УДК 004.41
О. Карнаухов, здобувач третього (освітньо-наукового) рівня вищої освіти
Т. Лобур, здобувач третього (освітньо-наукового) рівня вищої освіти
С. Марценко, к.т.н., доц.
Тернопільський національний технічний університет ім. І. Пулюя, Україна

АВТОМАТИЗОВАНА СИСТЕМА КОНТРОЛЮ І ОБЛІКУ ЕНЕРГОРЕСУРСІВ УНІВЕРСИТЕТУ

O. Karnaukhov, PhD student
T. Lobur, PhD student
S. Martsenko, Ph.D., Assoc. Prof.
Ternopil Ivan Puluj National Technical University, Ukraine

AUTOMATED SYSTEM FOR MONITORING AND ACCOUNTING FOR THE UNIVERSITY'S ENERGY RESOURCES

У роботі описано технічну реалізацію автоматизованої системи контролю і обліку енергоресурсів (АСКОЕ), створеної в Тернопільському національному технічному університеті імені І. Пулюя. Система призначена для збору, передачі та централізованого управління даними про споживання газу і води. Її архітектура базується на принципах сервісно-орієнтованого підходу (SOA) з контейнеризацією компонентів у Docker, що забезпечує масштабованість, ізоляцію функціональних модулів та простоту розгортання в університетській інфраструктурі. Архітектурна структура наведена на рис. 1.

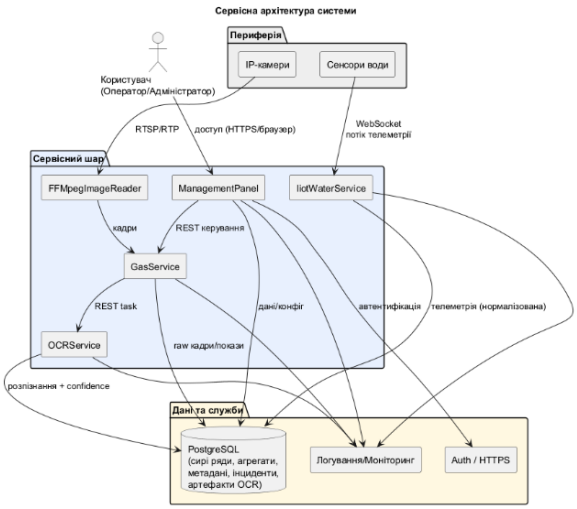

Рисунок 1. Архітектура системи

Сервісний шар системи включає ключові модулі GasService, IiotWaterService, OCRService, FFMpegImageReader та ManagementPanel, які взаємодіють між собою через REST API та WebSocket. Компоненти об’єднані єдиною базою даних PostgreSQL, що виконує роль центрального сховища. GasService відповідає за отримання кадрів з IP-камер за протоколом RTSP/RTP, які фіксують показники аналогових газових лічильників. Потік відео попередньо буферизується за допомогою FFMpegImageReader, що стабілізує частоту кадрів, забезпечує синхронізацію часових міток і усуває артефакти зображення. Після обробки кадри передаються до OCRService, який

використовує модель Microsoft TrOCR Large Printed для розпізнавання цифр лічильника. Результати містять значення показників, метадані та рівень впевненості (confidence), що разом з артефактами препроцесингу зберігаються у базі даних для подальшої перевірки оператором.

IiotWaterService збирає телеметрію від сенсорів води через WebSocket. Дані проходять валідацію (формат, діапазони, частота) та синхронізацію за часовими мітками, після чого зберігаються як часові ряди в БД і доступні для візуалізації в реальному часі.

ManagementPanel (Blazor) забезпечує операторське керування, налаштування вузлів та побудову інтерактивних дашбордів із графіками, таблицями й індикаторами. Система шаблонів дозволяє додавати нові компоненти візуалізації без втручання в ядро, а вбудований механізм – писати власні алгоритми обробки даних. Модуль Auth/HTTPS відповідає за автентифікацію та захищену комунікацію між сервісами. Логування/Моніторинг централізовано відстежує стан контейнерів, API-запити та затримки.

PostgreSQL зберігає часові ряди, метадані, інциденти, журнали OCR та конфігурації вузлів. Доступ через Entity Framework Core забезпечує незалежність від SQL і спрощує модифікацію структури даних.

Компоненти розгортаються в Docker-контейнерах (Docker Compose), що дозволяє ізолювати середовища та масштабувати сервіси. Інтеграція з периферією (IP-камери, сенсори) реалізована через RTSP/RTP, WebSocket і REST для сумісності з різним обладнанням.

Таким чином, система реалізує повнофункціональну архітектуру моніторингу двох типів енергоресурсів – газу та води – з можливістю розширення функціоналу. Вона підтримує інтерактивну побудову дашбордів, написання власних алгоритмів для аналізу даних, налаштування параметрів вузлів у реальному часі та централізоване керування сервісами через вебінтерфейс. Модульність, стандартизовані протоколи та контейнерна архітектура роблять систему технічно гнучкою, безпечною та придатною до інтеграції у ширші системи цифрової інфраструктури університету.

Літератруа

1. Li, M., Lv, T., Cui, L., Lu, Y., Florencio, D., Zhang, C., Li, Z. & Wei, F. (2021). *TrOCR: Transformer-based Optical Character Recognition with Pre-trained Models*. arXiv preprint arXiv:2109.10282. DOI: [10.48550/arXiv.2109.10282](https://doi.org/10.48550/arXiv.2109.10282)
2. FFmpeg Documentation [Електронний ресурс]. – Режим доступу: https://ffmpeg.org/documentation.html – Дата звернення: 21.10.2025.
3. PostgreSQL Documentation [Електронний ресурс]. – Режим доступу: https://www.postgresql.org/docs/ – Дата звернення: 21.10.2025.
4. ASP.NET Blazor Documentation [Електронний ресурс]. – Режим доступу: https://learn.microsoft.com/en-us/aspnet/core/blazor/ – Дата звернення: 21.10.2025.
5. Docker Documentation [Електронний ресурс]. – Режим доступу: https://docs.docker.com/ – Дата звернення: 21.10.2025.
6. WebSocket API Documentation [Електронний ресурс]. – Режим доступу: https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API – Дата звернення: 21.10.2025.

7. Real Time Streaming Protocol (RTSP). RFC 2326 [Електронний ресурс]. – Режим доступу: https://datatracker.ietf.org/doc/html/rfc2326 – Дата звернення: 21.10.2025.

УДК 004.056.5:004.738.5

В. Ю. Кащин, Т.А. Лечаченко доктор філософії
Тернопільський національний технічний університет імені Івана Пулюя

АНАЛІЗ ВРАЗЛИВОСТЕЙ HTTP REQUEST SMUGGLING ЗАСОБАМИ ДИФЕРЕНЦІАЛЬНОГО ФАЗЗИНГУ HTTP-ЗАПИТІВ

V. Kashchyn, T. Lechachenko Ph.D.

ANALYSIS OF HTTP REQUEST SMUGGLING VULNERABILITIES USING DIFFERENTIAL FUZZING OF HTTP REQUESTS

Вразливості типу HTTP Request Smuggling (HRS) залишаються однією з найнебезпечніших загроз для багаторівневих вебінфраструктур, у яких трафік послідовно проходить через балансувальники навантаження, проксі-сервери, брандмауери та бекенд-сервери [1]. Їхня природа полягає не у помилці окремого вебсервера, а в неузгодженості трактування структур HTTP/1.1-запитів різними компонентами ланцюга обробки. Конфлікт між заголовками Content-Length і Transfer-Encoding: chunked, а також неоднозначна обробка CRLF/LF-роздільників створюють умови, за яких один вузол вважає запит завершеним, тоді як інший продовжує читати потік як частину того самого або нового повідомлення. Це відкриває можливість інжекції прихованих запитів, обходу контролю доступу, отруєння кешу та реалізації DoS-сценаріїв.

Метою дослідження є експериментальне виявлення та класифікація уразливостей HTTP Request Smuggling шляхом застосування диференціального фазингу HTTP/1.1-запитів з використанням утиліти HTTP Request Smuggler у складі Burp Suite [2]. Для цього побудовано ізольований тестовий стенд типу Client → Nginx (reverse proxy) [3] → Apache HTTP Server (backend) [4], розгорнутий на віртуальних машинах під керуванням гіпервізора KVM [5] у середовищі Ubuntu Linux. Nginx налаштовано як зворотний проксі з акцентом на обробці конфліктних заголовків Content-Length і Transfer-Encoding, Apache виступає бекендом із власною логікою визначення меж тіла HTTP-повідомлень. У дослідженні сформовано репрезентативний набір HTTP/1.1-запитів із варіативними заголовками та роздільниками рядків: класичні CL- та TE-запити, конфліктні вектори CL.TE та TE.CL, дубльовані Content-Length із різними значеннями, подвійні Transfer-Encoding, некоректні або неповні заголовки, а також змішані варіанти CRLF/LF. Диференціальний фазинг реалізовано як багаторазове відтворення цих шаблонів через Burp Suite з HTTP Request Smuggler у напрямку Nginx → Apache з паралельною фіксацією артефактів. Для реєстрації результатів використано розширене логування access/error на обох серверах і перехоплення трафіку інструментом tshark у форматі pcap, що забезпечило повну картину обробки кожного запиту на мережевому й прикладному рівнях.

Отримані результати демонструють, що для ряду конфігурацій виникають стійкі парсингові розбіжності між Nginx і Apache. Для векторів CL.TE та TE.CL показано сценарії, за яких проксі орієнтується на Content-Length, а бекенд - на Transfer-Encoding: chunked, або навпаки, унаслідок чого в межах одного TCP-з’єднання з’являється «прихований» GET-запит, що фактично минає первинний рівень контролю. У випадку