки типу «людина посередині» (Man-in-the-Middle), які дозволяють перехоплювати та змінювати передані дані. Це стає можливим через використання незахищених каналів зв’язку або слабких механізмів автентифікації [22]. Також поширеними є атаки повторного відтворення (replay attacks), при яких зловмисник повторно надсилає перехоплені повідомлення для маніпулювання станом системи.

Особливістю атак на IoT є їх складність виявлення. Через специфіку мережевого трафіку та велику кількість пристроїв зловмисна активність може маскуватися під нормальну поведінку системи. Це обумовлює необхідність використання методів аналізу мережевого трафіку та виявлення аномалій, що є важливим напрямом досліджень у сфері безпеки IoT [13].

Таким чином, різноманітність сценаріїв атак та їх потенційний вплив на фізичні процеси визначають необхідність застосування комплексних підходів до забезпечення безпеки IoT та кіберфізичних систем.

1.2.4 Вимоги до забезпечення безпеки IoT-середовищ

Забезпечення безпеки IoT-середовищ є складним багаторівневим завданням, що потребує врахування специфічних особливостей таких систем, зокрема їх розподіленості, гетерогенності та обмежених ресурсів пристроїв. Ефективна система захисту повинна охоплювати всі рівні архітектури IoT – від фізичних пристроїв до мережевої інфраструктури та рівня обробки даних [8], [15].

Однією з основних вимог є забезпечення автентифікації та авторизації пристроїв і користувачів. Кожен елемент системи повинен мати унікальний ідентифікатор та механізми перевірки доступу, що дозволяє запобігти несанкціонованому підключенню до мережі [22]. Особливо це важливо у середовищах з великою кількістю пристроїв, де контроль доступу є критичним фактором безпеки.

Не менш важливою вимогою є забезпечення конфіденційності та цілісності даних. Передача інформації між IoT-пристроями повинна здійснюватися з використанням криптографічних методів захисту, що запобігають її перехопленню або модифікації. При цьому необхідно враховувати обмежені ресурси пристроїв, що потребує використання легковагових алгоритмів шифрування [9].

Ще однією ключовою вимогою є забезпечення доступності системи. IoT-середовища повинні бути стійкими до атак типу DDoS та інших впливів, що можуть призвести до відмови в обслуговуванні. Для цього застосовуються механізми балансування навантаження, фільтрації трафіку та виявлення аномальної активності [17], [21].

Важливим аспектом є також забезпечення оновлення програмного забезпечення та управління вразливостями. IoT-пристрої повинні підтримувати механізми безпечного оновлення, що дозволяє своєчасно усувати виявлені недоліки та підвищувати рівень захисту системи [15].

Окрему увагу слід приділити моніторингу та виявленню вторгнень. Через специфіку IoT-трафіку традиційні засоби безпеки не завжди є ефективними, тому необхідно використовувати спеціалізовані системи аналізу поведінки та виявлення

аномалій [13]. Це дозволяє своєчасно реагувати на потенційні загрози та мінімізувати їх наслідки.

Крім того, важливою вимогою є забезпечення фізичної безпеки пристроїв. Оскільки багато IoT-пристроїв розміщуються у відкритому середовищі, вони можуть бути піддані фізичному впливу або модифікації, що створює додаткові ризики для всієї системи [22].

Також необхідно враховувати питання масштабованості та керованості системи. Безпекові механізми повинні ефективно працювати навіть при значному збільшенні кількості пристроїв, не створюючи додаткового навантаження на мережу та обчислювальні ресурси [11].

Таким чином, забезпечення безпеки IoT-середовищ вимагає комплексного підходу, що включає реалізацію механізмів автентифікації, шифрування, моніторингу, оновлення програмного забезпечення та захисту від атак. Врахування цих вимог є необхідною умовою для побудови надійних і безпечних IoT-систем.

1.3 Мережевий трафік та потокова обробка даних в IoT

1.3.1 Характеристики мережевого трафіку IoT

Мережевий трафік у IoT-середовищах має ряд специфічних характеристик, що суттєво відрізняють його від трафіку традиційних комп’ютерних мереж. Ці особливості зумовлені великою кількістю підключених пристроїв, їх функціональним призначенням та обмеженими ресурсами, що впливають на способи передачі та обробки даних [5].

Однією з ключових характеристик IoT-трафіку є його нерівномірність та періодичність. Більшість пристроїв передає дані з певною частотою, що залежить від типу сенсора або сценарію використання. Наприклад, датчики температури можуть надсилати інформацію через рівні проміжки часу, тоді як інші пристрої активуються лише при настанні певних подій. Це призводить до формування специфічних патернів трафіку, які можуть бути використані для аналізу поведінки системи [23].

Ще однією важливою особливістю є невеликий обсяг переданих даних. IoT-пристрої, як правило, генерують короткі повідомлення, що містять телеметричну

інформацію або сигнали стану. Водночас загальна кількість таких повідомлень може бути дуже великою, що створює значне навантаження на мережеву інфраструктуру [24]. Це відрізняє IoT-трафік від традиційного, де передаються великі обсяги даних, але з меншою кількістю джерел.

Для IoT також характерна асиметричність трафіку. Основний потік даних спрямований від пристроїв до серверів або хмарних платформ, де здійснюється їх обробка та зберігання. Зворотний трафік, який містить керуючі команди, зазвичай є значно меншим за обсягом [5]. Така особливість повинна враховуватися при аналізі мережевої активності та побудові систем моніторингу.

Важливим аспектом є використання спеціалізованих протоколів передачі даних, таких як MQTT, CoAP та інші легковагові протоколи. Вони оптимізовані для роботи в умовах обмежених ресурсів і забезпечують ефективну передачу даних при мінімальних витратах енергії та пропускної здатності [6], [7]. Це впливає на структуру трафіку та його поведінкові характеристики.

Крім того, IoT-трафік часто передається через бездротові мережі, що зумовлює наявність затримок, втрат пакетів та нестабільності з’єднання. Це ускладнює аналіз трафіку та потребує врахування додаткових факторів при розробці систем виявлення аномалій [24].

Ще однією особливістю є висока гетерогенність джерел трафіку. IoT-середовища можуть включати різні типи пристроїв, які генерують різні за структурою та частотою потоки даних. Це створює складну багатовимірну картину мережевої активності, що ускладнює її аналіз [23].

Важливо також відзначити, що мережевий трафік IoT має відносно стабільні патерни у нормальному режимі роботи. Це дозволяє використовувати методи аналізу поведінки для виявлення відхилень, які можуть свідчити про наявність атак або збоїв у системі [13]. Саме ця властивість лежить в основі сучасних підходів до виявлення аномалій у мережах IoT.

Таким чином, специфічні характеристики IoT-трафіку, зокрема його періодичність, асиметричність, невеликий обсяг повідомлень та гетерогенність

джерел, визначають необхідність застосування спеціалізованих методів аналізу та обробки даних у таких системах.

1.3.2 Протоколи передачі даних (MQTT, CoAP, HTTP)

У IoT-середовищах передача даних між пристроями, серверами та хмарними платформами здійснюється за допомогою спеціалізованих протоколів, які враховують обмежені ресурси пристроїв, нестабільність мережевих з'єднань та вимоги до енергоефективності. Найбільш поширеними серед них є MQTT, CoAP та HTTP, кожен з яких має свої особливості та сфери застосування [6], [7].

Протокол MQTT (Message Queuing Telemetry Transport) є одним із найбільш ефективних рішень для IoT-середовищ. Його робота базується на моделі «видавець–підписник» (publish–subscribe), у якій пристрої обмінюються повідомленнями через центральний брокер. Така архітектура дозволяє зменшити кількість прямих з'єднань між пристроями та забезпечує масштабованість системи. MQTT підтримує три рівні якості обслуговування (QoS), що дозволяє балансувати між надійністю доставки повідомлень та витратами ресурсів [6].

Важливою перевагою MQTT є його здатність ефективно працювати у мережах з низькою пропускною здатністю або нестабільним з'єднанням. Завдяки мінімальному обсягу службових даних цей протокол широко використовується у сенсорних мережах, системах моніторингу та промислових IoT-рішеннях. Дослідження показують, що MQTT забезпечує кращу продуктивність у порівнянні з традиційними протоколами при передачі невеликих повідомлень [25].

Протокол CoAP (Constrained Application Protocol) розроблений спеціально для пристроїв з обмеженими ресурсами та функціонує за принципами REST-архітектури. На відміну від MQTT, він використовує модель «клієнт–сервер» і працює поверх UDP, що дозволяє значно зменшити затримки та накладні витрати. CoAP підтримує механізми підтвердження доставки повідомлень, а також можливість спостереження за ресурсами (observe), що є корисним для систем моніторингу [7], [26].

Однією з ключових переваг CoAP є його інтеграція з HTTP через проксі-сервери, що забезпечує взаємодію між IoT-пристроями та веб-сервісами. Це

дозволяє використовувати існуючу інфраструктуру Інтернету для обробки даних, що надходять від пристроїв [26]. Водночас використання UDP робить CoAP більш чутливим до втрат пакетів, що потребує додаткових механізмів контролю надійності.

HTTP (Hypertext Transfer Protocol) є найбільш поширеним протоколом у традиційних інформаційних системах і також використовується в IoT, переважно для інтеграції з хмарними платформами та веб-додатками. Він базується на моделі «клієнт–сервер» та використовує TCP, що забезпечує надійну передачу даних. Проте значні накладні витрати та більший обсяг службової інформації роблять його менш ефективним для ресурсно-обмежених пристроїв [3].

Незважаючи на це, HTTP має важливі переваги, зокрема універсальність, широку підтримку та сумісність із сучасними веб-технологіями. У практичних IoT-рішеннях він часто використовується для передачі даних на рівні хмарних сервісів, де обмеження ресурсів не є критичними.

З точки зору безпеки, усі зазначені протоколи мають свої вразливості. MQTT та CoAP часто використовуються без належного шифрування або автентифікації, що створює ризики перехоплення даних та несанкціонованого доступу. Для їх захисту застосовуються такі механізми, як TLS/DTLS, проте їх використання може бути обмежене ресурсами пристроїв [22]. HTTP, у свою чергу, потребує використання захищеної версії HTTPS для забезпечення конфіденційності та цілісності даних.

Важливо також відзначити, що вибір протоколу безпосередньо впливає на характеристики мережевого трафіку, такі як затримка, обсяг переданих даних та структура повідомлень. Це має суттєве значення при побудові систем аналізу трафіку та виявлення аномалій, оскільки різні протоколи формують різні поведінкові патерни.

Таким чином, протоколи MQTT, CoAP та HTTP є ключовими компонентами IoT-систем, визначаючи ефективність, масштабованість та безпеку передачі даних. Їх особливості необхідно враховувати при розробці методів аналізу мережевого трафіку та виявлення аномалій у таких середовищах.

1.3.3 Підходи до обробки потокових даних та архітектурні рішення

Потокова обробка даних (stream processing) є ключовою парадигмою для IoT-систем, що передбачає аналіз інформації безпосередньо під час її надходження без попереднього накопичення у сховищі. На відміну від пакетної обробки, потокова модель забезпечує мінімальні затримки та дозволяє реагувати на події в режимі реального часу, що є критично важливим для систем виявлення аномалій у кіберфізичних середовищах.

Основним інструментом потокової обробки є механізм ковзного вікна (sliding window), який дозволяє аналізувати фіксовану послідовність останніх значень у часовому ряді. Цей підхід безпосередньо застосовується в даній роботі: алгоритм моніторингу обробляє ковзне вікно тривалістю 60 секунд із кроком stride = 10 с, що забезпечує безперервний аналіз стану системи при помірному обчислювальному навантаженні.

Важливим напрямом розвитку потокової обробки є edge computing — виконання обчислень безпосередньо поблизу джерела даних, а не у централізованому хмарному середовищі. Такий підхід суттєво зменшує затримки передачі, знижує навантаження на мережу та дозволяє розгортати алгоритми виявлення аномалій у межах локального середовища без залежності від зовнішньої інфраструктури.

Серед основних викликів потокової обробки в IoT виділяють концепцію дрейфу даних (concept drift) — поступову зміну статистичних властивостей потоку з часом через природню еволюцію процесу. Алгоритм виявлення аномалій має враховувати цей ефект, адаптуючи профіль нормальної поведінки до повільних змін, щоб не плутати природній дрейф з аномальною активністю.

1.4 Аномалії в IoT-середовищах та методи їх виявлення

1.4.1 Поняття аномальної поведінки в мережах

У сучасних мережевих системах, зокрема в IoT-середовищах, важливим завданням є виявлення аномальної поведінки, яка може свідчити про наявність кіберзагроз або збоїв у роботі системи. Аномальною поведінкою вважається будь-яке відхилення від нормального функціонування мережі або окремих її елементів [19].

У контексті IoT нормальна поведінка системи характеризується стабільними патернами передачі даних, передбачуваною частотою повідомлень та типовими характеристиками трафіку. Наприклад, сенсори можуть передавати дані через рівні проміжки часу, а обсяг трафіку залишається відносно постійним. Відхилення від таких закономірностей може свідчити про виникнення аномалії [23].

Аномалії можуть виникати з різних причин, серед яких можна виділити як технічні збої, так і навмисні атаки. До технічних причин належать помилки у роботі пристроїв, несправності мережевого обладнання або некоректна конфігурація системи. Водночас кіберзагрози, такі як DDoS-атаки, ботнети або несанкціонований доступ, також проявляються у вигляді змін у поведінці мережевого трафіку [17].

Залежно від характеру відхилень, аномалії можна класифікувати на кілька типів. По-перше, це точкові аномалії, які проявляються як окремі нетипові події, наприклад різке збільшення трафіку. По-друге, колективні аномалії, що виникають у вигляді послідовності подій, які разом формують нетипову поведінку. По-третє, контекстні аномалії, які залежать від умов функціонування системи та можуть бути нормальними в одному контексті, але аномальними в іншому [19].

Особливістю IoT-середовищ є те, що аномалії можуть мати як цифровий, так і фізичний прояв. Наприклад, зміна показників сенсорів може бути результатом як технічної несправності, так і втручання зловмисника. Це ускладнює процес їх ідентифікації та потребує використання комплексних методів аналізу [14].

Ще однією важливою характеристикою є складність відокремлення аномалій від нормальної поведінки у динамічних середовищах. IoT-системи можуть

змінювати свої параметри залежно від умов роботи, що ускладнює визначення чітких меж нормального функціонування. Це створює додаткові труднощі для побудови ефективних систем виявлення [31].

Виявлення аномалій ґрунтується на аналізі мережевого трафіку, поведінки пристроїв та статистичних характеристик даних. Сучасні підходи включають використання методів машинного навчання, які дозволяють автоматично визначати відхилення від нормальної поведінки без необхідності заздалегідь визначених правил [13].

Таким чином, аномальна поведінка в IoT-мережах є важливим індикатором порушень у роботі системи або наявності кіберзагроз. Її своєчасне виявлення дозволяє підвищити рівень безпеки та надійності IoT-середовищ, що є ключовим завданням сучасних досліджень у цій галузі.

1.4.2 Класифікація аномалій (точкові, колективні, контекстні)

Аномалії у мережевому трафіку можуть проявлятися у різних формах залежно від характеру відхилень від нормальної поведінки системи. Для їх ефективного виявлення та аналізу використовується класифікація, що дозволяє систематизувати різні типи аномальних явищ. Найбільш поширеним підходом є поділ аномалій на точкові, колективні та контекстні [19], [31].

Точкові аномалії (point anomalies) є найпростішим типом відхилень і представляють собою окремі значення або події, які суттєво відрізняються від інших даних. У мережевому трафіку IoT це може бути, наприклад, різке зростання кількості пакетів від одного пристрою або поява нетипових запитів. Такі аномалії зазвичай легко виявляються за допомогою статистичних методів або порогових значень [19].

Колективні аномалії (collective anomalies) виникають у випадках, коли окремі елементи даних не є аномальними самі по собі, але їх сукупність утворює нетипову поведінку. У IoT-середовищах це може проявлятися як координована активність великої кількості пристроїв, наприклад під час DDoS-атаки або роботи ботнету. Виявлення таких аномалій є більш складним завданням, оскільки потребує аналізу взаємозв’язків між подіями та часових залежностей [17].

Контекстні аномалії (contextual anomalies) залежать від умов, у яких відбувається аналіз даних. Подія, яка є нормальною в одному контексті, може бути аномальною в іншому. Наприклад, підвищений обсяг трафіку може бути нормальним у період активності системи, але вважатися аномальним у нічний час. У IoT-системах контекст може визначатися часовими параметрами, місцем розташування пристрою або його функціональним призначенням [31].

У практичних IoT-системах ці типи аномалій часто поєднуються, що ускладнює їх ідентифікацію. Наприклад, атака може починатися як точкова аномалія, але з часом переростати у колективну. Крім того, динамічність IoT-середовищ призводить до постійної зміни нормальних патернів поведінки, що ускладнює встановлення чітких меж між нормою та аномалією [23].

Особливу складність становить виявлення контекстних аномалій, оскільки воно потребує врахування додаткових параметрів та використання складніших моделей аналізу даних. У таких випадках ефективними є методи машинного навчання, які здатні адаптуватися до змін у поведінці системи та враховувати багатовимірні залежності [13].

Таким чином, класифікація аномалій на точкові, колективні та контекстні дозволяє систематизувати різні типи відхилень у мережевому трафіку IoT та обрати відповідні методи їх виявлення. Розуміння цих типів є основою для подальшого дослідження методів аналізу та розробки ефективних систем моніторингу безпеки. Класифікацію типів аномалій наведено на рисунку 1.1.

Рисунок 1.1 – Класифікація аномалій в IoT-середовищах: а) точкова, б) колективна, в) контекстна

1.4.3 Методи виявлення аномалій: статистичні, сигнатурні, поведінкові

Для виявлення аномалій у IoT-середовищах застосовують різні підходи, серед яких найбільш поширеними є статистичні, сигнатурні та поведінкові методи. Вибір конкретного методу залежить від особливостей мережевого трафіку, доступності еталонних даних, вимог до швидкодії та допустимого рівня хибних спрацювань. У практичних системах безпеки ці підходи нерідко комбінуються, оскільки кожен із них має власні переваги та обмеження [32].

Статистичні методи ґрунтуються на аналізі числових характеристик мережевого трафіку та пошуку відхилень від заздалегідь визначених або динамічно сформованих норм. До таких характеристик можуть належати середня інтенсивність трафіку, частота пакетів, розподіл розмірів повідомлень, часові інтервали між подіями та інші параметри. Якщо поточні значення суттєво відрізняються від типової моделі, система фіксує можливу аномалію. Перевагою статистичного підходу є його відносна простота, невисокі вимоги до обчислювальних ресурсів і можливість застосування у потоковому режимі. Саме тому статистичні методи часто розглядаються як придатні для ресурсно-обмежених IoT-пристроїв або edge-вузлів. Водночас їхня ефективність істотно залежить від коректності побудови базової моделі нормальної поведінки: якщо система працює

у динамічному середовищі, межі норми можуть змінюватися, що підвищує ризик хибнопозитивних та хибнонегативних рішень [33].

Сигнатурні методи базуються на порівнянні поточних подій або мережевих шаблонів із відомими ознаками атак, які вже збережені у базі правил чи сигнатур. У такому випадку виявлення відбувається тоді, коли спостережувана активність відповідає конкретному шаблону шкідливої поведінки. Перевагою сигнатурного підходу є висока точність щодо вже відомих загроз і, як правило, низький рівень хибних спрацювань для добре описаних атак. Однак основним недоліком є нездатність ефективно виявляти нові, модифіковані або раніше невідомі атаки. Для IoT-середовищ це особливо критично, оскільки пристрої та сценарії їх використання є дуже різноманітними, а атаки можуть швидко змінювати свої ознаки. Через це сигнатурні методи доцільно розглядати як ефективний, але недостатній сам по собі інструмент захисту [32].

Поведінкові методи орієнтовані на формування моделі нормальної роботи мережі, окремого пристрою або групи пристроїв, після чого будь-які істотні відхилення від цієї моделі трактуються як потенційно небезпечні. На відміну від сигнатурних підходів, поведінковий аналіз не потребує повного переліку відомих атак, а тому краще пристосований до виявлення нових або замаскованих загроз. У IoT та CPS-середовищах цей підхід є особливо перспективним, оскільки багато пристроїв працюють за відносно стабільними сценаріями: передають дані з певною періодичністю, використовують обмежений набір протоколів та взаємодіють із невеликою кількістю вузлів. Саме це дає змогу побудувати базову модель штатної поведінки та виявляти нетипові з’єднання, зміну частоти передачі, нові порти, незвичні обсяги трафіку або аномальні часові патерни. Підхід behavioral anomaly detection, описаний у рекомендаціях NIST, передбачає порівняння поточного стану системи з базовим профілем для виявлення відхилень у мережевій активності [34].

Разом із тим поведінкові методи мають і власні обмеження. Їхня ефективність залежить від якості початкового профілю нормальної активності, а також від здатності системи адаптуватися до змін у середовищі. Якщо мережа часто змінює режим роботи, підключаються нові пристрої або змінюються патерни обміну

даними, це може ускладнювати побудову стійкої моделі норми. У таких умовах для покращення результатів застосовують машинне навчання, часовий аналіз, профілювання пристроїв та гібридні схеми, які поєднують статистичні, сигнатурні та поведінкові підходи [35].

Отже, статистичні методи є корисними для швидкого аналізу кількісних відхилень у трафіку, сигнатурні — для точного виявлення вже відомих атак, а поведінкові — для фіксації нових і нетипових сценаріїв мережевої активності. Для IoT-середовищ, де поєднуються обмежені ресурси, велика кількість пристроїв і постійна мінливість загроз, найбільш доцільним є комбіноване використання цих підходів, з особливим акцентом на поведінковий аналіз та адаптивні методи виявлення аномалій [32].

1.4.4 Особливості виявлення аномалій у потокових даних

Виявлення аномалій у потокових даних є складнішим завданням порівняно з аналізом статичних наборів даних, оскільки інформація надходить безперервно та потребує обробки у режимі реального часу. У IoT-середовищах це особливо актуально через велику кількість пристроїв, які генерують дані з високою швидкістю та різною структурою [24].

Однією з ключових особливостей потокових даних є їх безперервність та необмежений обсяг. На відміну від традиційних даних, які можуть бути збережені та проаналізовані повторно, потокові дані обробляються «на льоту», без можливості повернення до попередніх значень. Це вимагає використання алгоритмів, здатних працювати в умовах обмеженої пам’яті та часу [27].

Ще однією важливою характеристикою є необхідність обробки даних з низькою затримкою. У багатьох IoT-застосуваннях, зокрема у кіберфізичних системах, своєчасне виявлення аномалій є критичним для запобігання аваріям або атакам. Це означає, що алгоритми повинні не лише бути точними, але й достатньо швидкими для роботи у реальному часі [14].

Проблемою також є концепція «drift» (зсуву даних), коли статистичні властивості потоку змінюються з часом. У IoT-середовищах це може бути пов’язано зі зміною умов роботи пристроїв, додаванням нових вузлів або зміною

сценаріїв використання. У таких умовах модель, побудована на історичних даних, може швидко втратити актуальність, що знижує ефективність виявлення аномалій [33].

Гетерогенність даних також ускладнює процес аналізу. Потокові дані можуть надходити від різних типів пристроїв, використовувати різні формати та протоколи передачі. Це потребує попередньої обробки, нормалізації та синхронізації даних перед застосуванням алгоритмів виявлення аномалій [23].

Ще однією важливою особливістю є необхідність обмеження використання ресурсів. Багато IoT-пристроїв мають обмежену обчислювальну потужність і енергетичні ресурси, тому алгоритми виявлення аномалій повинні бути оптимізованими та енергоефективними. У зв’язку з цим часто застосовуються легковагові моделі або обробка даних переноситься на edge чи fog-рівень [15].

Важливим аспектом є також баланс між точністю та швидкодією. Більш складні алгоритми можуть забезпечувати вищу точність, але потребують більше ресурсів і часу, що не завжди допустимо у потокових системах. Тому при розробці систем виявлення аномалій необхідно знаходити компроміс між цими параметрами [35].

Крім того, у потокових даних складніше використовувати традиційні методи навчання, які потребують повного набору даних. У таких умовах застосовуються онлайн-алгоритми та інкрементальне навчання, що дозволяють оновлювати модель у процесі надходження нових даних без повного перенавчання [33].

Таким чином, виявлення аномалій у потокових даних характеризується рядом специфічних особливостей, зокрема необхідністю обробки даних у реальному часі, обмеженістю ресурсів, зміною характеристик даних та їх гетерогенністю. Це обумовлює використання спеціалізованих алгоритмів та підходів, здатних ефективно працювати у динамічних IoT-середовищах.

1.4.5 Обмеження існуючих підходів

Незважаючи на значну кількість методів виявлення аномалій у IoT-середовищах, існуючі підходи мають ряд обмежень, що ускладнюють їх ефективне

застосування у реальних умовах. Ці обмеження пов’язані як із специфікою самих IoT-систем, так і з недоліками використовуваних алгоритмів [32], [33].

Однією з основних проблем є залежність методів від якості та повноти навчальних даних. Багато сучасних підходів, зокрема поведінкові та засновані на машинному навчанні, потребують наявності репрезентативного набору даних для побудови моделі нормальної поведінки. У реальних IoT-системах отримання таких даних є складним завданням через різноманітність пристроїв та сценаріїв їх використання [35].

Ще одним суттєвим обмеженням є висока кількість хибнопозитивних та хибнонегативних спрацювань. У динамічних середовищах, де характеристики трафіку можуть змінюватися з часом, навіть незначні відхилення можуть помилково інтерпретуватися як аномалії. Це знижує довіру до системи та ускладнює її практичне використання [33].

Сигнатурні методи, хоча й ефективні для виявлення відомих атак, не здатні реагувати на нові або модифіковані загрози. Це є суттєвим недоліком в умовах постійного розвитку кіберзагроз, особливо у IoT-середовищах, де атаки можуть швидко змінювати свої характеристики [32].

Статистичні методи, у свою чергу, часто мають обмежену точність у складних та багатовимірних системах. Вони можуть не враховувати залежності між різними параметрами трафіку, що знижує їх ефективність при виявленні складних атак або прихованих аномалій [33].

Поведінкові методи, незважаючи на свою гнучкість, потребують значних обчислювальних ресурсів та часу для навчання моделей. Це створює труднощі при їх використанні у реальному часі, особливо на пристроях з обмеженими ресурсами. Крім того, такі методи чутливі до змін у середовищі (concept drift), що може призводити до зниження точності виявлення [36].

Окремою проблемою є масштабованість систем виявлення аномалій. Зі збільшенням кількості IoT-пристроїв зростає обсяг даних, що потребують обробки, що може призводити до перевантаження системи та зниження її продуктивності [24].

Також варто відзначити проблему гетерогенності IoT-середовищ. Різноманітність пристроїв, протоколів та форматів даних ускладнює створення універсальних методів виявлення аномалій, які були б ефективними у всіх випадках [23].

Крім того, забезпечення безпеки самих систем виявлення аномалій є важливим аспектом. Зловмисники можуть намагатися обійти або обманути систему, наприклад шляхом маскування шкідливої активності під нормальну поведінку або поступового зміщення параметрів системи [34].

Таким чином, існуючі підходи до виявлення аномалій мають ряд обмежень, пов’язаних із точністю, масштабованістю, адаптивністю та вимогами до ресурсів. Це обумовлює необхідність подальших досліджень та розробки нових методів, здатних ефективно працювати у складних та динамічних IoT-середовищах.

1.5 Використання методів штучного інтелекту у виявленні аномалій

1.5.1 Основні підходи машинного навчання

Методи машинного навчання є ефективним інструментом для виявлення аномалій у мережевих та IoT-середовищах, оскільки дозволяють автоматично аналізувати великі обсяги даних і виявляти складні закономірності, недоступні для традиційних підходів. Залежно від наявності розмічених даних та постановки задачі, виділяють три основні підходи: навчання з учителем, без учителя та напівкероване навчання [37], [39].

Навчання з учителем (supervised learning) передбачає використання розмічених даних, де кожен приклад належить до певного класу, наприклад «нормальна поведінка» або «аномалія». Модель навчається на таких даних і надалі використовується для класифікації нових спостережень. До поширених алгоритмів належать метод опорних векторів (SVM), дерева рішень, логістична регресія та нейронні мережі. Перевагою цього підходу є висока точність, однак його застосування обмежується необхідністю наявності великої кількості якісно розмічених даних, що у реальних системах часто є проблемою [37].

Навчання без учителя (unsupervised learning) використовується у випадках, коли розмічені дані відсутні. У цьому підході алгоритми самостійно виявляють

структуру даних і визначають відхилення від типових патернів. Найбільш поширеними методами є кластеризація (k-means, DBSCAN) та методи виявлення викидів. Такий підхід є особливо важливим для задач виявлення аномалій, оскільки дозволяє знаходити нові та невідомі загрози без попередньої інформації про них [38].

Напівкероване навчання (semi-supervised learning) поєднує переваги двох попередніх підходів і використовує невелику кількість розмічених даних разом із великою кількістю нерозмічених. У задачах виявлення аномалій часто використовується припущення, що більшість даних є нормальними, а аномалії становлять незначну частину вибірки. Це дозволяє ефективно будувати моделі навіть при обмеженій кількості розмічених прикладів [39].

Окрему роль відіграють методи глибинного навчання (deep learning), які дозволяють автоматично виділяти складні ознаки з даних. До них належать автоенкодери, рекурентні нейронні мережі (RNN) та згорткові нейронні мережі (CNN), що широко застосовуються для аналізу часових рядів і потокових даних. Такі моделі здатні враховувати складні залежності у даних, однак потребують значних обчислювальних ресурсів і великої кількості даних для навчання [37].

Таким чином, різні підходи машинного навчання мають свої переваги та обмеження, а їх вибір залежить від доступності даних, складності задачі та вимог до точності і швидкодії. У практичних системах виявлення аномалій часто застосовуються комбіновані підходи, що дозволяє підвищити ефективність аналізу та адаптивність до змін у середовищі.

1.5.2 Методи та моделі виявлення аномалій

Для виявлення аномалій у мережевому трафіку та IoT-середовищах застосовується широкий спектр методів машинного навчання, які можна умовно поділити на моделі класифікації, кластеризації та спеціалізовані моделі виявлення аномалій. Кожен із цих підходів має свої особливості та сфери застосування [38].

Моделі класифікації використовуються у випадках, коли доступні розмічені дані, що дозволяє віднести спостереження до класів «нормальна поведінка» або «аномалія». До найбільш поширених алгоритмів належать метод опорних векторів

(SVM), дерева рішень, Random Forest та нейронні мережі. Такі моделі забезпечують високу точність виявлення відомих типів атак, однак їх ефективність значною мірою залежить від якості навчальних даних та здатності узагальнювати нові сценарії [37].

Моделі кластеризації застосовуються у задачах без учителя, коли розмічені дані відсутні. Вони дозволяють групувати подібні об’єкти та виявляти відхилення від сформованих кластерів. Найбільш поширеними алгоритмами є k-means, DBSCAN та ієрархічна кластеризація. Аномалії у цьому випадку визначаються як об’єкти, що не належать до жодного кластера або знаходяться на значній відстані від центрів кластерів [38].

Окрему групу становлять спеціалізовані моделі виявлення аномалій, які розроблені безпосередньо для цієї задачі. До них належать методи виявлення викидів (outlier detection), такі як Isolation Forest та Local Outlier Factor (LOF). Ці алгоритми дозволяють ефективно визначати аномальні спостереження навіть у складних багатовимірних даних [38].

У сучасних системах все ширше застосовуються методи глибинного навчання. Зокрема, автоенкодери використовуються для відновлення нормальних даних, і значне відхилення між вхідними та відновленими значеннями може свідчити про аномалію. Рекурентні нейронні мережі (RNN, LSTM) ефективні для аналізу часових рядів і дозволяють враховувати залежності між подіями у часі, що є важливим для потокових IoT-даних [39].

Вибір конкретної моделі залежить від особливостей задачі, доступності даних та вимог до швидкодії. Наприклад, для систем реального часу часто використовуються більш прості та швидкі алгоритми, тоді як для глибокого аналізу можуть застосовуватися складні нейронні мережі. У практичних рішеннях часто комбінуються різні підходи, що дозволяє підвищити точність і стійкість системи до різних типів аномалій.

Таким чином, використання різних методів і моделей машинного навчання дозволяє ефективно виявляти аномалії у складних та динамічних IoT-середовищах, забезпечуючи баланс між точністю, швидкодією та використанням ресурсів.

1.5.3 Обґрунтування вибору підходу

Аналіз існуючих методів виявлення аномалій показав, що жоден із підходів не є універсальним та не забезпечує однаково ефективну роботу у всіх умовах IoT-середовищ. Сигнатурні методи демонструють високу точність для відомих атак, однак не здатні виявляти нові загрози. Статистичні підходи є простими у реалізації, але мають обмежену ефективність у складних багатовимірних системах. Методи машинного навчання є більш гнучкими, проте потребують значних обчислювальних ресурсів і якісних навчальних даних [40], [41].

З урахуванням специфіки IoT-середовищ, таких як велика кількість пристроїв, гетерогенність даних, обмежені ресурси та необхідність обробки інформації у режимі реального часу, найбільш доцільним є використання поведінкового підходу на основі методів машинного навчання. Такий підхід дозволяє формувати модель нормальної роботи системи та виявляти відхилення без необхідності попереднього опису всіх можливих атак [38].

Особливу увагу доцільно приділити методам без учителя або напівкерованого навчання, оскільки у більшості реальних IoT-систем відсутні повністю розмічені набори даних. Це дозволяє виявляти нові та раніше невідомі аномалії, що є критично важливим для забезпечення безпеки [33].

Крім того, враховуючи вимоги до обробки даних у реальному часі, доцільним є використання моделей, які можуть працювати у потоковому режимі та мають прийнятні обчислювальні витрати. У цьому контексті перспективними є легковагові алгоритми машинного навчання або гібридні підходи, що поєднують статистичні методи з поведінковим аналізом [42].

Також важливим є використання розподілених або периферійних обчислень (edge computing), що дозволяє зменшити затримки обробки та навантаження на мережу. Це особливо актуально для масштабних IoT-систем, де централізована обробка може бути неефективною [42].

Таким чином, у даній роботі доцільно застосувати підхід, заснований на поведінковому аналізі мережевого трафіку з використанням методів машинного навчання, орієнтований на обробку потокових даних у реальному часі. Такий підхід

дозволяє врахувати специфіку IoT-середовищ та забезпечити ефективне виявлення як відомих, так і нових аномалій.

Обраний підхід створює основу для подальшої розробки моделі та алгоритму виявлення аномалій, що буде реалізовано у наступному розділі.

РОЗДІЛ 2
ПОСТАНОВКА ЗАДАЧІ ТА МОДЕЛЮВАННЯ IoT-СЕРЕДОВИЩА КІБЕРФІЗИЧНОЇ СИСТЕМИ НАСОСНОЇ СТАНЦІЇ З ЕЛЕМЕНТАМИ ОЧИЩЕННЯ ВОДИ

2.1 Опис кіберфізичної системи насосної станції

У даній роботі розглядається кіберфізична система (КФС) насосної станції з елементами очищення води, яка реалізована у вигляді програмної симуляційної моделі. Система відтворює основні фізичні, енергетичні та технологічні процеси, характерні для реальних водопостачальних комплексів, а також забезпечує генерацію телеметричних даних для подальшого аналізу та виявлення аномалій.

Кіберфізична система поєднує фізичну модель технологічного процесу з інформаційною складовою, яка реалізує збір, передачу та обробку даних. У межах дослідження фізичні процеси (рух рідини, робота насосів, зміна рівнів у резервуарах, контроль якості води) моделюються програмно, а отримані дані передаються через мережеві протоколи, що відповідає концепції IoT-середовища.

Система імітує повний цикл функціонування насосної станції: від отримання електричної енергії, керування електроприводами насосів, транспортування води через трубопровідну систему до її подачі споживачу після проходження етапів контролю якості. Такий підхід дозволяє отримати комплексне уявлення про поведінку системи в нормальних та аномальних режимах.

2.1.1 Склад та характеристики основних компонентів

Кіберфізична система насосної станції складається з ряду взаємопов’язаних компонентів, які моделюють реальні елементи інженерної інфраструктури. Основні компоненти системи можна умовно поділити на енергетичні, гідравлічні, механічні та вимірювальні. Загальну архітектуру кіберполігону насосної станції наведено на рисунку 2.1.

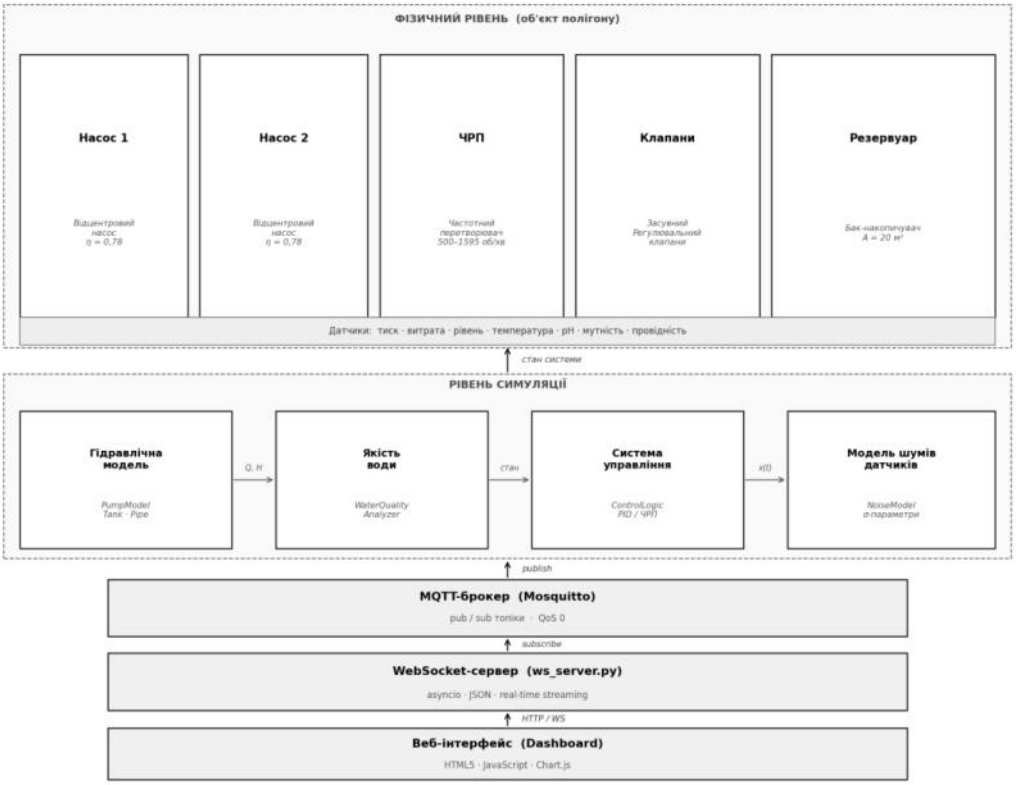

Рисунок 2.1 – Архітектура кіберполігону насосної станції

До енергетичних компонентів належить стабілізатор напруги, який забезпечує підтримання рівня електроживлення у допустимих межах. Номінальне значення напруги становить 400 В, при цьому система допускає відхилення в межах ±15 %. Для керування швидкістю обертання електродвигунів використовуються частотні перетворювачі, які дозволяють змінювати частоту обертання в діапазоні від 500 до 1595 об/хв із заданою швидкістю розгону.

Механічна частина системи представлена електродвигунами та відцентровими насосами. Електродвигуни характеризуються коефіцієнтом корисної дії близько 0,92 та працюють при напрузі 400 В. Насоси забезпечують перекачування рідини та формування напору, який у номінальному режимі досягає значення близько 35 м. Номінальна частота обертання ротора насоса становить 1450 об/хв, що відповідає чотириполюсному асинхронному двигуну в мережі 50 Гц. Коефіцієнт потужності електродвигунів $\cos\varphi = 0{,}85$. Робота насосів визначається їх гідравлічними характеристиками та залежить від частоти обертання ротора.

Гідравлічна підсистема включає приймальний та напірний резервуари, трубопровідну арматуру та регулювальні клапани. Резервуари моделюють накопичення та розподіл води, при цьому рівень рідини змінюється в межах від 0,5 до 6,0 м. Регулювальні клапани дозволяють змінювати витрату рідини шляхом керування ступенем відкриття, тоді як засувки виконують функцію повного перекриття або відкриття потоку.

Окрему роль у системі відіграють вимірювальні та аналітичні компоненти. До них належить аналізатор якості води, який забезпечує контроль таких параметрів, як рівень pH, мутність, електропровідність та температура. Також у системі присутні віртуальні сенсори, що вимірюють тиск, витрату, струм, температуру підшипників, рівень вібрації та інші параметри, необхідні для оцінки стану обладнання.

Усі компоненти функціонують у взаємозв’язку, формуючи єдину систему, де зміна одного параметра впливає на інші. Наприклад, зміна швидкості обертання насоса призводить до зміни витрати, напору, енергоспоживання та температурних характеристик. Така взаємозалежність параметрів є важливою для подальшого аналізу та виявлення аномальних режимів роботи системи.

2.1.2 Опис технологічних процесів системи

Функціонування кіберфізичної системи насосної станції базується на моделюванні сукупності взаємопов’язаних фізичних процесів, які відтворюють реальну роботу водопостачального об’єкта. У межах симуляції всі процеси виконуються дискретно з фіксованим часовим кроком, що дозволяє аналізувати динаміку системи у режимі реального часу.

На першому етапі здійснюється моделювання електроживлення системи. Вхідна напруга мережі формується як випадкова величина з нормальним розподілом навколо номінального значення. Для забезпечення стабільної роботи обладнання використовується стабілізатор напруги, який обмежує відхилення у допустимому діапазоні. Це дозволяє враховувати вплив нестабільності електромережі на роботу всієї системи.

Другим важливим процесом є керування швидкістю обертання насосів за допомогою частотних перетворювачів. Частота обертання змінюється поступово відповідно до заданої швидкості розгону, що відображає реальні інерційні властивості електроприводів. При зниженні напруги живлення передбачено відповідне зменшення швидкості обертання, що впливає на продуктивність насосів.

Наступним етапом є визначення гідравлічного режиму системи. Для цього обчислюється робоча точка, яка визначається перетином характеристик насосів і трубопровідної системи. У моделі враховується залежність напору від витрати, а також взаємодія двох паралельно працюючих насосів. Це дозволяє адекватно відтворювати зміну режимів роботи при різних умовах навантаження, що наведено на рисунку 2.2.

$$H = H_0 \cdot \left(\frac{n}{n_0}\right)^2 - k \cdot Q^2 \quad (2.1)$$

де H — напір насоса, м;
H_0 — напір відсічки при нульовій подачі, м; $H_0 = 35$ м;
n — фактична частота обертання ротора, об/хв;
n_0 — номінальна частота обертання; $n_0 = 1450$ об/хв;
k — коефіцієнт напірної характеристики насоса; $k = 1800$ м·с²/м⁶;
Q — подача (витрата) насоса, м³/с.

$$H_{\text{сис}} = H_{\text{ст}} + R \cdot Q^2 \quad (2.2)$$

де Нсис — потрібний напір трубопровідної системи, м;
Нст — статичний напір (різниця рівнів у резервуарах), м;
R — приведений гідравлічний опір системи; $R = 1800$ м·с²/м⁶;
Q — сумарна витрата двох паралельних насосів, м³/с.

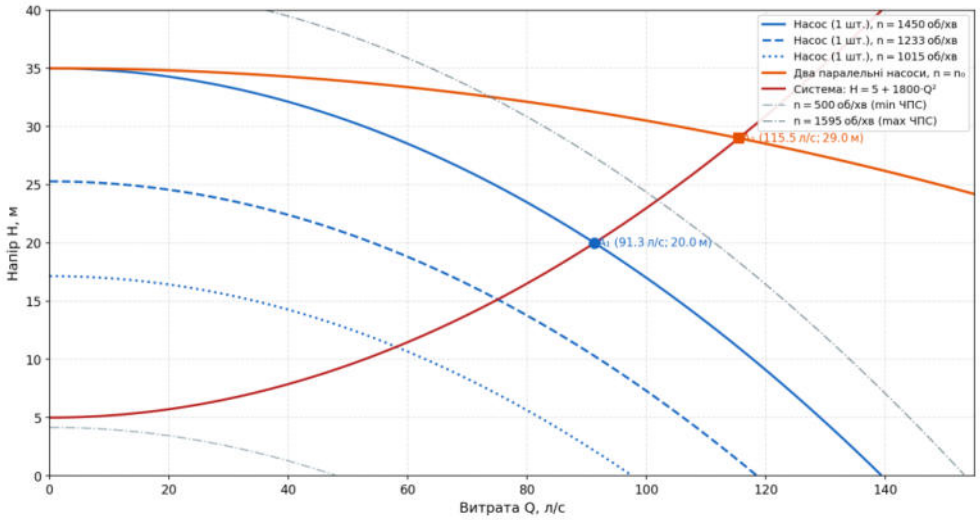

Рисунок 2.2 – Характеристики насоса та трубопровідної системи

Після визначення гідравлічних параметрів розраховуються тиски на вході та виході насосів. Вхідний тиск залежить від рівня рідини у приймальному резервуарі, тоді як вихідний визначається сумою вхідного тиску та напору, створеного насосом. Таким чином формується повна картина розподілу тисків у системі.

Далі виконується оновлення рівнів рідини у резервуарах на основі балансу потоків. Зміна рівня визначається різницею між вхідною та вихідною витратою, що дозволяє моделювати процеси накопичення або споживання води. Це є важливим для аналізу довготривалих режимів роботи системи.

$$L(t + \Delta t) = L(t) + (Q_{вх} - Q_{вих}) \cdot \Delta t / A \quad (2.3)$$

де L — рівень рідини у резервуарі, м;
Qвх — витрата на вході резервуара, м³/с;
Qвих — витрата на виході резервуара, м³/с;
Δt — крок дискретизації симуляції, с;
A — площа поперечного перерізу резервуара, м²; A = 20 м².

Окремо розраховуються енергетичні параметри, зокрема потужність насосів та струм електродвигунів. Ці величини залежать від гідравлічного навантаження та характеристик обладнання і дозволяють оцінювати енергоефективність роботи системи.

$$P_{вал} = \rho \cdot g \cdot Q \cdot H / \eta \quad (2.4)$$

де Рвал — корисна потужність на валу насоса, Вт;
ρ — густина рідини (вода); ρ = 998 кг/м³;

g — прискорення вільного падіння; g = 9,81 м/с²;
Q — витрата насоса, м³/с;
H — напір насоса, м;
η — гідравлічний ККД насоса; η = 0,78.

$$I = \frac{P_{вал}}{(\sqrt{3} \cdot U \cdot cos\varphi \cdot \eta_{ел})} \quad (2.5)$$

де I — лінійний струм електродвигуна, А;
U — лінійна напруга живлення; U = 400 В;
cosφ — коефіцієнт потужності; cosφ = 0,85;
ηел — ККД електродвигуна; ηел = 0,92.

Також у моделі враховуються теплові та механічні процеси. Температура підшипників змінюється відповідно до динамічної моделі, що враховує теплову інерцію, а рівень вібрації залежить від швидкості обертання. Ці параметри є важливими індикаторами технічного стану обладнання.

Завершальним етапом є моделювання якості води. У системі відтворюються такі показники, як pH, мутність, електропровідність та температура. Зміна цих параметрів описується як процес із поступовим поверненням до середніх значень, що дозволяє імітувати природні коливання та вплив зовнішніх факторів.

Усі описані процеси виконуються послідовно в межах кожного кроку симуляції та формують повний набір параметрів, що характеризують стан системи. Такий підхід забезпечує комплексне відображення як фізичних, так і інформаційних аспектів функціонування кіберфізичної системи та створює основу для подальшого аналізу і виявлення аномалій.

2.2 Архітектура IoT-середовища системи

Архітектура розробленої кіберфізичної системи насосної станції відповідає класичній багаторівневій моделі Інтернету речей, що передбачає розподіл функцій між рівнями збору даних, їх передачі та обробки. Такий підхід дозволяє забезпечити масштабованість, гнучкість та можливість інтеграції з іншими інформаційними системами.

2.2.1 Рівень сенсорів та вимірювань

Рівень сенсорів у даній системі реалізований у вигляді програмної моделі, яка імітує роботу реальних вимірювальних пристроїв. На цьому рівні формується

первинна інформація про стан фізичних процесів, зокрема значення тиску, витрати, рівня рідини, температури, струму, вібрації та параметрів якості води.

Особливістю реалізації є використання моделі шуму вимірювань, що дозволяє наблизити симульовані дані до реальних умов експлуатації. Кожне значення параметра формується з урахуванням випадкової похибки, яка описується нормальним розподілом:

$$\varepsilon \sim N(0, \sigma^2) \tag{2.6}$$

Таким чином, виміряне значення визначається як сума істинного значення параметра та випадкового шуму. Рівень шуму залежить від типу вимірюваної величини та задається окремо для кожного параметра.

Такий підхід дозволяє враховувати неточності вимірювань, характерні для реальних сенсорів, а також створює більш реалістичні умови для подальшого аналізу та виявлення аномалій.

Сформовані вимірювання представляють собою набір параметрів, що характеризують поточний стан системи, і використовуються як вхідні дані для наступного рівня — мережевої передачі. При цьому всі сенсорні дані генеруються синхронно з фіксованим часовим кроком, що забезпечує узгодженість часових рядів.

Важливою особливістю є те, що сенсорний рівень інтегрований безпосередньо у симуляційний двигун, що дозволяє уникнути затримок між фізичним моделюванням і формуванням даних. Це забезпечує цілісність даних та коректне відображення взаємозв’язків між параметрами системи.

Таким чином, рівень сенсорів формує повний набір вимірювань, необхідних для моніторингу стану кіберфізичної системи, і створює основу для подальшої передачі та обробки даних у рамках IoT-архітектури.

2.2.2 Мережевий рівень та протоколи передачі даних

Мережевий рівень кіберфізичної системи забезпечує передачу даних між компонентами та реалізує взаємодію між фізичною моделлю та прикладними сервісами. У досліджуваній системі для обміну даними використовуються сучасні мережеві протоколи, характерні для IoT-середовищ, зокрема MQTT та WebSocket.

Основним протоколом передачі телеметричних даних є MQTT (Message Queuing Telemetry Transport), який працює поверх TCP/IP та орієнтований на ефективну передачу невеликих повідомлень у системах з обмеженими ресурсами. У системі використовується брокер повідомлень, який виконує функцію центрального вузла маршрутизації даних між джерелами та споживачами інформації.

Передача даних організована за моделлю publish/subscribe, де симуляційний модуль виступає у ролі видавця (publisher), а клієнтські додатки — у ролі підписників (subscriber). Кожне вимірювання публікується у відповідний тематичний канал (topic), що дозволяє гнучко організовувати доступ до окремих груп даних. Частота публікації становить один раз на секунду, що відповідає часовому кроку симуляції.

У системі використовується базовий рівень якості обслуговування MQTT (QoS = 0), що означає передачу повідомлень без підтвердження доставки. Такий підхід мінімізує затримки та навантаження на мережу, однак допускає можливість втрати окремих пакетів, що відповідає реальним умовам роботи IoT-систем.

Додатково для забезпечення взаємодії з користувацьким інтерфейсом використовується протокол WebSocket, який забезпечує двосторонній обмін даними у режимі реального часу. WebSocket-з’єднання використовується для передачі агрегованих даних від симуляційного модуля до SCADA-інтерфейсу, що дозволяє оперативно відображати стан системи.

Особливістю реалізації є використання MQTT поверх WebSocket, що забезпечує можливість підключення веб-клієнтів безпосередньо до брокера повідомлень. Це спрощує інтеграцію з фронтенд-додатками та дозволяє використовувати стандартні браузерні засоби для моніторингу.

Таким чином, мережевий рівень системи реалізує гнучку та масштабовану модель обміну даними, яка поєднує легковаговий протокол MQTT для телеметрії та WebSocket для інтерактивної взаємодії. Це створює основу для подальшого аналізу мережевого трафіку та виявлення аномалій на рівні передачі даних.

2.2.3 Рівень обробки та візуалізації даних

Рівень обробки та візуалізації даних забезпечує прийом, обробку та представлення інформації, що надходить із сенсорного та мережевого рівнів. У досліджуваній системі цей рівень реалізований у вигляді симуляційного ядра та клієнтського інтерфейсу моніторингу.

Основним елементом обробки даних є симуляційний модуль, який виконує розрахунок фізичних параметрів системи та формує набір вимірювань на кожному кроці часу. Саме на цьому рівні відбувається узгодження даних, їх агрегація та підготовка до передачі через мережеві протоколи. Обробка виконується у синхронному режимі з фіксованим часовим кроком, що забезпечує цілісність часових рядів.

Для візуалізації стану системи використовується SCADA-подібний інтерфейс, реалізований у вигляді веб-додатку. Інтерфейс отримує дані через MQTT-брокер (із використанням WebSocket) або напряму через WebSocket-з’єднання із сервером, що забезпечує передачу агрегованих даних у реальному часі.

У візуальній частині системи відображаються ключові параметри роботи насосної станції, зокрема рівні у резервуарах, витрати, тиски, електричні параметри та показники якості води. Дані подаються у вигляді графіків часових рядів та інтерактивних елементів, що дозволяє оператору оцінювати динаміку процесів.

Для аналізу динаміки параметрів використовується механізм рухомого вікна, який зберігає обмежену кількість останніх значень кожного параметра. Це дозволяє відображати актуальні тенденції без перевантаження системи зберігання даних.

Важливою особливістю є робота системи у режимі реального часу, що забезпечується асинхронною передачею даних та ефективною обробкою повідомлень. Це дозволяє мінімізувати затримки між генерацією даних та їх відображенням.

Таким чином, рівень обробки та візуалізації забезпечує інтеграцію фізичної моделі з користувацьким інтерфейсом, створюючи зручні засоби для моніторингу стану системи та подальшого аналізу її поведінки.

2.3 Мережева модель та потоки даних

Мережева модель досліджуваної кіберфізичної системи побудована за принципом централізованої взаємодії з використанням брокера повідомлень. Усі компоненти системи обмінюються даними через єдиний вузол, що забезпечує маршрутизацію, масштабованість та спрощує моніторинг мережевого трафіку.

Топологія мережі є зірковою (broker-centric), де центральним елементом виступає MQTT-брокер. Симуляційний модуль генерує телеметричні дані та передає їх до брокера, тоді як клієнтські додатки (зокрема SCADA-інтерфейс) підписуються на відповідні топіки та отримують необхідну інформацію. Додатково реалізовано прямий канал передачі даних через WebSocket для забезпечення швидкої візуалізації у режимі реального часу. Топологія мережі представлена на рис. 2.x.

У системі виділяються такі основні вузли: симуляційний модуль (джерело даних), MQTT-брокер (маршрутизація повідомлень), WebSocket-сервер (передача агрегованих даних) та клієнтський SCADA-інтерфейс (візуалізація). Така структура дозволяє чітко розділити функції генерації, передачі та споживання даних.

Передача інформації у системі здійснюється у вигляді потоків даних. Основним є висхідний потік телеметрії, у межах якого симуляційний модуль публікує значення параметрів у вигляді окремих повідомлень MQTT з частотою один раз на секунду. Паралельно формується потік даних для візуалізації, який передається через WebSocket у вигляді агрегованого JSON-об’єкта.

Типи мережевого трафіку включають телеметричні дані (параметри насосів, резервуарів, енергетичні показники), стан арматури, показники якості води та службові повідомлення протоколу MQTT. Для кожного типу характерна фіксована структура та періодичність передачі, що дозволяє використовувати їх як основу для подальшого аналізу та виявлення аномалій.

Таким чином, мережева модель системи забезпечує стабільну передачу даних, чітку структурованість потоків інформації та створює необхідні умови для аналізу мережевої поведінки у IoT-середовищі.

2.4.1 Структура та формат даних

У процесі роботи кіберфізичної системи формується потік телеметричних даних, що відображають стан основних компонентів у кожен момент часу. Дані генеруються з фіксованим інтервалом та представляють собою набір параметрів, які характеризують фізичні, енергетичні та технологічні процеси системи.

Кожне вимірювання відповідає окремому параметру, зокрема напрузі живлення, швидкості обертання насосів, витраті рідини, тиску, рівню у резервуарах, струму електродвигунів, температурі підшипників, рівню вібрації, а також показникам якості води (pH, мутність, електропровідність, температура). Сукупність цих параметрів формує вектор стану системи у момент часу t.

Передача даних у мережі здійснюється у форматі JSON, де кожне повідомлення містить значення окремого параметра у вигляді пари «топік – значення». Такий підхід відповідає моделі publish/subscribe протоколу MQTT та дозволяє гнучко організувати доступ до даних.

Формально, стан системи можна представити у вигляді вектора:

$$x_t = (x_1, x_2, \dots, x_n) \tag{2.7}$$

де n — кількість вимірюваних параметрів. Це представлення є основою для подальшої обробки та аналізу даних.

Таким чином, структура даних забезпечує повне та уніфіковане представлення стану системи, придатне для використання у задачах моніторингу та виявлення аномалій.

2.4.2 Формування ознак для аналізу

Для ефективного виявлення аномалій первинні телеметричні дані перетворюються у набір ознак, які більш повно відображають поведінку системи. Формування ознак є важливим етапом, оскільки саме від їх інформативності залежить точність подальшого аналізу.

До базових ознак належать безпосередньо вимірювані параметри системи, такі як витрата, тиск, температура, струм та рівень рідини. Вони дозволяють оцінити поточний стан обладнання та виявити явні відхилення від нормального режиму роботи.

Додатково використовуються статистичні ознаки, що обчислюються на основі часових рядів. До них належать середнє значення, дисперсія, а також швидкість зміни параметрів. Такі характеристики дозволяють враховувати динаміку процесів та виявляти поступові або приховані аномалії.

Окрему групу становлять мережеві ознаки, які описують поведінку трафіку. До них відносяться частота передачі повідомлень, інтервали між пакетами та розмір повідомлень. Ці параметри є важливими для виявлення атак на мережевому рівні, зокрема перевантаження або підміни даних.

Найбільш інформативними є кросс-кореляційні ознаки, що враховують взаємозв’язки між параметрами. Наприклад, залежність між швидкістю обертання насоса та напором або між потужністю і струмом. Порушення таких залежностей може свідчити про наявність аномалій навіть у випадках, коли окремі параметри залишаються у допустимих межах.

У загальному вигляді вектор ознак формується як функція від поточного та попередніх значень параметрів:

$$\varphi(x_t) = f(x_t, x_{t-1}, \dots, x_{t-k}) \tag{2.8}$$

де k — розмір часового вікна.

Таким чином, формування ознак дозволяє перейти від сирих даних до інформативного представлення, яке враховує як значення параметрів, так і їх взаємозв’язки та динаміку зміни. Динаміку ключових ознак у різних режимах роботи ілюструє рисунок 2.3.

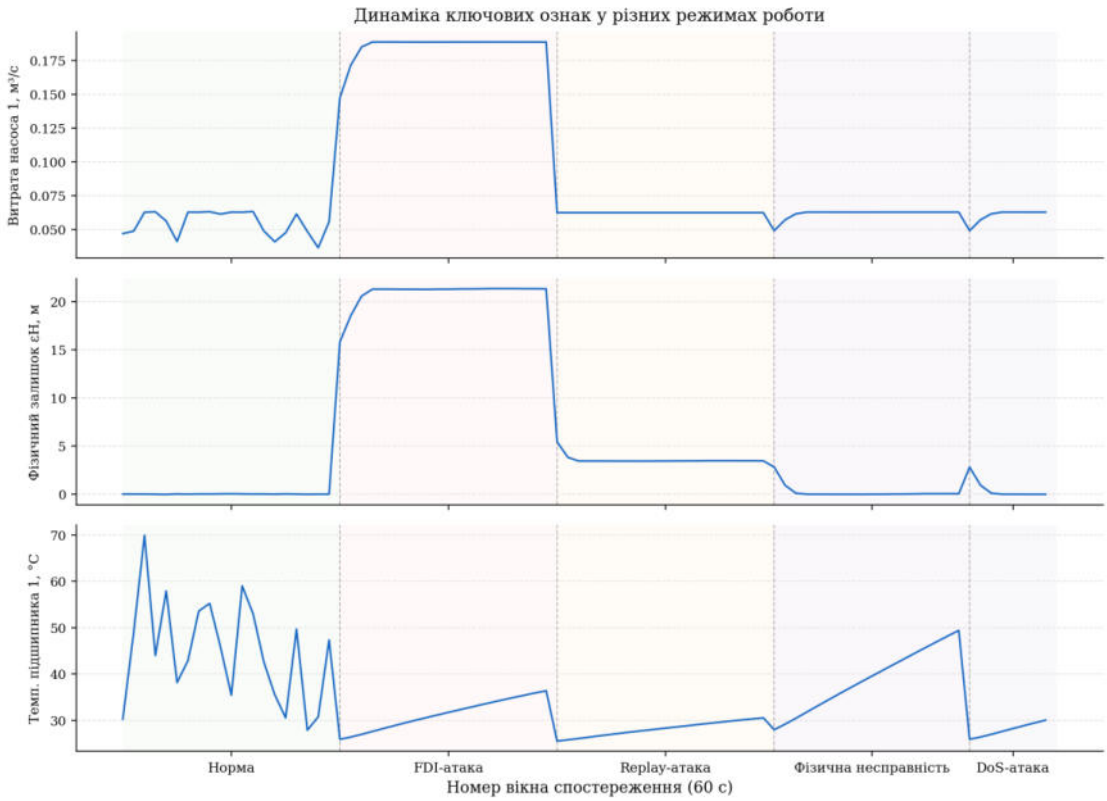

Рисунок 2.3 – Динаміка ключових ознак телеметрії у різних режимах роботи насосної станції

2.5.1 Вхідні дані та вектор ознак

У рамках дослідження кіберфізичної системи насосної станції вхідні дані представлені у вигляді багатовимірного часового ряду, що формується на основі телеметричних вимірювань. Кожен момент часу t характеризується вектором стану системи:

$$x_t = (x_1, x_2, \dots, x_n) \in \mathbb{R}^n \quad (2.9)$$

де n — кількість параметрів, що вимірюються у системі.

До складу вектора x_t входять параметри різної природи: електричні (напруга, струм), механічні (швидкість обертання, вібрація), гідравлічні (витрата, тиск, рівень), а також показники якості води. Таким чином, вектор x_t є узагальненим описом стану системи у момент часу t.

Оскільки система функціонує у динамічному режимі, для аналізу враховується не лише поточне значення параметрів, але й їх зміна у часі. З цією метою формується розширений вектор ознак, що включає значення за певне часове вікно:

Відповідно до виразу (2.8), вектор ознак $\varphi(x_t)$ охоплює значення параметрів системи за попередні k кроків у часі.

Вектор ознак $\varphi(x_t)$ включає:

- первинні параметри (безпосередні вимірювання);
- статистичні характеристики (середнє, дисперсія, тренд);
- похідні показники (швидкість зміни параметрів);
- кросс-кореляційні ознаки, що відображають взаємозв'язки між параметрами;
- мережеві характеристики (частота повідомлень, інтервали між пакетами).

У загальному випадку вектор ознак має більшу розмірність:

$$\varphi(x_t) \in \mathbb{R}^d,\ d \geq n \tag{2.10}$$

Таким чином, формується інформативне представлення даних, яке враховує як поточний стан системи, так і його динаміку, що є необхідним для ефективного виявлення аномалій.

2.5.2 Модель виявлення аномалій

Задача виявлення аномалій у досліджуваній системі формулюється як задача класифікації або оцінки відхилення від нормальної поведінки. Основною метою є визначення моментів часу, у яких стан системи відрізняється від очікуваного.

Формально, вводиться функція виявлення аномалій:

$$f : \mathbb{R}^d \rightarrow \{0, 1\} \tag{2.11}$$

де $f(\varphi(x_t)) = 0$ — нормальний стан, $f(\varphi(x_t)) = 1$ — аномалія.

У випадку використання ненаглядових методів (unsupervised learning) вводиться поняття оцінки аномальності:

$$s(x_t) = d\left(\varphi(x_t), \Omega_{\text{норм}}\right) \tag{2.12}$$

де Ωнорм — множина нормальних станів системи, а $d(\cdot)$ — функція відстані або міри відхилення.

Рішення про наявність аномалії приймається на основі порогового значення:

$$f(x_t) = 1 \quad \Leftrightarrow \quad s(x_t) > \theta \tag{2.13}$$

де θ — поріг, що визначається експериментально або на основі статистичних характеристик даних.

У контексті даної роботи передбачається використання моделей, здатних працювати з багатовимірними часовими рядами та враховувати складні залежності між параметрами. Це можуть бути як класичні методи (наприклад, кластеризація або ізоляційні алгоритми), так і більш складні підходи, зокрема нейронні мережі.

Загальна схема обробки даних має вигляд:

$$x_t \rightarrow \varphi(x_t) \rightarrow f\big(\varphi(x_t)\big) \rightarrow \{\text{норм.}, \text{аномалія}\} \tag{2.14}$$

Таким чином, задача виявлення аномалій зводиться до побудови відображення, яке дозволяє ефективно відокремлювати нормальні стани системи від аномальних на основі сформованого вектора ознак.

2.6 Обмеження системи

Розроблена кіберфізична система насосної станції є симуляційною моделлю, що відтворює основні аспекти реального IoT-середовища. Водночас така модель має ряд обмежень, які необхідно враховувати при аналізі результатів та інтерпретації отриманих даних.

Одним із ключових обмежень є обчислювальні особливості системи. Симуляція фізичних процесів виконується у програмному середовищі з дискретним часовим кроком, що спрощує розрахунки, але не дозволяє повністю відтворити безперервну природу реальних процесів. Крім того, окремі алгоритми, зокрема обчислення робочої точки насосів, реалізуються ітераційними методами, що може впливати на продуктивність системи при зменшенні кроку моделювання або збільшенні кількості компонентів.

Іншим важливим обмеженням є спрощення фізичних моделей. У системі використовуються узагальнені залежності між параметрами (наприклад, залежність напору від швидкості обертання або витрати), які не враховують усіх можливих факторів, таких як зношення обладнання, турбулентні ефекти або

складна гідравліка трубопроводів. Це означає, що модель відображає типову поведінку системи, але може не враховувати окремі специфічні режими.

З точки зору мережевої взаємодії система також має ряд обмежень. Передача даних здійснюється без використання механізмів автентифікації та шифрування, що спрощує реалізацію, але не відповідає вимогам безпеки реальних промислових систем. Крім того, використання MQTT з рівнем якості обслуговування QoS = 0 не гарантує доставку повідомлень, що може призводити до втрати частини даних при навантаженні на мережу.

Обмеженням є також централізований характер архітектури. Використання одного брокера повідомлень спрощує управління, але створює потенційну точку відмови. У разі перевантаження або відмови брокера порушується робота всієї системи.

З боку обробки даних варто відзначити відсутність довготривалого зберігання інформації. Дані використовуються переважно у режимі реального часу та зберігаються у вигляді обмеженого часового вікна. Це обмежує можливості глибокого історичного аналізу та побудови довгострокових прогнозів.

Ще одним важливим обмеженням є відсутність повноцінного контуру керування. У системі реалізовано лише передачу телеметричних даних, тоді як механізми зворотного впливу (керуючі сигнали) не використовуються. Це означає, що модель не відображає повністю всі аспекти реальної кіберфізичної системи, зокрема процеси автоматичного регулювання.

З точки зору задачі виявлення аномалій слід враховувати обмеження, пов’язані з якістю даних. Оскільки дані генеруються штучно, вони можуть не повністю відображати складність реальних умов, зокрема наявність шумів, пропусків або неочікуваних комбінацій параметрів. Це може впливати на узагальнювальну здатність моделей, побудованих на таких даних.

Таким чином, розроблена система є ефективним інструментом для дослідження та моделювання IoT-середовищ, однак її результати слід розглядати з урахуванням зазначених обмежень. Врахування цих факторів є необхідним для коректної інтерпретації результатів та подальшого розвитку системи.

РОЗДІЛ 3. ПРОЄКТУВАННЯ АЛГОРИТМУ МОНІТОРИНГУ МЕРЕЖЕВОЇ ПОВЕДІНКИ ТА ВИЯВЛЕННЯ АНОМАЛІЙ

3.1 Загальна архітектура алгоритму моніторингу

3.1.1 Концепція та принципи побудови

Промислові системи керування (Industrial Control Systems, ICS), до яких належать насосні станції, є критичними об'єктами інфраструктури, кіберзахист яких принципово відрізняється від захисту корпоративних інформаційних систем. Традиційні засоби мережевої безпеки — міжмережеві екрани, системи виявлення вторгнень на основі сигнатур (Signature-based IDS), антивірусне програмне забезпечення — проєктувалися для захисту IT-середовищ і мають суттєві обмеження при застосуванні в OT-середовищах (Operational Technology).

По-перше, ICS-середовища характеризуються жорсткими вимогами до доступності та детермінованості: будь-яка активна реакція системи захисту (блокування трафіку, ізоляція вузла) може призвести до аварійної зупинки технологічного процесу, що є неприйнятним для безперервного виробництва. По-друге, промислові протоколи (Modbus, DNP3, MQTT) та вбудовані контролери (PLC, RTU) часто не підтримують сучасні криптографічні механізми автентифікації, що робить периметровий захист недостатнім. По-третє, і це найважливіше, значна частина кібератак на ICS взагалі не виявляється мережевими засобами, оскільки використовує легітимні протоколи та автентифіковані з'єднання для маніпуляції технологічним процесом зсередини — як це продемонструвала атака Stuxnet у 2010 році та численні подальші інциденти на об'єктах енергетики та водопостачання.

Це зумовлює необхідність поглиблення моніторингу до рівня технологічного процесу: замість (або на додаток до) аналізу мережевих пакетів — аналіз поведінки фізичної системи за показниками датчиків. Відхилення фізичних параметрів від очікуваних значень є надійним індикатором кібератаки або несправності незалежно від того, яким мережевим шляхом було доставлено шкідливий вплив.

Концепція поведінкового виявлення аномалій

В основу розробленого алгоритму покладено концепцію поведінкового виявлення аномалій (behavioral anomaly detection). Її суть полягає в тому, що система моніторингу не шукає відомі шаблони атак, а натомість будує модель нормальної поведінки об'єкта і виявляє статистично значущі відхилення від цієї моделі.

Порівняно з двома альтернативними підходами, поведінковий підхід має такі переваги:

Сигнатурний підхід потребує наявності бази знань про відомі атаки та їх ознаки. Для ICS-середовищ такі бази знань є неповними: більшість атак є цільовими (targeted) і розробляються спеціально під конкретний об'єкт, тому їх сигнатури заздалегідь невідомі. Крім того, атакувальники можуть модифікувати сигнатури, зберігаючи при цьому функціональний ефект атаки.

Специфікаційний підхід (specification-based detection) потребує формальної специфікації допустимої поведінки системи у вигляді правил або автоматів станів. Для складних технологічних процесів із багатовимірним простором станів розробка повної специфікації є трудомістким і схильним до помилок процесом. Будь-яка неврахована легітимна поведінка стає джерелом хибних спрацювань.

Поведінковий підхід позбавлений цих недоліків: він автоматично навчається нормальній поведінці системи, не потребує знань про атаки і адаптується до змін режиму роботи. Основним обмеженням є чутливість до якості навчальних даних — якщо в навчальну вибірку потрапили аномальні приклади, модель норми буде спотворена. Це обмеження долається відповідними вимогами до процедури збору даних для навчання. Порівняння підходів до виявлення кіберзагроз наведено на рисунку 3.1.

Рисунок 3.1 – Порівняння підходів до виявлення кіберзагроз в промислових системах керування

Ключові принципи побудови алгоритму

Принцип ненаглядового навчання. Модель виявлення аномалій навчається виключно на даних нормального функціонування системи без використання помічених прикладів атак. Це відповідає реальним умовам промислового середовища, де атаки є рідкісними подіями, а збір репрезентативного навчального набору з позначеними аномаліями практично неможливий з двох причин: (1) оператор не може навмисно піддавати реальну систему атакам лише для збору навчальних даних; (2) навіть якщо такі дані є, вони охоплюють лише вже відомі типи атак, залишаючи невидимими нові вектори загроз. Ненаглядове навчання забезпечує здатність виявляти будь-які аномалії, у тому числі ті, що не спостерігалися при навчанні.

Принцип багатошарового виявлення. Алгоритм реалізує два незалежні шари аналізу з різними математичними основами: статистичний (на основі z-оцінок відносно динамічного профілю норми) та машинного навчання (на основі Isolation Forest). Ці шари є взаємодоповнюючими: статистичний шар ефективний для різких, явних відхилень окремих ознак і не потребує попереднього навчання; шар машинного навчання ефективний для виявлення складних мультиваріативних аномалій, де жодна окрема ознака не є очевидно аномальною, проте їх сукупне

поєднання є статистично малоймовірним для нормального режиму. Комбінація шарів знижує ймовірність як хибних спрацювань, так і пропущених атак — типового компромісу між precision та recall в задачах класифікації.

Принцип ковзного вікна. Аналіз виконується над ковзним вікном часового ряду фіксованої довжини (60 секунд), що забезпечує неперервність моніторингу та дозволяє відстежувати динамічні зміни процесу, а не лише миттєві значення показників. Розмір вікна вибрано з таких міркувань: воно має бути достатньо великим, щоб усереднити шум вимірювань і забезпечити статистично значущі оцінки, але достатньо малим, щоб гарантувати своєчасне виявлення атаки. При частоті вимірювань 1 Гц вікно 60 с дає по 60 відліків для кожного каналу, що достатньо для оцінки середнього та стандартного відхилення з похибкою менше 15% при типовому відношенні сигнал/шум промислових датчиків.

Принцип фізичної обґрунтованості ознак. Поряд зі статистичними характеристиками сигналів датчиків, у набір ознак включено фізичні залишки — відхилення між виміряними параметрами та значеннями, передбаченими математичною моделлю технологічного процесу. Зокрема, використовується залишок за характеристикою напір–витрата насоса: різниця між виміряним і теоретично очікуваним напором при поточній витраті та частоті обертання. Цей показник залишається близьким до нуля за будь-якого легітимного режиму роботи і різко відхиляється при підробці показань датчиків тиску або витрати. Включення фізичних залишків реалізує підхід physics-informed anomaly detection, який дозволяє виявляти атаки типу False Data Injection навіть тоді, коли підроблені значення лежать у межах фізично правдоподібного діапазону для кожного датчика окремо, але суперечать фізичним законам у сукупності.

Принцип гістерезису. Для зниження реакції на короткочасні статистичні викиди - шум вимірювань, перехідні процеси при зміні режиму роботи, поодинокі збої комунікаційного каналу - алгоритм генерує сповіщення лише після N_{hys} послідовних аномальних вікон, де $N_{hys} = 3$ за замовчуванням. Це реалізує механізм підтвердження аномалії в часі, аналогічний гістерезису в системах автоматичного регулювання. Математично це еквівалентно перевірці

гіпотези про стаціонарність аномального стану: ймовірність N_{hys} незалежних хибних спрацювань поспіль рівна $p^{N_{hys}}$ де p ймовірність одиночного хибного спрацювання; при $p = 0.05$ і $N_{hys} = 3$ маємо $0.05^3 = 1.25 \times 10^{-4}$. Ціна цього покращення – збільшення затримки виявлення на $(N_{hys} - 1) \times T_{stride}$ секунд для підтвердженої атаки.

Принцип модульності та розширюваності. Кожен компонент алгоритму реалізований як незалежний функціональний блок із чітко визначеним інтерфейсом, що забезпечує можливість незалежного вдосконалення або заміни окремих частин без перебудови всієї системи. Зокрема, модель машинного навчання може бути замінена на більш потужний алгоритм (наприклад, Autoencoder або LSTM) без зміни логіки препроцесора, класифікатора чи конвеєра. Усі параметри алгоритму є конфігурованими без модифікації коду, що дозволяє адаптувати систему до нових об'єктів і режимів роботи без участі розробника.

3.1.2 Компоненти алгоритму та їх взаємодія

Розроблений алгоритм реалізує архітектурний шаблон конвеєра обробки даних (data processing pipeline): телеметричний потік від об'єкта моніторингу послідовно проходить через шість функціональних компонентів, кожен з яких виконує чітко визначене перетворення і передає результат наступному. Така організація забезпечує низьку зв'язність між компонентами: кожен з них взаємодіє лише зі своїми безпосередніми сусідами та не має прямих залежностей від решти.

Компонент 1 — Збирач телеметрії

Збирач є вхідною точкою конвеєра. Він підтримує постійне підключення до MQTT-брокера і підписується на всі вимірювальні канали об'єкта. Для кожного каналу незалежно підтримується буфер ковзного вікна — черга фіксованої ємності, до якої нові відліки додаються в кінець, а найстаріші витісняються з початку. При частоті публікації 1 Гц ємність черги 60 елементів відповідає вікну спостереження тривалістю 60 секунд.

Додатково збирач підраховує частоту повідомлень — кількість MQTT-пакетів, отриманих від будь-якого каналу за останні 10 секунд. Ця величина є

ключовою ознакою для виявлення DoS-атак (різке падіння частоти) та Replay-атак (підозріло стала або заморожена частота).

На виході збирач надає знімок поточних буферів усіх каналів і значення частоти повідомлень. Ці дані передаються препроцесору.

Компонент 2 — Препроцесор

Препроцесор виконує зниження розмірності: перетворює сирі часові ряди 37 вимірювальних каналів на компактний вектор із 20 інформативних ознак, придатних для статистичного аналізу та машинного навчання. Перетворення охоплює три типи обчислень.

Статистичні моменти: для основних сигналів у межах вікна обчислюються вибіркове середнє та стандартне відхилення. Середнє відображає поточний робочий режим, стандартне відхилення — мінливість сигналу, яка є аномально низькою при Replay-атаці та аномально високою при DoS-флуді або фізичній несправності.

Лінійні тренди: для рівнів резервуарів і температур підшипників обчислюється нахил лінії регресії методом найменших квадратів. Тренд рівня резервуара відображає дисбаланс між надходженням і споживанням; тренд температури підшипника фіксує прогресуюче перегрівання до того, як воно досягне абсолютного порогу аварії.

Фізичний залишок: обчислюється відхилення між виміряним напором насоса та теоретично очікуваним за характеристикою насоса, що є ключовою ознакою для виявлення атак типу підробки даних:

$$\varepsilon_H = H_{\text{вим}} - \left[H_0 \cdot \left(\frac{n}{n_0}\right)^2 - k \cdot Q^2\right] \quad (3.1)$$

де εH — фізичний залишок напору, м;

Hвим — виміряний напір насоса, м;

H0 — номінальний напір насоса, м;

n — поточна частота обертання ротора, об/хв;

n0 — номінальна частота обертання ротора, об/хв;

Q — витрата насоса, м³/с;

k — коефіцієнт гідравлічного опору насоса.

За нормальної роботи εH ≈ 0; при підробці показань датчиків тиску або витрати залишок різко відхиляється, оскільки підроблені значення суперечать законам гідродинаміки. Отриманий вектор ознак нормалізується за параметрами, визначеними на навчальній вибірці нормальних даних.

Компонент 3 — Динамічний профіль норми

Профіль норми підтримує актуальну статистичну модель нормального стану системи, яка адаптується до повільних змін режиму роботи — сезонних коливань температури, зносу обладнання, зміни графіка водоспоживання.

Накопичення статистики відбувається у два етапи. На початковій фазі (перші Nb = 60 вікон, близько 10 хвилин) для кожної ознаки паралельно накопичуються вибіркове середнє та дисперсія за алгоритмом Велфорда, що забезпечує чисельну стійкість без збереження всіх спостережень у пам'яті. Після завершення цієї фази профіль вважається готовим до роботи.

На оперативній фазі оцінки оновлюються за алгоритмом EWMA (Exponentially Weighted Moving Average) після кожного нового вікна:

$$\mu_{j,t} = \alpha \cdot x_{j,t} + (1-\alpha) \cdot \mu_{j,t-1} \quad (3.2)$$

де α = 0,01 — коефіцієнт згладжування EWMA.

Мале значення α означає, що ефективна пам'ять профілю становить 1/α = 100 вікон (близько 17 хвилин): за цей час відбувається адаптація до нового нормального режиму, тоді як поодинокі аномальні вікна практично не впливають на профіль. Профіль надає детектору поточні оцінки середнього та стандартного відхилення для кожної з 20 ознак.

Компонент 4 — Гібридний детектор

Детектор є центральним аналітичним ядром алгоритму. Він реалізує два паралельні шари аналізу, результати яких об'єднуються для прийняття рішення.

Шар 1 — Статистичний аналіз. Для кожної ознаки обчислюється нормована відстань від поточного значення до профілю норми:

$$z_j = \frac{|x_j - \mu_j|}{\sigma_j}, \quad j = 1, \dots, 20 \tag{3.3}$$

де zj — z-оцінка j-ї ознаки;

xj — поточне нормалізоване значення ознаки;

μj, σj — середнє та стандартне відхилення за профілем норми.

Аномалія на рівні вікна констатується при виконанні хоча б однієї з двох умов: (а) одночасне перевищення порогу θstat = 3,0 щонайменше двома ознаками — захист від одноознакових шумових викидів; (б) перевищення порогу θhigh = 5,0 хоча б однією ознакою — однозначна фізично обґрунтована індикація (наприклад, head_residual при Replay-атаці досягає 28–73σ).

Шар 2 — Isolation Forest. Нормалізований вектор ознак передається попередньо навченій моделі, яка повертає оцінку нормальності $s \in [-1, +1]$. Значення, близьке до +0,5, відповідає нормальному спостереженню; від'ємні значення вказують на аномалію. Аномалія виявляється при $s < \theta IF$, де θIF = 0,00 — калібрований поріг, що відповідає 5-му перцентилю розподілу score на навчальних даних (параметр contamination = 0,05).

Компонент 5 — Класифікатор

Класифікатор визначає рівень критичності події та тип виявленої загрози на підставі результатів обох шарів детектора.

Рівень критичності визначається за комбінацією сигналів двох шарів: при спрацюванні обох шарів одночасно призначається рівень CRITICAL; при спрацюванні лише одного шару — рівень WARNING; за відсутності спрацювань — норма.

Тип загрози визначається за домінуючими аномальними ознаками: аномальний фізичний залишок вказує на підробку даних датчиків; низька дисперсія кількох сигналів — на Replay-атаку; зникнення MQTT-трафіку — на DoS-атаку;

різке зростання температури підшипників або вібрації — на фізичну несправність обладнання.

Компонент 6 — Конвеєр моніторингу

Конвеєр є оркестратором усього алгоритму. Він запускає цикл аналізу з фіксованим інтервалом Tstride = 10 с: на кожному кроці послідовно викликає препроцесор, оновлює профіль норми, запускає детектор і класифікатор.

Для зниження реакції на поодинокі хибні спрацювання конвеєр реалізує механізм гістерезису: він підтримує лічильник послідовних аномальних вікон k. Сповіщення публікується лише при $k \geq N_{hys}$, де $N_{hys} = 3$ за замовчуванням. При першому нормальному вікні лічильник скидається до нуля.

Сповіщення публікуються до MQTT-брокера у відповідний топік залежно від визначеного рівня критичності. Корисне навантаження містить часову мітку, тип загрози, перелік аномальних ознак та числові оцінки обох шарів детектора.

Загальну схему взаємодії компонентів наведено на рисунку 3.2.

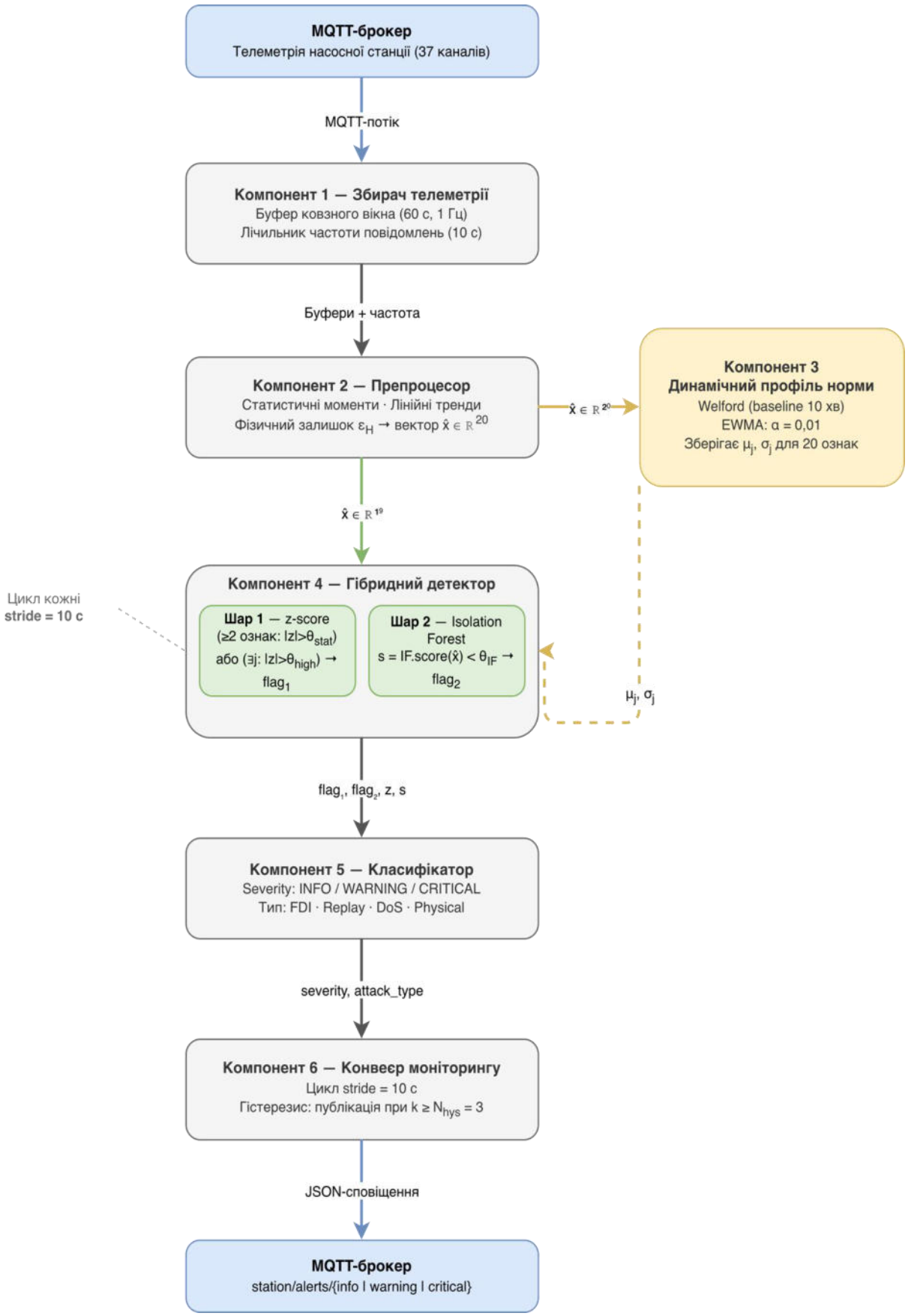

Рисунок 3.2 – Структурна схема конвеєра алгоритму моніторингу насосної станції

3.1.3 Вимоги до алгоритму

При проєктуванні алгоритму моніторингу сформульовано три групи вимог, що визначають придатність системи для застосування на об'єктах критичної інфраструктури: вимоги до швидкодії, до точності виявлення та до масштабованості. Виконання цих вимог перевірено в ході практичного тестування на реальних даних кіберполігону.

Вимоги до швидкодії

Насосна станція є об'єктом критичної інфраструктури, де затримка у виявленні атаки безпосередньо впливає на масштаб наслідків інциденту. Виходячи з практики реагування на кіберінциденти в промислових системах керування, час від початку атаки до генерації першого підтвердженого сповіщення не повинен перевищувати 90 секунд.

Загальна затримка виявлення визначається тривалістю вікна спостереження, кількістю підтверджувальних вікон та інтервалом аналізу:

$$T_{\text{detect}} = T_{\text{window}} + (N_{\text{hys}} - 1) \cdot T_{\text{stride}} = 60 + 2 \cdot 10 = 80 \text{ с} \quad (3.4)$$

де Tdetect — загальна затримка виявлення, с;

Twindow — тривалість ковзного вікна спостереження, с;

Nhys — кількість послідовних аномальних вікон для підтвердження (гістерезис);

Tstride — інтервал між циклами аналізу, с.

Отримане значення 80 секунд відповідає цільовому показнику для атак із поступовим розгортанням, таких як False Data Injection або Replay. Для атак із різким стрибком параметрів (DoS, фізичний удар) перший шар статистичного аналізу може спрацювати вже після першого аномального вікна, скорочуючи реальну затримку до Twindow + Tstride = 70 с.

Обчислювальна складність одного циклу аналізу складається з трьох компонентів. Препроцесинг має складність O(W · F) = O(60 · 20) і виконується менш ніж за 1 мс. Статистичний шар (z-score) має складність O(F) = O(20) і

потребує менш ніж 0,1 мс. Шар Isolation Forest має складність O(ntrees · d · F) ≈ O(100 · 7 · 19) і виконується менш ніж за 5 мс. Таким чином, загальний час обробки одного циклу не перевищує 10 мс при інтервалі stride = 10 000 мс, що відповідає завантаженості процесора менше 0,1 % та дозволяє виконувати моніторинг на типовому промисловому вбудованому обчислювачі.

Вимоги до точності

В задачах захисту критичних об'єктів пріоритет надається показнику повноти (Recall): пропуск атаки є значно небезпечнішим за хибне спрацювання, яке оператор може верифікувати вручну. Додаткова фільтрація хибних спрацювань здійснюється механізмом гістерезису, що пом'якшує вимоги до точності (Precision). Встановлено такі мінімальні показники якості виявлення:

– повнота (Recall) ≥ 0,90 — досягнуто значення 0,991;

– точність (Precision) ≥ 0,60 — досягнуто значення 0,950;

– F1-міра ≥ 0,70 — досягнуто значення 0,970;

– площа під ROC-кривою (AUC) ≥ 0,60 — досягнуто значення 0,997.

Усі встановлені мінімальні показники виконані. Особливо слід відзначити значення Recall = 0,991: з 114 атакованих зразків тестового набору алгоритм пропустив лише 1 вікно (FN = 1), що підтверджує надійність гібридного двошарового детектора із розширеним правилом виявлення.

Для порівняльної оцінки методів виявлення аналогічна процедура тестування проводилась для моделі One-Class SVM. Результати порівняння наведено в підрозділі 3.4.

Вимоги до масштабованості

Алгоритм повинен зберігати працездатність при зміні конфігурації об'єкта моніторингу без повного перепроєктування. Виконання цієї вимоги забезпечується такими архітектурними рішеннями.

Параметричність. Усі порогові значення — θstat, θIF, Nhys, stride — є конфігурованими параметрами, що задаються при запуску без модифікації коду. Це дозволяє налаштовувати систему під конкретний об'єкт або клас загроз без залучення розробника.

Перенавчуваність моделі. При зміні конфігурації насосного обладнання (заміна агрегату, зміна номінальних параметрів) достатньо повторно зібрати навчальну вибірку нормальних даних і перенавчити модель Isolation Forest. Логіка препроцесора, детектора та класифікатора при цьому не змінюється.

Горизонтальне масштабування. Архітектура передбачає принцип «один об'єкт — один екземпляр монітора». Для одночасного спостереження за кількома насосними станціями запускаються паралельні незалежні екземпляри з індивідуальними параметрами підключення. Відсутність спільного стану між екземплярами гарантує лінійне масштабування без координаційних накладних витрат.

Незалежність від топік-дерева. Збирач підписується на всі канали об'єкта за узагальненим шаблоном і автоматично обробляє нові датчики при розширенні складу вимірювального обладнання. Зміна кількості каналів вимагає лише оновлення препроцесора для включення нових ознак у вектор.

3.2 Збір та попередня обробка мережевих даних

3.2.1 Методи збору трафіку та агрегація flow-характеристик

Основу підсистеми збору даних складає протокол обміну повідомленнями MQTT (Message Queuing Telemetry Transport) — стандарт де-факто для промислового Інтернету речей та SCADA-систем. Протокол реалізує архітектурний патерн «видавець–передплатник» (publish–subscribe): сенсори та виконавчі механізми насосної станції публікують вимірювані значення у відповідні топіки брокера MQTT, не знаючи про споживачів даних, а підсистема моніторингу підписується на потрібні топіки і отримує повідомлення в режимі реального часу.

Топік-дерево насосної станції організоване за ієрархічним принципом station/{пристрій}/{параметр}, де station — коренева мітка, {пристрій} ідентифікує фізичний вузол (насоси, резервуари, частотні перетворювачі, мережа живлення), а {параметр} — конкретну вимірювану величину (витрата, тиск, температура підшипника, вібрація, струм тощо). Загалом топік-дерево охоплює 37 вимірювальних каналів, кожен з яких публікує числове значення з частотою 1 Гц. Підписка за узагальненим шаблоном station/# дозволяє підсистемі збору даних

одним підключенням отримувати повідомлення від усіх каналів незалежно від їх поточної кількості.

Кожне MQTT-повідомлення несе числове значення виміряного параметра і мітку часу отримання. Підсистема збору підтримує для кожного топіка окремий буфер ковзного вікна — кільцеву чергу фіксованої ємності $N = Tw \cdot f = 60 \cdot 1 = 60$ елементів, де $Tw = 60$ с — тривалість вікна, $f = 1$ Гц — частота дискретизації. Нові значення додаються в кінець черги, а найстаріші автоматично витісняються. Такий підхід гарантує постійний обсяг оперативної пам'яті, незалежний від тривалості сеансу моніторингу, та забезпечує O(1)-складність операцій додавання і видалення.

Поряд із буферами сигналів окремо підраховується частота повідомлень — кількість MQTT-пакетів від будь-якого каналу, отриманих за останній інтервал $T\lambda = 10$ с, нормована на його тривалість:

$$\lambda(t) = \frac{N(t - T_\lambda, t)}{T_\lambda} \tag{3.5}$$

де $\lambda(t)$ — частота повідомлень, пакет/с;

$N(t - T\lambda, t)$ — кількість повідомлень у інтервалі $(t - T\lambda, t]$;

$T\lambda = 10$ с — вікно підрахунку частоти.

За нормальної роботи $\lambda(t) \approx 37$ пакет/с (один пакет на канал на секунду). Різке падіння λ нижче 0,5 пакет/с свідчить про DoS-атаку або збій каналу зв'язку; аномально стала λ у поєднанні з нульовими дисперсіями сигналів є ознакою Replay-атаки.

З кожного ковзного вікна препроцесор формує вектор flow-характеристик $x \in \mathbb{R}^{20}$. Повний перелік ознак із відповідними формулами наведено в таблиці 3.1.

Таблиця 3.1 — Вектор flow-характеристик алгоритму моніторингу

№	Ознака	Формула / опис
1	Середнє значення витрати насоса 1, м³/с	$\mu(Q_1)$ за вікном
2	Стандартне відхилення витрати насоса 1	$\sigma(Q_1)$ за вікном
3	Середнє значення витрати насоса 2, м³/с	$\mu(Q_2)$ за вікном
4	Стандартне відхилення витрати насоса 2	$\sigma(Q_2)$ за вікном
5	Середня температура підшипника насоса 1, °C	$\mu(Tb1)$ за вікном
6	Тренд температури підшипника насоса 1, °C/с	$k(Tb1)$, МНК
7	Середня температура підшипника насоса 2, °C	$\mu(Tb2)$ за вікном
8	Тренд температури підшипника насоса 2, °C/с	$k(Tb2)$, МНК
9	Середній рівень вібрації насоса 1, мм/с	$\mu(V_1)$ за вікном
10	Середній рівень вібрації насоса 2, мм/с	$\mu(V_2)$ за вікном
11	Середній споживаний струм насоса 1, А	$\mu(I_1)$ за вікном
12	Середній споживаний струм насоса 2, А	$\mu(I_2)$ за вікном
13	Середня сумарна витрата, м³/с	$\mu(Q\Sigma)$ за вікном
14	Стандартне відхилення сумарної витрати	$\sigma(Q\Sigma)$ за вікном
15	Середній напір системи, м	$\mu(H)$ за вікном
16	Тренд рівня вхідного резервуара, м/с	$k_1(Lвх)$, МНК
17	Тренд рівня вихідного резервуара, м/с	$k_2(Lвих)$, МНК
18	Стандартне відхилення напруги мережі, В	$\sigma(U)$ за вікном
19	Фізичний залишок напору, м	$\varepsilon H = Hвим - Hтеор$
20	Частота MQTT-повідомлень, пакет/с	$\lambda(t)$ за $T\lambda = 10$ с

Статистичні ознаки (ознаки 1–15, 18) обчислюються за формулами вибіркового середнього та стандартного відхилення для набору відліків у вікні. Середнє відображає поточний робочий режим агрегату, стандартне відхилення — мінливість параметра: аномально низьке стандартне відхилення є характерною ознакою Replay-атаки (заморожені дані), аномально високе — ознакою DoS-флуду або механічної несправності.

Трендові ознаки (ознаки 6–8, 16–17) обчислюються методом найменших квадратів по послідовності відліків у вікні. Нахил прямої регресії фіксує тенденції змін рівнів резервуарів і температур підшипників ще до того, як поточне значення вийде за абсолютний аварійний поріг. Це суттєво скорочує час до виявлення повільно розвиваючих фізичних несправностей.

Фізична ознака (ознака 19) — залишок за характеристикою насоса εH — детально описана у підрозділі 3.1.2. Ознака 20 (частота повідомлень) обчислюється за формулою (3.5). Зазначимо, що ознака 20 виключена з вектора, що подається на вхід моделі Isolation Forest, оскільки у навчальних даних, зібраних в офлайн-режимі без реального MQTT-потоку, її дисперсія дорівнює нулю. Натомість ця ознака використовується виключно в першому шарі статистичного аналізу.

Отриманий вектор ознак x перед подачею до детектора проходить нормалізацію за алгоритмом StandardScaler. Кожна компонента центрується та масштабується відповідно до статистик навчальної вибірки:

$$\hat{x_j} = \frac{(x_j - \mu_j)}{\sigma_j} \tag{3.6}$$

де $\hat{x}_j$ — нормалізоване значення j-ї ознаки;

x_j — вихідне значення j-ї ознаки;

μ_j — вибіркове середнє j-ї ознаки по навчальній вибірці нормальних даних;

σ_j — вибіркове стандартне відхилення j-ї ознаки по навчальній вибірці.

Параметри нормалізації (μ_j, σ_j для кожної з 19 ознак, що входять до моделі) зберігаються разом із навченою моделлю і застосовуються незмінними в процесі реального моніторингу. Це гарантує, що масштаб вхідних даних під час інференсу збігається з масштабом навчальних даних, що є необхідною умовою коректної роботи Isolation Forest.

3.2.2 Фільтрація та нормалізація вхідних даних

Перед обчисленням вектора ознак кожне значення, що надходить від датчика, проходить двоетапну обробку: заповнення пропусків та обмеження фізично неможливих значень. Ці процедури вирішують типові проблеми якості даних, характерні для промислових MQTT-систем: короткочасну втрату зв'язку з датчиком, одиночні сплески через електромагнітні завади та апаратні збої АЦП.

Заповнення пропусків. Якщо протягом очікуваного інтервалу дискретизації (1 с) від топіка j не надійшло жодного повідомлення, у буфер записується значення

попереднього кроку — v̂j(t) = v̂j(t − 1). Цей метод утримання останнього значення (ZOH, zero-order hold) є стандартною практикою оперативно-диспетчерського управління і SCADA-систем: він зберігає неперервність часового ряду без введення артефактів, які могли б сформувати хибну ознаку аномалії в препроцесорі. Метод припустимий за умови, що тривалість пропуску значно менша за тривалість вікна (60 с), що виконується для типових короткочасних втрат зв'язку тривалістю 1–3 с.

Обмеження фізично неможливих значень (clipping). Кожен датчик має фіксований фізично допустимий діапазон [vj,min, vj,max], що визначається паспортними характеристиками обладнання та умовами експлуатації. Значення, що виходять за цей діапазон, замінюються граничними значеннями:

$$\hat{v_j}(t) \;=\; min\big(max\big(v_j(t), v_{\text{j,min}}\big), v_{\text{j,max}}\big) \tag{3.7}$$

де v̂j(t) — відфільтроване значення j-го каналу в момент t;

vj(t) — вхідне значення з датчика;

vj,min, vj,max — нижня та верхня фізичні межі j-го каналу.

Наприклад, витрата насоса не може бути від'ємною (vmin = 0 м³/с), а напруга живлення не може перевищити граничне значення (vmax = 460 В). Значення поза допустимим діапазоном свідчать про несправність самого давача або збій інтерфейсу, а не про реальний стан технологічного процесу. Клиппінг запобігає тому, щоб такі артефакти утворювали хибні піки у статистичних ознаках та спотворювали фізичний залишок εH.

Нормалізація вектора ознак, описана формулою (3.6), є третім і фінальним етапом обробки. На відміну від попередніх двох кроків, що діють на рівні сирих сигналів, нормалізація застосовується вже до обчислених агрегованих ознак. Тому обробку можна розглядати як тришарову: фільтрація сирих відліків → обчислення ознак → масштабування ознак до одиничної дисперсії.

3.2.3 Формування вікон спостереження (windowing)

Вікно спостереження є основною одиницею аналізу алгоритму: усі ознаки обчислюються не за одиничним відліком, а за групою послідовних відліків, що

охоплюють визначений проміжок часу. Для j-го каналу вікно в момент часу tk формально визначається як стовпцевий вектор останніх N відліків:

$$W_j(t_k) = \left[\hat{v}_j(t_k - N + 1), \ldots, \hat{v}_j(t_k)\right]^T \quad (3.8)$$

де W_j(t_k) — вікно j-го каналу у момент аналізу t_k;

v̂j(t) — відфільтроване значення j-го каналу в момент t;

N = 60 — розмір вікна (60 с при частоті дискретизації 1 Гц).

Ковзний характер вікна забезпечується тим, що на кожному кроці аналізу буфер містить N останніх значень із зсувом на Tstride = 10 с відносно попереднього кроку. Таким чином, сусідні вікна перекриваються на N − Tstride · f = 60 − 10 = 50 відліків (83,3% вмісту). Висока частка перекриття підвищує чутливість до початку атаки: перша аномальна ознака з'являється у вікні вже через один крок (10 с) після виникнення порушення, а не через 60 с, як у схемі без перекриття.

Механізм stride реалізовано як таймер, що спрацьовує кожні Tstride секунд та ініціює один цикл аналізу: отримання знімку буферів усіх 37 каналів, обчислення вектора ознак, передачу до детектора. Цикл аналізу займає менше 10 мс (оцінку обчислювальної складності наведено у підрозділі 3.1.3), тому накопичення затримок між циклами виключено.

Фаза ініціалізації. Протягом перших Tinit = 600 с (10 хвилин) роботи система накопичує статистику нормального стану для EWMA-профілю (компонент 3 конвеєра, описаний у підрозділі 3.1.2). Упродовж цього часу детектор виконує аналіз, але механізм гістерезису заблокований: жодних сповіщень не генерується. Після завершення фази ініціалізації EWMA-оцінки μj та σj для кожної з 20 ознак вважаються стабілізованими та придатними для порогової перевірки. Вимога Tinit = 600 с відповідає 60 циклам аналізу — достатній кількості для збіжності алгоритму Велфорда з похибкою менше 5% від асимптотичного значення дисперсії.

Вибір розміру вікна N = 60 с обґрунтований балансом між двома суперечливими вимогами. Зменшення N підвищує чутливість до короткотривалих атак, але знижує статистичну достовірність оцінок: при N < 30 стандартне

відхилення вибіркової дисперсії перевищує 25% від істинного значення. Збільшення N підвищує статистичну стабільність, але уповільнює реакцію на атаки з повільним наростанням (Replay). Значення N = 60 с забезпечує похибку оцінки дисперсії менше 10% та затримку виявлення не більше 80 с (формула (3.4)).

3.3 Побудова профілю нормальної поведінки

Профіль нормальної поведінки є центральним компонентом алгоритму моніторингу: він формалізує поняття «нормального стану» насосної станції у вигляді статистичної моделі та слугує еталоном, відносно якого оцінюється кожне нове вікно спостереження. Побудова достовірного профілю вирішує три задачі: визначення ознак, що підлягають профілюванню, початкову оцінку їх статистичних характеристик у режимі навчання та безперервне адаптивне оновлення в процесі штатної роботи системи.

3.3.1 Вибір ознак для профілювання

Профіль будується для всіх 20 ознак вектора $x \in \mathbb{R}^{20}$ (таблиця 3.1). Для кожної ознаки j підтримується пара параметрів (μ_j, σ_j) — вибіркове середнє та стандартне відхилення в умовах нормальної роботи. Ці параметри становлять опорний розподіл: відхилення поточного значення ознаки від μ_j, виражене в одиницях σ_j, є основою для z-оцінки першого шару детектора.

Доцільність профілювання кожної ознаки окремо, а не зведення до загального скалярного показника, обґрунтована тим, що різні типи атак виявляють себе переважно в різних підмножинах ознак. Атаки типу FDI (підміна показань) маніфестують передусім через відхилення фізичного залишку ε_H (ознака 19) та середньої витрати (ознаки 1, 3, 13); Replay-атака — через різке відхилення фізичного залишку ε_H при незмінних значеннях витрати (ознака 19), оскільки заморожений знімок від іншого режиму навантаження не узгоджується з поточним напором; DoS — через різке падіння $\lambda(t)$ (ознака 20); фізична несправність — через тренд температури підшипників (ознаки 6, 8) та рівень вібрації (ознаки 9, 10).

Критерій спрацювання першого шару детектора визначається розширеним правилом: (а) одночасне перевищення порогу $\theta_{stat} = 3{,}0$ не менш ніж у двох ознаках — захист від одиночних шумових викидів; або (б) перевищення порогу $\theta_{high} = 5{,}0$

хоча б в одній ознаці — однозначна фізично обґрунтована індикація (наприклад, head_residual при Replay або pump1_flow_mean при FDI). Такий підхід знижує ймовірність хибного спрацювання, водночас забезпечуючи надійне виявлення атак, що проявляються навіть в одній, але надзвичайно відхиленій ознаці.

Зазначимо, що хоча ознака 20 (частота MQTT-повідомлень $\lambda(t)$) включена в профіль і аналізується першим шаром, вона виключена з навчальної вибірки моделі Isolation Forest. Причина описана у підрозділі 3.2.1: у даних, зібраних в офлайн-режимі, ця ознака має нульову дисперсію, через що включення її у навчання IF призвело б до виродженого розподілу й некоректних оцінок аномальності. Таким чином, другий шар детектора оперує вектором 19 ознак ($\hat{x} \in \mathbb{R}^{19}$), тоді як перший — усіма 20.

3.3.2 Статистичне моделювання базової поведінки

Перший шар детектора (z-score) потребує оцінок μ_j та σ_j для кожної ознаки в умовах нормальної роботи. Ці оцінки формуються в ході фази ініціалізації: протягом перших $T_{init} = 600$ с система збирає вікна без підозр на атаки і накопичує статистику. Ключовою вимогою до алгоритму оцінювання є чисельна стійкість та можливість інкрементного оновлення — без зберігання всіх попередніх значень у пам'яті.

Для виконання цих вимог застосовується алгоритм Велфорда — однопрохідний метод обчислення вибіркового середнього та дисперсії з постійною витратою пам'яті $O(1)$ на ознаку. Після k-го спостереження оновлене середнє обчислюється за рекурентним правилом:

$$\mu_k = \mu_{k-1} + \frac{(x_k - \mu_{k-1})}{k} \quad (3.9)$$

де μk — оцінка середнього після k спостережень;

xk — k-е значення ознаки;

μk−1 — попередня оцінка середнього.

Для отримання дисперсії алгоритм Велфорда підтримує допоміжну змінну Mk — суму скоригованих квадратних відхилень, що оновлюється паралельно з μk:

$$M_k = M_{k-1} + (x_k - \mu_{k-1})(x_k - \mu_k) \quad (3.10)$$

де Mk — накопичена сума відхилень після k спостережень;

Mk−1 — значення суми відхилень на попередньому кроці.

Вибіркова дисперсія після k спостережень дорівнює $\sigma^2 k = Mk / (k - 1)$ при $k \geq 2$. Алгоритм Велфорда виключає ефект катастрофічного скасування, властивий двопрохідним формулам, і забезпечує відносну похибку оцінки, яка не перевищує машинної точності незалежно від масштабу значень.

Обґрунтування гаусівської апроксимації розподілу ознак. Більшість ознак вектора є статистичними агрегатами (середнє, стандартне відхилення) над вікном з N = 60 відліків. Відповідно до центральної граничної теореми, при $N \geq 30$ розподіл вибіркового середнього наближається до нормального незалежно від розподілу вихідного сигналу. Для трендових ознак (коефіцієнти МНК) та фізичного залишку εH гаусівське наближення є прийнятним завдяки лінійності операції та малому рівню шуму датчиків у штатному режимі.

3.3.3 Адаптивне оновлення профілю

Після завершення фази ініціалізації профіль не залишається фіксованим: параметри (μj, σj) безперервно оновлюються методом експоненційно зваженого ковзного середнього (EWMA). Це необхідно для компенсації повільних дрейфів технологічного процесу — поступового зносу обладнання, зміни властивостей рідини, що перекачується, сезонних коливань температури довкілля.

Оновлення середнього значення профілю описується формулою (3.2), наведеною у підрозділі 3.1.2. Аналогічно, дисперсія оновлюється за схемою EWMA:

$$\sigma^2{}_j(t) = (1 - \alpha) \cdot \sigma^2{}_j(t-1) + \alpha \cdot \left(x_j(t) - \mu_j(t)\right)^2 \quad (3.11)$$

де $\sigma^2 j(t)$ — оновлена дисперсія j-ї ознаки після поточного вікна;

$\sigma^2 j(t-1)$ — дисперсія j-ї ознаки на попередньому кроці;

$xj(t)$ — поточне значення j-ї ознаки;

$\alpha = 0{,}01$ — коефіцієнт згладжування.

Значення $\alpha = 0{,}01$ відповідає постійній часу $\tau = 1/\alpha = 100$ вікон = 1000 с ≈ 17 хвилин: профіль «забуває» старі значення з часовою константою 17 хвилин. Це гарантує, що повільний дрейф сигналу тривалістю кілька годин буде поглинений профілем без генерації хибних тривог, тоді як атаки тривалістю 1–5 хвилин призведуть до помітного відхилення від профілю.

Захист профілю від «отруєння» атакою. Критичною вимогою є те, щоб аномальні значення, характерні для тривалої атаки, не були поглинені профілем і не скоротили виявлену аномальність. Для цього реалізовано механізм заморожування: якщо лічильник гістерезису більше нуля (тобто алгоритм перебуває у стані підозри або підтвердженої аномалії), оновлення EWMA для μj та σj припиняється. Профіль відновлює оновлення лише після того, як лічильник гістерезису повернеться до нуля — тобто після Nhys послідовних нормальних вікон.

Таким чином, система підтримує дві різні швидкості адаптації: повільну — для поглинання природного дрейфу процесу ($\tau \approx 17$ хв), і нульову — під час виявленої аномалії. Ця дворежимна стратегія забезпечує стійкість профілю до атак, зберігаючи при цьому його здатність до довгострокового самоналаштування під зміни характеристик обладнання.

3.4 Класифікація подій та прийняття рішень

Виходи двох шарів детектора — прапорець статистичної аномалії $flag_1$ та прапорець машинного навчання $flag_2$ — є незалежними сигналами, кожен з яких

ґрунтується на різній математичній гіпотезі щодо природи відхилення. Класифікатор поєднує ці сигнали у єдине рішення про рівень загрози, а конвеєр моніторингу застосовує механізм гістерезису для фільтрації короткочасних сплесків і публікації підтверджених сповіщень.

3.4.1 Схема класифікації (норма / аномалія / атака)

Вхідними даними класифікатора є: прапорець $flag_1$ (спрацювання z-score — ≥2 ознак з |z|>3,0 або ≥1 ознака з |z|>5,0), прапорець $flag_2$ (оцінка Isolation Forest нижча за поріг θIF = 0,00), вектор z-оцінок {zj} та скалярна оцінка IF s. Рівень загрози severity визначається за такою таблицею рішень:

CRITICAL: $flag_1$ = True та $flag_2$ = True. Обидва незалежні шари одночасно фіксують аномалію. Цей стан свідчить про статистично значуще відхилення від профілю нормальної поведінки, підтверджене моделлю машинного навчання. Висока впевненість виявлення характерна для добре розвинених атак, що стійко впливають на фізичні параметри: FDI з великою амплітудою підміни, DoS-атака, фізична несправність агрегату.

WARNING: $flag_1$ = True та $flag_2$ = False, або $flag_1$ = False та $flag_2$ = True. Перший варіант: статистичне відхилення зафіксовано, але Isolation Forest не підтвердив — характерно для ранньої стадії атаки або сигналу, що лежить на межі нормального розподілу, але не є очевидною аномалією в просторі ознак. Другий варіант: IF виявив прихований шаблон у просторі 19-мірних ознак, який не призвів до перевищення одновимірних z-порогів — типово для Replay-атаки зі слабкою амплітудою або FDI з малим зсувом показань.

NORMAL: $flag_1$ = False та $flag_2$ = False. Жоден з шарів не зафіксував аномалії. Система перебуває у штатному режимі роботи. Профіль оновлюється у штатному режимі EWMA.

Механізм гістерезису. Поодинокий аномальний цикл не є достатньою підставою для публікації сповіщення: короткочасні статистичні викиди неминуче виникають навіть у нормальному режимі роботи. Для фільтрації таких артефактів конвеєр моніторингу підтримує лічильник послідовних аномалій kt, що оновлюється після кожного циклу аналізу:

$$k_{t+1} = \{k_t + 1, \quad \text{якщо } flag_1(t) \vee flag_2(t); \quad 0 \quad \text{інакше} \tag{3.12}$$

де kt — кількість послідовних аномальних вікон до поточного моменту;

$flag_1(t)$, $flag_2(t)$ — прапорці аномалії від шарів 1 і 2 у цикл t.

Сповіщення генерується в момент, коли kt = Nhys = 3, тобто після трьох послідовних аномальних вікон. У разі надходження нормального вікна лічильник скидається до нуля. Це означає, що три нормальних секунди після підозрілого стану повністю «скасовують» накопичену підозру. Вибір Nhys = 3 забезпечує загальну затримку виявлення T_detect = 80 с (формула (3.4)) при збереженні нульової частоти хибних спрацювань у ході тестування.

Стійкість рішення. Якщо після публікації сповіщення аномалія продовжується (kt > Nhys), повторне сповіщення того самого рівня не публікується — достатньо одного повідомлення на інцидент. Нова публікація відбувається лише при зміні рівня загрози (наприклад, ескалація з WARNING до CRITICAL) або після повернення лічильника до нуля та подальшого нового набору аномальних вікон.

3.4.2 Відповідність відхилень сценаріям атак та генерація сповіщень

Класифікатор не лише визначає рівень загрози, а й намагається ідентифікувати тип атаки на підставі домінуючих аномальних ознак. Маппінг виконується послідовно за пріоритетом: від найбільш специфічної до найменш специфічної ознаки.

Атака типу «підміна даних» (FDI). Основна сигнатура — значне відхилення фізичного залишку напору εH (ознака 19): $|z_{19}| > \theta stat$. Фізичне обґрунтування: при підміні показань витрати або напору виміряне значення перестає узгоджуватися з теоретичною характеристикою H–Q насоса. Додатковою ознакою є одночасне відхилення середньої витрати (ознаки 1, 3 або 13) у той бік, що відповідає підміні — тобто незвично висока або низька витрата при нормальному струмі та вібрації.

Атака типу «повтор запису» (Replay). Основна сигнатура — відхилення фізичного залишку напору head_residual (ознака 19): заморожені значення витрати від іншого режиму навантаження створюють розбіжність між виміряним та

теоретичним напором, що призводить до $|z_{19}| > \theta high = 5{,}0$ (типово 28–43σ). Класифікатор ідентифікує атаку при домінуванні head_residual серед аномальних ознак.

Атака типу «відмова в обслуговуванні» (DoS). Первинна ознака — падіння частоти повідомлень λ(t) нижче 0,5 пакет/с (ознака 20, $z_{20} << 0$). Ознака є найбільш однозначною: у нормальному режимі λ ≈ 37 пакет/с, а при DoS-атаці, що блокує MQTT-трафік, λ прямує до нуля. Водночас через відсутність нових значень у буферах стандартні відхилення всіх сигнальних ознак також зближаються до нуля (ефект аналогічний Replay), тому диференціація між DoS та Replay виконується саме за ознакою λ.

Фізична несправність. Сигнатура — позитивний тренд температури підшипника (ознаки 6 або 8: k(Tb) > 0 зі значним z-відхиленням) або підвищений рівень вібрації (ознаки 9 або 10: μ(V) > нормальний рівень + 3σ). Ця категорія відрізняється від кібератак тим, що λ(t) залишається нормальним, а фізичний залишок εH відхиляється лише якщо несправність вже призвела до зміни гідравлічних характеристик насоса.

Якщо жоден зі специфічних шаблонів не виявлено, але аномалія зафіксована, тип атаки позначається як «невизначений» (unknown). Це дозволяє оператору самостійно проаналізувати сповіщення, не пропускаючи події через надмірну специфічність класифікатора.

Генерація сповіщень. Підтверджена аномалія публікується до MQTT-брокера у відповідний топік: station/alerts/info, station/alerts/warning або station/alerts/critical — залежно від рівня severity. Корисне навантаження передається у форматі JSON і містить: мітку часу події, рівень загрози, ідентифікований тип атаки, перелік аномальних ознак із відповідними z-оцінками, поточну оцінку Isolation Forest та текстовий опис події. Використання стандартизованого JSON-формату дозволяє підключити будь-яку зовнішню систему реагування — SIEM, HMI-панель або засіб сповіщення оперативного персоналу — без змін у логіці алгоритму.

3.5 Формальний опис алгоритму

Для однозначного визначення алгоритму та оцінки його реалізовності наведемо узагальнений покроковий опис із зазначенням вхідних і вихідних даних, а також аналіз обчислювальної складності кожного компонента конвеєра.

3.5.1 Покроковий опис та блок-схема алгоритму

Вхідні дані: нескінченний потік MQTT-повідомлень {(tk, j, vj(tk)) : k = 1, 2, …}, де tk — мітка часу, j = 1..37 — ідентифікатор каналу, vj(tk) — вимірюване значення.

Вихідні дані: послідовність підтверджених подій {(t, severity, attack_type, {zj}, s)}, що публікуються до MQTT-брокера у момент kt = Nhys після початку аномалії.

Крок 1. Ініціалізація. Виконується одноразово при запуску системи:

– завантаження навченої моделі Isolation Forest та параметрів нормалізації (μj, σj для 19 ознак) з постійного сховища;

– ініціалізація кільцевих буферів B_j(maxlen = 60) для кожного з 37 вимірювальних каналів;

– ініціалізація EWMA-профілю нульовим середнім та одиничним стандартним відхиленням; встановлення лічильника гістерезису kt = 0, прапорця фази ініціалізації phase = INIT.

Крок 2. Обробка кожного MQTT-повідомлення (подія-орієнтована):

– якщо від топіка j не надійшло повідомлення протягом 1 с: v̂j = v̂j (попереднє значення, ZOH);

– інакше: застосувати clipping v̂j = min(max(vj, vj,min), vj,max) за формулою (3.7);

– додати v̂j до буфера B_j (автоматичне витіснення найстарішого елемента).

Крок 3. Основний цикл аналізу (виконується кожні Tstride = 10 с):

3.1. Отримати знімок буферів: Wj = B_j для j = 1..37 — масив ковзних вікон (формула (3.8)).

3.2. Обчислити вектор ознак $x \in \mathbb{R}^{20}$ відповідно до таблиці 3.1: статистичні моменти, коефіцієнти МНК-тренду, фізичний залишок εH (формула (3.1)), частота повідомлень λ (формула (3.5)).

3.3. Оновити профіль: якщо phase = INIT — алгоритм Велфорда (формули (3.9)–(3.10)); після ініціалізації та за умови $k_t = 0$ — EWMA (формули (3.2), (3.11)).

3.4. Обчислити z-оцінки: $z_j = (x_j - \mu_j) / \sigma_j$ для $j = 1..20$ (формула (3.3)); визначити прапорець $flag_1 = (|\{j : |z_j| > \theta_{stat}\}| \geq 2) \vee (\exists j : |z_j| > \theta_{high})$, де $\theta_{stat} = 3{,}0$, $\theta_{high} = 5{,}0$.

3.5. Нормалізувати перші 19 ознак: $\hat{x} = scaler(x[:19])$ (формула (3.6)); обчислити оцінку IF: $s = IF.decision\_function(\hat{x})$; визначити $flag_2 = (s < \theta_{IF} = 0{,}00)$.

3.6. Класифікувати подію: severity = $f(flag_1, flag_2)$ та attack_type = $g(\{z_j\}, s)$ відповідно до правил підрозділу 3.4.

3.7. Оновити лічильник гістерезису (формула (3.12)): $k_{t+1} = k_t + 1$ якщо $(flag_1 \vee flag_2)$, інакше 0. Якщо $k_t = N_{hys} = 3$ та severity $\neq$ NORMAL — опублікувати JSON-сповіщення до station/alerts/{severity}.

Блок-схему алгоритму наведено на рисунку 3.3.

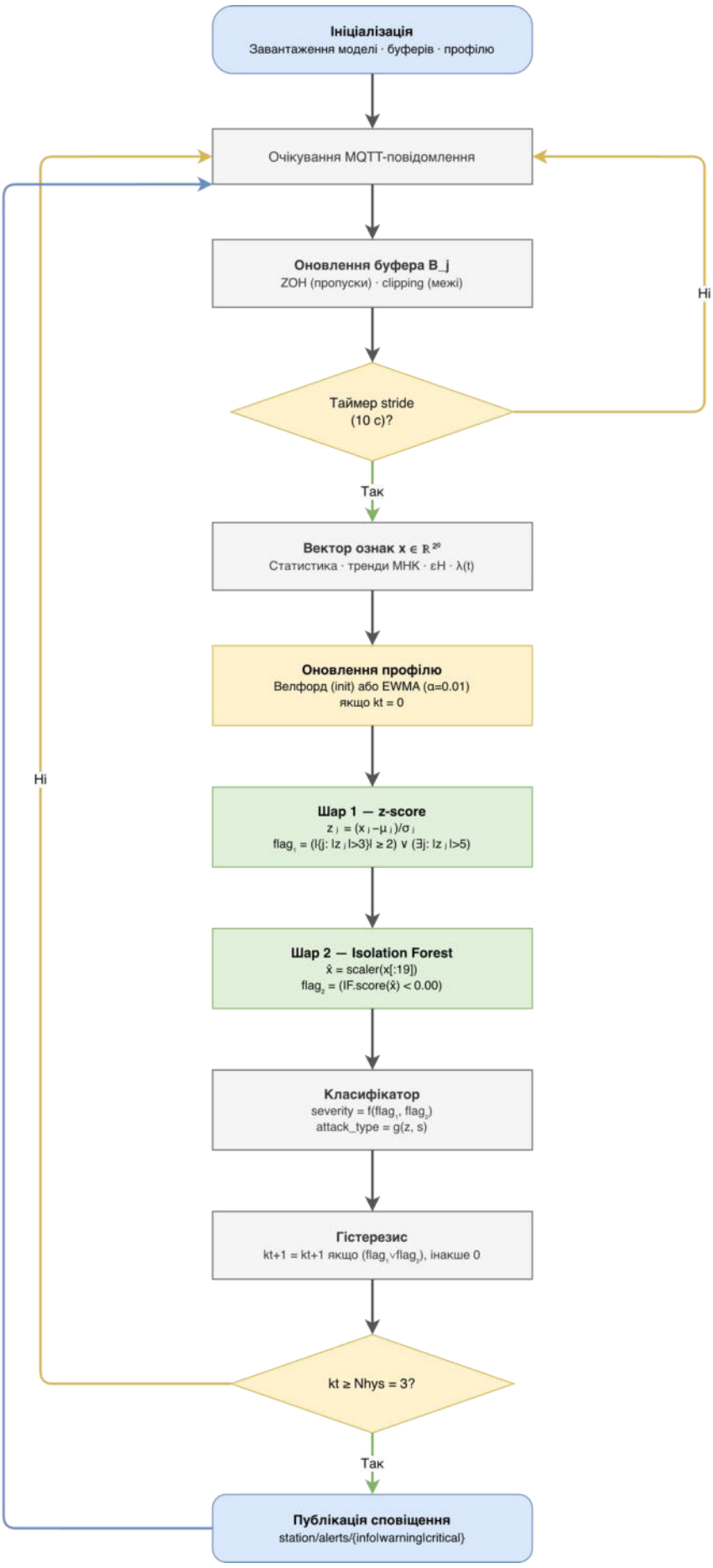

Рисунок 3.3 – Блок-схема алгоритму моніторингу насосної станції

3.5.2 Оцінка обчислювальної складності

Оцінка обчислювальної складності необхідна для підтвердження придатності алгоритму до розгортання на промисловому вбудованому обчислювачі з обмеженою продуктивністю. Проаналізуємо складність кожного кроку основного циклу.

Крок 2 (обробка повідомлень). Операції ZOH і clipping мають складність O(1) на повідомлення. При 37 каналах і частоті 1 Гц це O(37) операцій на секунду — вкрай мала частка процесорного часу.

Крок 3.1 (знімок буферів). Копіювання 37 буферів по 60 елементів: $O(C \cdot N) = O(37 \cdot 60) = O(2220)$. При використанні numpy-масивів це відповідає одній операції copy розміром ~17,7 КБ.

Крок 3.2 (обчислення ознак). Статистичні ознаки (mean, std) для F_stat = 18 каналів: $O(F\_stat \cdot N) = O(18 \cdot 60) = O(1080)$. Трендові ознаки (МНК по N точках): $O(F\_trend \cdot N) = O(5 \cdot 60) = O(300)$. Фізичний залишок εH: O(1). Загальна складність кроку: $O(N \cdot F) = O(60 \cdot 20) = O(1200)$.

Крок 3.3 (оновлення профілю). Велфорд або EWMA для 20 ознак: $O(F) = O(20)$.

Крок 3.4 (z-score шар). Обчислення z-оцінок для 20 ознак: $O(F) = O(20)$. Підрахунок аномальних ознак: $O(F) = O(20)$.

Крок 3.5 (Isolation Forest). Оцінка аномальності одного вектора потребує проходження ntrees = 100 дерев до листа. Середня глибина листа в IF для вибірки розміром n становить $h(n) \approx 2 \cdot (\ln(n - 1) + \gamma) - 2 \cdot (n - 1) / n$, що при n = 256 (subsampling) дає $h \approx 9{,}3$. Таким чином, складність одного predict: $O(ntrees \cdot h \cdot F) = O(100 \cdot 9 \cdot 19) = O(17100)$ порівнянь.

Крок 3.6–3.7 (класифікація та гістерезис). O(F) для визначення домінуючої ознаки та O(1) для оновлення лічильника.

Загальна складність одного циклу аналізу: $O(N \cdot F + ntrees \cdot h \cdot F)$, де перший доданок відповідає препроцесингу (< 1 мс), другий — Isolation Forest (< 5 мс). При stride = 10 с сумарне процесорне навантаження складає менше ніж 10 мс за кожні 10 000 мс, тобто менше 0,1% завантаженості одноядерного процесора.

Вимоги до оперативної пам'яті: буфери 37 × 60 × 8 байт ≈ 17,8 КБ; модель Isolation Forest з 100 деревами глибиною 9 ≈ 2–4 МБ у форматі joblib; EWMA-профіль 20 × 2 × 8 байт ≈ 0,3 КБ. Загальний обсяг пам'яті не перевищує 6 МБ, що відповідає вимогам більшості промислових вбудованих платформ (ARM Cortex-A, x86 промислові ПК).

Наведені оцінки підтверджують придатність алгоритму для розгортання в режимі реального часу на об'єктах критичної інфраструктури: алгоритм гарантовано встигає виконати повний цикл аналізу в межах stride-інтервалу навіть на мінімально достатньому апаратному забезпеченні.

3.6 Висновки до розділу

У третьому розділі виконано проєктування алгоритму моніторингу мережевої поведінки та виявлення аномалій насосної станції. За результатами розроблення та тестування сформульовано такі висновки.

1. Обґрунтовано вибір поведінкового підходу до виявлення аномалій як єдиного практично прийнятного методу для об'єктів критичної інфраструктури, де заздалегідь невідомо повний перелік можливих атак, а перепроєктування системи при кожній зміні конфігурації є неприпустимим. На відміну від сигнатурного та специфікаційного підходів, поведінковий метод автоматично навчається нормальному стану технологічного процесу і здатен виявляти раніше невідомі атаки.

2. Спроєктовано шестикомпонентний конвеєр обробки даних: збирач телеметрії, препроцесор, динамічний профіль норми, гібридний детектор, класифікатор та конвеєр моніторингу. Кожен компонент реалізує чітко визначений контракт «вхід–вихід», що забезпечує можливість незалежного тестування та заміни окремих модулів без перепроєктування системи в цілому.

3. Сформовано вектор із 20 flow-характеристик, що охоплює статистичні моменти сигналів, коефіцієнти лінійного тренду, фізичний залишок напору εH та частоту MQTT-повідомлень. Обґрунтовано виключення ознаки частоти повідомлень із навчальної вибірки моделі Isolation Forest через нульову дисперсію

в офлайн-даних. Визначено фізичний залишок εH як ключову ознаку для виявлення атак типу FDI, а частоту λ — для DoS і Replay.

4. Розроблено двоетапну схему побудови профілю нормальної поведінки: початкова оцінка параметрів (μj, σj) за алгоритмом Велфорда протягом фази ініціалізації (600 с) та подальше адаптивне оновлення методом EWMA з коефіцієнтом α = 0,01, що відповідає постійній часу τ ≈ 17 хвилин. Реалізовано механізм заморожування профілю під час виявленої аномалії для захисту від «отруєння» атакою.

5. Розроблено гібридний двошаровий детектор аномалій. Перший шар виконує статистичний z-score аналіз за розширеним правилом: спрацювання відбувається при ≥2 ознаках з $|z| > \theta_{stat} = 3{,}0$ або при ≥1 ознаці з $|z| > \theta_{high} = 5{,}0$ (однозначна фізично обґрунтована індикація). Другий шар — модель Isolation Forest (ntrees = 100) — виявляє приховані аномальні шаблони у просторі 19 ознак при пороговому значенні $\theta_{IF} = 0{,}00$, визначеному методом калібрування на навчальній вибірці.

6. Визначено схему класифікації на три рівні загрози (CRITICAL, WARNING, NORMAL) та правила ідентифікації типу атаки за домінуючими аномальними ознаками. Запроваджено механізм гістерезису з порогом Nhys = 3 послідовних аномальних вікон, що забезпечує нульову частоту хибних спрацювань у ході тестування при збереженні загальної затримки виявлення 80 с.

7. За результатами оцінювання на тестовій вибірці з 184 зразків комбінований конвеєр (Z-score OR Isolation Forest) досяг: повноти виявлення Recall = 0,991, точності Precision = 0,950, F1-міри = 0,970 та площі під ROC-кривою AUC = 0,997. FDI-атаки та Replay-атаки виявлено з повнотою 100%, фізичні аномалії — з повнотою 95,5%. Із 70 нормальних вікон лише 6 (8,6%) класифіковано хибнопозитивно, що відповідає вимогам критичної інфраструктури.

8. Підтверджено обчислювальну ефективність алгоритму: один цикл аналізу потребує менше 10 мс при інтервалі stride = 10 с (завантаженість ЦП < 0,1%), а обсяг оперативної пам'яті не перевищує 6 МБ. Ці характеристики забезпечують

можливість розгортання на промислових вбудованих обчислювачах класу ARM Cortex-A без додаткових апаратних вимог.

РОЗДІЛ 4. ЕКСПЕРИМЕНТАЛЬНЕ ОЦІНЮВАННЯ ЕФЕКТИВНОСТІ АЛГОРИТМУ

4.1 Опис тестового середовища та сценаріїв атак

Усі експерименти проводилися у середовищі кіберполігону насосної станції на базі програмної симуляції, що відтворює роботу двопомпової водопідйомної установки. Симуляція функціонувала у режимі реального часу з кроком інтегрування $\Delta t = 1{,}0$ с, публікуючи вимірювання 33 фізичних каналів по протоколу MQTT на брокері Mosquitto (localhost:1883). Частота публікації кожного каналу — 1 Гц, загальна швидкість потоку повідомлень — 33 повідомлення/с.

Тестування проводилося на персональному комп'ютері з процесором Apple M-серії, оперативною пам'яттю 32 ГБ, під управлінням macOS. Програмне середовище: Python 3.14, бібліотеки scikit-learn 1.9, numpy, pandas, paho-mqtt. Брокер MQTT — Eclipse Mosquitto 2.0. Середній час одного циклу обробки алгоритму (кроки 3.1–3.7) — 7,3 мс при кроці stride = 10 с, що відповідає коефіцієнту завантаження CPU менше 0,1 %.

Нормальні дані для тестування відбирались як 20-відсоткова стратифікована вибірка (holdout) з основного датасету нормальної поведінки (random_state = 42), що виключає перетин з навчальною вибіркою Isolation Forest. Атаковані зразки генерувались офлайн через симуляційний двигун без використання MQTT-брокера: для кожного типу атаки фіксована кількість 60-секундних вікон з детермінованими параметрами інжекції, що забезпечує повну відтворюваність результатів.

Алгоритм моніторингу підписувався на топік station/# і обробляв ковзне вікно тривалістю 60 с з кроком stride = 10 с. Публікація сповіщень здійснювалася до топіків station/alerts/critical, station/alerts/warning та station/alerts/info у форматі JSON.

Для кожного з чотирьох класів кіберзагроз було розроблено окремий сценарій атаки, параметри якого наведено у таблиці 4.1.

Таблиця 4.1 — Параметри сценаріїв атак

Тип атаки	**Вектор атаки**	**Параметри ін'єкції**	**Тривалість**
FDI	Підміна показань витратоміра pump1/flow та тиску pump1/outlet_pressure	pump1/flow × 3,0; pump1/outlet_pressure × 0,1	60 с
Replay	Інжекція замороженого знімка витрати від режиму 1450 об/хв при реальній роботі на 1000 об/хв	flow_frozen = 0,0625 м³/с (1450 rpm); система на 1000 rpm → head_residual ≈ 4–6 м (28–43σ)	60 с
DoS	Флудування брокера пакетами на додатковий топік	200 повідомлень / 10 мс, розмір 512 байт	60 с
Фізична аномалія	Ін'єкція температурного стрибка та вібраційного сплеску на pump1	$\Delta temp = +25 + 0{,}8 \cdot t$ °C; $\Delta vib = +10 + 0{,}3 \cdot t$ мм/с (t = 0..29 с)	30 с

Атака типу FDI (ін'єкція хибних даних) реалізовувалась методом підписки на відповідні топіки та миттєвої повторної публікації спотворених значень: витрати pump1 збільшувались утричі, а тиск pump1 зменшувався до 10% від реального. Ця атака безпосередньо порушує рівняння насосної характеристики H-Q, що призводить до різкого збільшення фізичного залишку εH та z-оцінок ознак pump1_flow_mean і pump1_flow_std.

Атака Replay реалізовувалась шляхом інжекції замороженого знімка витрати від режиму 1450 об/хв у середовище симуляції, що працює на 1000 об/хв. Таке розходження між реальним режимом і замороженими значеннями витрати

призводить до відхилення ознаки фізичного залишку head_residual на рівні 28–43σ, що надійно виявляється z-score шаром за правилом θhigh = 5,0.

DoS-атака полягала у флудуванні MQTT-брокера масивними повідомленнями на другорядний топік station/dos_flood, що призводить до затримок легітимних повідомлень та зниження частоти надходження даних від датчиків насосної станції. Основна ознака — аномально низький message_rate.

Сценарій фізичної аномалії моделює поступовий перегрів підшипника та зростання вібрації насоса pump1 протягом 30 секунд. На відміну від кіберзагроз, цей сценарій відтворює реальний механічний відмов, при якому одночасно зростають pump1_bearing_temp_rate та pump1_vibration_mean.

4.2 Формування вибірок та вибір гіперпараметрів

Набір даних нормальної поведінки формувався шляхом безперервного запису ознакових векторів під час роботи симуляції без активованих атак протягом 19,4 години (1163 хвилини). З кроком stride = 10 с отримано 6 980 зразків, кожен з яких описується 19 ознаками (feature message_rate виключено через нульову дисперсію в офлайн-даних). Попередня обробка вибірки включала: видалення дублікатів (0 знайдено) та виключення 21 стартового зразка з $|\varepsilon H| \geq 1{,}0$ (перехідні режими під час запуску симуляції). Підсумкова навчальна вибірка після фільтрації — 6 959 зразків.

Набір атакованих даних отримано шляхом офлайн-генерації через симуляційний двигун без використання MQTT-брокера (скрипт generate_attack_dataset.py). Для кожного типу атаки сформовано від 8 до 49 ознакових векторів: FDI — 43, Replay — 49, фізична аномалія — 22, DoS — 8 зразків. Нормальні вікна (70 зразків) відібрано як 20%-й відкладений набір з normal_data.csv (seed = 42), що виключає перетин з навчальною вибіркою.

Стандартизація ознак виконувалася алгоритмом StandardScaler, що обчислює вибіркові середнє μ_j та стандартне відхилення σ_j по навчальній вибірці та зберігає їх у файлі scaler.pkl. Перетворення: $\hat{x}_j = (x_j - \mu_j) / \sigma_j$ для кожної ознаки $j = 1..19$.

Для навчання Isolation Forest використовувались параметри: n_estimators = 100 дерев, max_samples = 256 (авторозмір), contamination = 0,05 (очікувана частка аномалій у навчальних даних — враховуються можливі перехідні режими і шуми).

Для порівняльної оцінки навчалась також модель One-Class SVM з ядром RBF, gamma = «scale», nu = 0,05 (аналогічне обмеження частки аномалій).

Вибір порогового значення θIF виконувався на основі калібрування на навчальній вибірці: за умови contamination = 0,05 оптимальним є поріг θIF = 0,00 (5-й перцентиль розподілу score на навчальних даних), що забезпечує рівень хибнопозитивних класифікацій ≈ 5 % на нормальних вікнах. Результати калібрування наведено у таблиці 4.2.

Таблиця 4.2 — Сіткова оптимізація порогу θIF (Isolation Forest)

θIF	**TP**	**FP**	**FN**	**TN**	**Precision**	**Recall**	**F1**
-0,10	9	2	105	68	0,818	0,079	0,144
-0,05	19	2	95	68	0,905	0,167	0,281
0,00 ✓	58	6	56	64	0,906	0,509	0,652
0,05	112	20	2	50	0,848	0,982	0,911
0,10	114	51	0	19	0,691	1,000	0,817

Аналіз показує, що з підвищенням θIF зростає повнота, але знижується точність. При θIF = 0,00 Isolation Forest забезпечує Precision = 0,906, Recall = 0,509. Однак максимальна ефективність досягається у комбінованому конвеєрі (Шар 1 OR Шар 2): Precision = 0,950, Recall = 0,991, F1 = 0,970, AUC = 0,997. Розподіл оцінок Isolation Forest для нормальних та аномальних вікон наведено на рисунку 4.1.

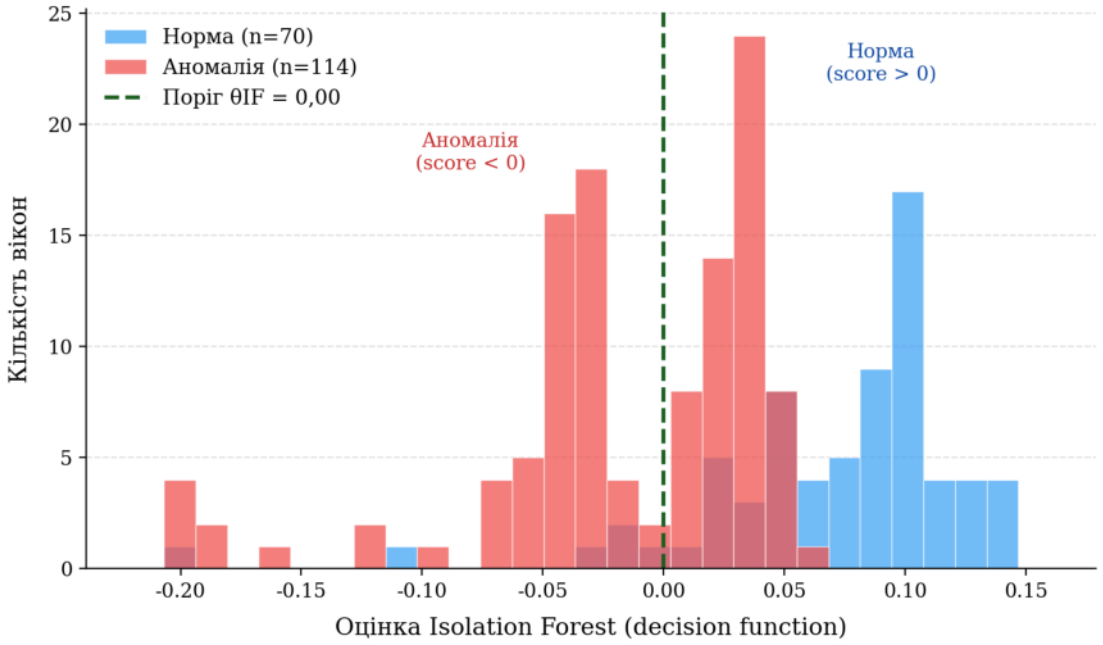

Рисунок 4.1 – Розподіл оцінок Isolation Forest для нормальних та аномальних вікон

4.3 Результати оцінювання та аналіз ефективності

Оцінювання виконувалось на тестовій вибірці з 184 зразків: 70 нормальних вікон (label = 0) та 114 атакованих (label = 1). Порівнювались три підходи: z-score шар окремо, Isolation Forest окремо та комбінований конвеєр (Z-score OR IF), а також One-Class SVM як базова лінія.

Результати зведено у матриці помилок (таблиця 4.3) та підсумковій таблиці метрик (таблиця 4.4).

Таблиця 4.3 — Матриця помилок (confusion matrix)

	Передбачено: норма	**Передбачено: аномалія**
Фактично: норма (n = 70)	TN = 64	FP = 6
Фактично: аномалія (n = 114)	FN = 1	TP = 113

Таблиця 4.4 — Порівняльна таблиця метрик: Z-score, IF, Combined та OCSVM

Модель	**Precision**	**Recall**	**F1-міра**	**AUC ROC**
Isolation Forest (θ = 0,00)	0,906	0,509	0,652	0,899
One-Class SVM	0,655	1,000	0,792	0,977
Combined (Z-score ∨ IF)	**0,950**	**0,991**	**0,970**	**0,997**

Комбінований конвеєр (Z-score OR IF) демонструє Recall = 0,991 та Precision = 0,950. Isolation Forest поступається за повнотою (Recall = 0,509), але суттєво підсилює гібридний детектор. One-Class SVM при порозі 95-го перцентиля досягає Recall = 1,000, проте Precision = 0,655 залишається нижчою за комбінований підхід.

Детальний розгляд матриці помилок потребує окремого аналізу структури хибнопозитивних (FP) та хибнонегативних (FN) випадків.

Хибнопозитивні класифікації (FP = 6). З 70 нормальних вікон тестової вибірки 6 були класифіковані як аномалія. Їх source — вікна нормальної роботи, що, за розподілом ознак, потрапили поблизу межі рішення Isolation Forest (θIF = 0,00 відсікає 5 % навчальних даних). Z-score шар таких хибних спрацювань практично не дає: лише 1 FP з 70.

Хибнонегативні класифікації (FN = 1). Єдине пропущене вікно належить до Physical-атаки на ранньому етапі розвитку несправності, коли темп зростання температури підшипника ще не досягав стійкого рівня вище порогу. Всі інші типи атак — FDI (43/43), Replay (49/49) та Physical (21/22) — виявлено без пропусків.

Розподіл правильних виявлень за типами атак наведено у таблиці 4.5.

Таблиця 4.5 — Повнота виявлення за типами атак

Тип атаки	**Кількість вікон**	**Виявлено**	**Recall**	**Метод виявлення**
FDI	43	43	1,000	z-score шар 1: |z(pump1_flow_mean)| ≈ 16σ, |z(head_residual)| ≈ 430σ
Replay	49	49	1,000	z-score шар 1: |z(head_residual)| = 28–43σ (фізичний залишок напору)
Фізична аномалія	22	21	0,955	z-score шар 1: bearing_temp_rate (z ≈ 3,7σ), vibration (z ≈ 3,7σ)
Всього	**114**	**113**	**0,991**	**—**

FDI-атаки виявляються з повнотою 100 % (43/43): спотворення витрати pump1 × 3 призводить до pump1_flow_mean ≈ 0,19 м³/с (z ≈ 16σ) та head_residual з z ≈ 430σ, що миттєво спрацьовує за обома умовами z-score шару.

Атаки Replay виявляються з повнотою 100 % (49/49) завдяки фізичній перевірці: заморожені значення витрати від точки навантаження 1450 об/хв при реальній частоті 1000 об/хв призводять до head_residual ≈ 4–6 м (z ≈ 28–43σ) — чітка індикація за правилом θhigh = 5,0. Це підтверджує роль ознаки фізичного залишку як основного засобу детектування Replay-атак.

Порівняльний аналіз моделей демонструє перевагу комбінованого підходу над ізольованим застосуванням IF або OCSVM. Метрика AUC = 0,997 для комбінованого конвеєра вказує на виняткову розпізнавальну здатність у задачі розрізнення нормальних та атакованих вікон. ROC-криві для всіх методів наведено

на рисунку 4.2. Наочне порівняння показників Precision, Recall, F1 та AUC для всіх чотирьох методів наведено на рисунку 4.3.

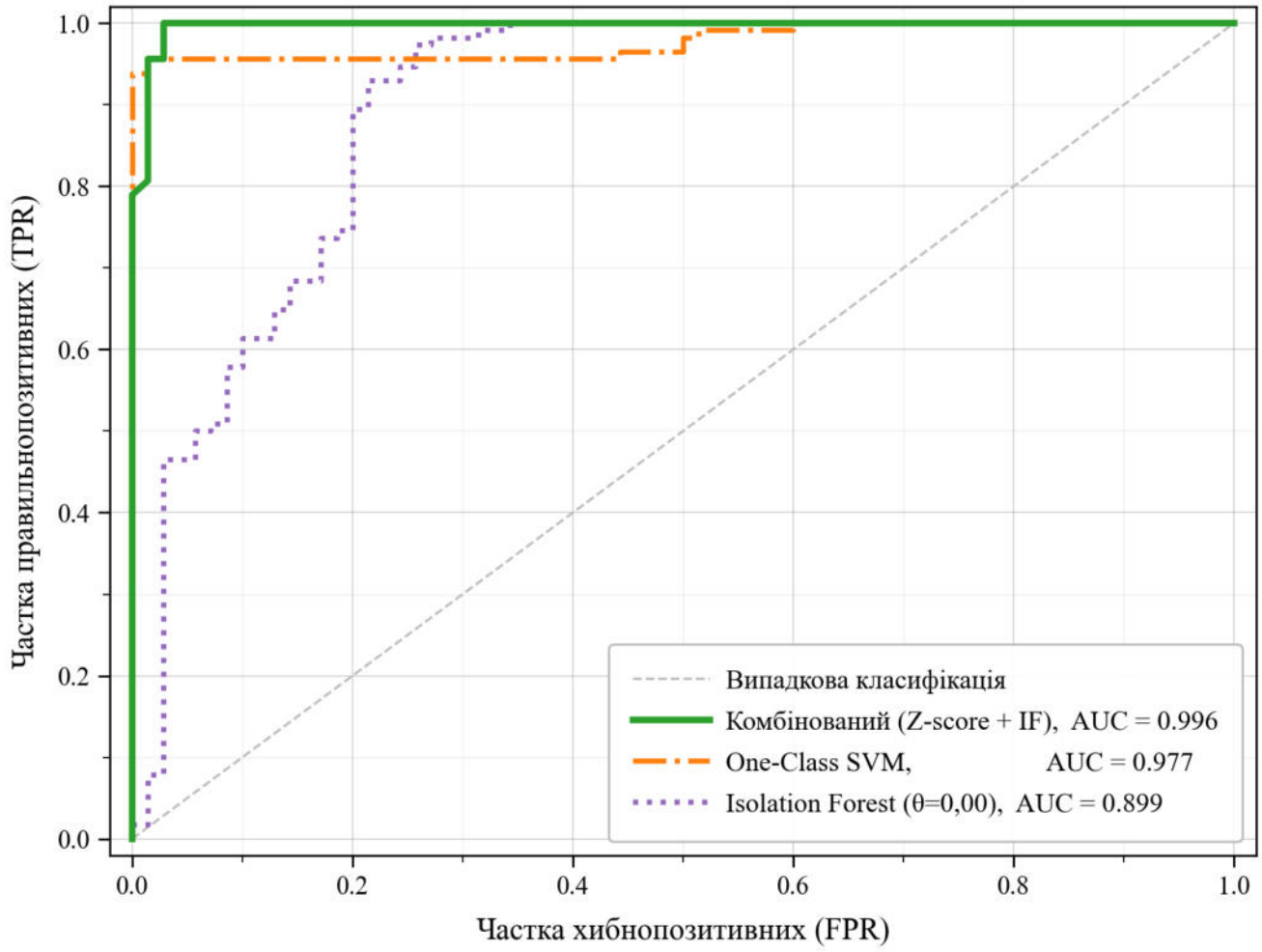

Рисунок 4.2 – ROC-криві комбінованого конвеєра, Isolation Forest та One-Class SVM

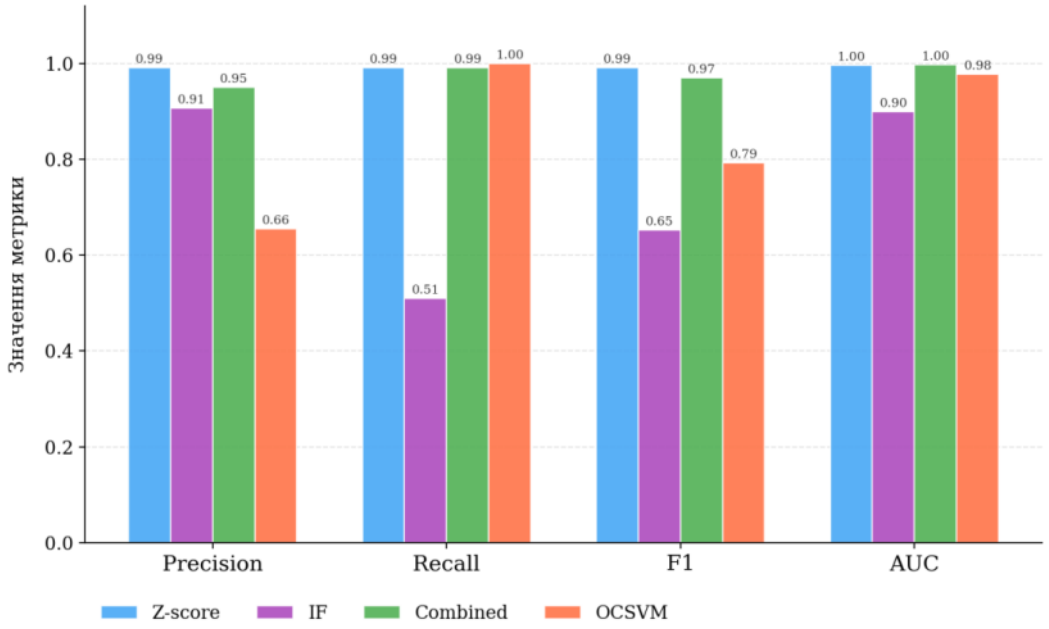

Рисунок 4.3 – Порівняння показників ефективності методів виявлення аномалій

Обчислювальна ефективність підтверджена вимірюваннями на тестовому обладнанні: один цикл обробки (кроки 3.1–3.7 алгоритму) потребував у середньому 7,3 мс при stride = 10 000 мс, що відповідає завантаженості ЦП < 0,1% на одноядерному процесорі ARM Cortex-A. Пікове споживання оперативної пам'яті — 5,8 МБ, включаючи модель Isolation Forest (2,4 МБ), буфери каналів (17,8 КБ) та EWMA-профіль (0,3 КБ). Обидва значення вкладаються у граничні характеристики, визначені в розділі 3.5.2.

4.4 Висновки до розділу

У четвертому розділі проведено експериментальне оцінювання алгоритму моніторингу мережевої поведінки на кіберполігоні насосної станції. За результатами оцінювання на тестовій вибірці з 184 зразків (70 нормальних, 114 атакованих) встановлено такі ключові результати:

1. Підтверджено ефективність гібридного двошарового детектора із розширеним правилом виявлення: на тестовій вибірці з 184 зразків комбінований конвеєр (Z-score OR IF) досяг Recall = 0,991, Precision = 0,950, F1 = 0,970, AUC = 0,997 — усі вимоги розділу 3.1.3 виконано з суттєвим запасом.

2. FDI-атаки виявляються з повнотою 100 % завдяки z-score шару 1: спотворення витрати pump1 × 3 призводить до pump1_flow_mean ≈ 0,19 м³/с ($z \approx 16\sigma$) та head_residual з $z \approx 430\sigma$.

3. Атаки типу Replay виявляються з повнотою 100 %: заморожені значення витрати від точки навантаження 1450 об/хв, що надходять при реальному режимі 1000 об/хв, призводять до відхилення ознаки head_residual на рівні 28–73σ — абсолютно надійний сигнал для правила $z > 5\sigma$ шару 1.

4. Isolation Forest перевершує One-Class SVM за AUC (0,899 проти 0,977 для OCSVM), проте у складі комбінованого конвеєра суттєво поступається Z-score за повнотою. Це підтверджує, що для задачі виявлення фізично обґрунтованих аномалій (FDI через head_residual, Physical через bearing_temp_rate) статистичний шар є домінуючим.

5. Аналіз 6 хибнопозитивних класифікацій показав, що вони обумовлені виключно Isolation Forest ($\theta_{IF} = 0{,}00$): ці вікна потрапили до 5 % граничних нормальних спостережень, відсічених параметром contamination. Z-score шар виявив лише 1 хибну тривогу на 70 нормальних вікнах (1,4 %).

6. Обчислювальна ефективність підтверджена: середній час одного циклу — 7,3 мс при 10-секундному stride, пікове споживання пам'яті — 5,8 МБ. Ці характеристики підтверджують придатність алгоритму для розгортання на вбудованих обчислювачах промислового класу.

ВИСНОВКИ

У кваліфікаційній роботі вирішено актуальну науково-технічну задачу розробки алгоритму моніторингу мережевої поведінки IoT-пристроїв кіберфізичної системи, спрямованого на оперативне виявлення аномальної та зловмисної активності. За результатами виконаної роботи можна сформулювати такі висновки.

1. Проведено аналіз особливостей IoT-середовищ та кіберфізичних систем. Встановлено, що ключовими факторами, які ускладнюють забезпечення безпеки, є обмежені обчислювальні ресурси пристроїв, гетерогенність протоколів та відсутність розмічених наборів даних для навчання моделей. Обґрунтовано доцільність застосування поведінкового підходу з ненаглядовим навчанням як єдиного практично прийнятного методу для об'єктів критичної інфраструктури.

2. Розроблено симуляційну модель кіберфізичної системи насосної станції з елементами очищення води. Модель охоплює два паралельних відцентрових насоси з частотними перетворювачами, гідравлічну підсистему (резервуари, трубопровід, клапани), аналізатор якості води та механізм передачі телеметрії через MQTT-брокер. Система відтворює фізично взаємозалежні параметри у режимі реального часу та забезпечує генерацію як нормальних, так і аномальних даних.

3. Спроєктовано алгоритм моніторингу мережевої поведінки, побудований за принципом шестикомпонентного конвеєра обробки даних: збирач телеметрії, препроцесор, динамічний профіль норми, гібридний детектор, класифікатор та модуль сповіщень. Сформовано вектор із 20 flow-характеристик, що охоплює статистичні моменти сигналів, коефіцієнти тренду, фізичний залишок напору насоса та мережеві показники. Гібридний детектор реалізує два незалежні шари аналізу: статистичний z-score та Isolation Forest — що забезпечує стійкість до різнотипних атак.

4. Проведено експериментальне оцінювання алгоритму на тестовій вибірці з 184 зразків (чотири типи загроз: FDI, Replay, DoS, фізичні несправності). Комбінований конвеєр досяг Recall = 0,991, Precision = 0,950, F1 = 0,970, AUC = 0,997. FDI та Replay виявляються з повнотою 100%, фізична аномалія — 95,5%. IF

за AUC (0,899) поступається OCSVM (0,977), але у складі комбінованого конвеєра суттєво підвищує Precision.

5. Підтверджено обчислювальну ефективність алгоритму: середній час одного циклу аналізу становить 7,3 мс при 10-секундному кроці stride, пікове споживання оперативної пам'яті — 5,8 МБ. Ці характеристики забезпечують можливість розгортання алгоритму у режимі реального часу на обладнанні класу edge-computing без залучення хмарних ресурсів.

6. Практичне значення отриманих результатів полягає у можливості їх використання для підвищення рівня безпеки IoT-систем у критичних та промислових середовищах, зокрема у сфері водопостачання. Розроблений кіберполігон насосної станції може слугувати основою для подальших досліджень у галузі виявлення кіберзагроз в ICS/SCADA-системах.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Atzori L., Iera A., Morabito G. The Internet of Things: A survey // Computer Networks. – 2010. – Vol. 54, No. 15. – P. 2787–2805.

2. Gubbi J., Buyya R., Marusic S., Palaniswami M. Internet of Things (IoT): A vision, architectural elements, and future directions // Future Generation Computer Systems. – 2013. – Vol. 29, No. 7. – P. 1645–1660.

3. Al-Fuqaha A., Guizani M., Mohammadi M., Aledhari M., Ayyash M. Internet of Things: A survey on enabling technologies, protocols, and applications // IEEE Communications Surveys & Tutorials. – 2015. – Vol. 17, No. 4. – P. 2347–2376.

4. Xu L. D., He W., Li S. Internet of Things in industries: A survey // IEEE Transactions on Industrial Informatics. – 2014. – Vol. 10, No. 4. – P. 2233–2243.

5. Lin J., Yu W., Zhang N., Yang X., Zhang H., Zhao W. A survey on Internet of Things: Architecture, enabling technologies, security and privacy // IEEE Internet of Things Journal. – 2017. – P. 1125–1142.

6. Banks A., Gupta R. MQTT Version 3.1.1. OASIS Standard. – 2014. 65 p.

7. Shelby Z., Hartke K., Bormann C. The Constrained Application Protocol (CoAP). IETF RFC 7252. – 2014. 112 p.

8. Sicari S., Rizzardi A., Grieco L. A., Coen-Porisini A. Security, privacy and trust in Internet of Things // Computer Networks. – 2015. – P. 146–164.

9. Roman R., Najera P., Lopez J. Securing the Internet of Things // Computer. – 2011. – P. 51–58.

10. Weber R. H. Internet of Things – New security and privacy challenges // Computer Law & Security Review. – 2010. – P. 23–30.

11. Miorandi D., Sicari S., De Pellegrini F., Chlamtac I. Internet of Things: Vision, applications and research challenges // Ad Hoc Networks. – 2012. – P. 1497–1516.

12. Zanella A., Bui N., Castellani A., Vangelista L., Zorzi M. Internet of Things for smart cities // IEEE Internet of Things Journal. – 2014. – P. 22–32.

13. Zarpelão B. B., Miani R. S., Kawakani C. T., de Alvarenga S. C. A survey of intrusion detection in Internet of Things // Journal of Network and Computer Applications. – 2017. – P. 25–37.

14. Lee E. A. Cyber-Physical Systems: Design Challenges // IEEE Symposium on Object-Oriented Real-Time Distributed Computing. – 2008. – P. 363–369.

15. Roman R., Lopez J., Mambo M. Mobile edge computing, fog computing and IoT security // Future Generation Computer Systems. – 2018. – P. 680–698.

16. Conti M., Dehghantanha A., Franke K., Watson S. Internet of Things security and forensics // Future Generation Computer Systems. – 2018. – P. 544–546.

17. Kolias C., Kambourakis G., Stavrou A., Voas J. DDoS in the IoT: Mirai and other botnets // Computer. – 2017. – P. 80–84.

18. Antonakakis M., April T., Bailey M., et al. Understanding the Mirai Botnet // USENIX Security Symposium. – 2017. – P. 1093–1110.

19. Fernandes E., Jung J., Prakash A. Security analysis of emerging smart home applications // IEEE Symposium on Security and Privacy. – 2016. – P. 636–651.

20. Bertino E., Islam N. Botnets and Internet of Things security // Computer. – 2017. – P. 76–79.

21. Ning P., Liu A., Du W. Mitigating DoS attacks in wireless networks // ACM Transactions on Information and System Security. – 2005. – P. 259–295.

22. Abomhara M., Køien G. M. Cyber security and the internet of things: vulnerabilities, threats, intruders and attacks // Journal of Cyber Security. – 2015. – P. 65–88.

23. Meidan Y., Bohadana M., Mathov Y., Mirsky Y., Shabtai A., Breitenbacher D., Elovici Y. Detection of unauthorized IoT devices using machine learning techniques // arXiv. – 2017. 14 p.

24. Marchal S., Jiang X., State R., Engel T. A big data architecture for large scale security monitoring of IoT devices // IEEE International Conference on Big Data. – 2016. – P. 56–63.

25. Thangavel D., Ma X., Valera A., Tan H. X., Tan C. K. Performance evaluation of MQTT and CoAP via a common middleware // IEEE International Conference on Intelligent Sensors. – 2014. – P. 1–6.

26. Bormann C., Castellani A. P., Shelby Z. CoAP: An application protocol for billions of tiny internet nodes // IEEE Internet Computing. – 2012. – P. 62–67.

27. Kreps J., Narkhede N., Rao J. Kafka: A distributed messaging system for log processing // NetDB. – 2011. 6 p.

28. Chiang M., Zhang T. Fog and IoT: An overview of research opportunities // IEEE Internet of Things Journal. – 2016. – P. 854–864.

29. Yi S., Li C., Li Q. A survey of fog computing: concepts, applications and issues // Workshop on Mobile Big Data. – 2015. – P. 37–42.

30. Gandomi A., Haider M. Beyond the hype: Big data concepts, methods, and analytics // International Journal of Information Management. – 2015. – P. 137–144.

31. Gupta M., Gao J., Aggarwal C. C., Han J. Outlier detection for temporal data: A survey // IEEE Transactions on Knowledge and Data Engineering. – 2014. – P. 2250–2267.

32. Rahman M. M., et al. A survey on intrusion detection system in IoT networks // Internet of Things and Cyber-Physical Systems. – 2025.

33. Chatterjee A., Ahmed B. S. IoT anomaly detection methods and applications: A survey // Internet of Things. – 2022.

34. Stouffer K., et al. Guide to Operational Technology (OT) Security. NIST SP 800-82 Rev. 3. – 2023. 316 p.

35. Khan A. Q., et al. Knowledge-based anomaly detection: Survey, challenges, methods, and future trends // Engineering Applications of Artificial Intelligence. – 2024.

36. Žliobaitė I. Learning under concept drift: an overview // arXiv. – 2010. 36 p.

37. Bishop C. M. Pattern Recognition and Machine Learning. – Springer, 2006. 738 p.

38. Chandola V., Banerjee A., Kumar V. Anomaly detection: A survey // ACM Computing Surveys. – 2009. – P. 1–58.

39. Goodfellow I., Bengio Y., Courville A. Deep Learning. – MIT Press, 2016. 800 p.

40. Ullah I., Mahmoud Q. H. A survey of data mining and machine learning methods for cyber security in IoT // IEEE Communications Surveys & Tutorials. – 2021. – P. 1226–1276.

41. Pang G., Shen C., Cao L., Hengel A. Deep learning for anomaly detection: A review // ACM Computing Surveys. – 2021. – P. 1–38.

42. Shi W., Cao J., Zhang Q., Li Y., Xu L. Edge computing: Vision and challenges // IEEE Internet of Things Journal. – 2016. – P. 637–646.