3 Міжнародна науково-практична конференція, Innovative Approaches in Modern Science and Technology
Секція 11 - Інформаційні технології та Кібербезпека

ПРОЄКТУВАННЯ СИСТЕМИ ВЕБ-ЗАХИСТУ НА ОСНОВІ МАШИННОГО НАВЧАННЯ

Єфімов Михайло
здобувач вищої освіти магістерського рівня
Долгова Наталя
к.т.н, доцент
Кафедра кібербезпеки та інформаційних технологій
Харківський національний економічний університет імені Семена Кузнеця

Актуальність дослідження визначається стійкістю класичних веб-загроз (передусім SQL-ін’єкцій і XSS) на тлі переходу галузі до мікросервісів, SPA та багатоканальних API [1-2]. Розширення площі атаки та зростання різноманіття параметрів призводять до накопичення хибнопозитивних спрацьовувань і подовження часу перевірок у традиційних сканерах. Відповіддю на цей виклик є інтеграція машинного навчання (ML) не як допоміжного модуля, а як керуючого механізму, що дисциплінує послідовність дій, перерозподіляє обчислювальні ресурси та зменшує обсяг активних запитів без втрати повноти виявлення [3-4].

Запропонований підхід виходить із потреб трьох ключових категорій користувачів: розробників, фахівців із безпеки та викладачів. Для кожної з них критичними є висока точність при керованому рівні хибнопозитивів, прозора звітність, відтворюваність експериментів і можливість інтеграції в CI/CD-процеси. Визначені вимоги формують обмеження на архітектуру, вибір алгоритмів і організацію потоків даних, а також диктують необхідність версіонування артефактів (моделей, токенізаторів, конфігурацій), що гарантує стабільність результатів у часі [5].

Теоретичне підґрунтя проєкту спирається на порівняльний аналіз WAF/сигнатурних рішень, SAST/DAST і сучасних ML-підходів. Сигнатури ефективні для відомих загроз, однак слабо адаптуються до нових; статичний і динамічний аналіз забезпечують дисципліну коду та реалістичність перевірок, але часто «шумлять» і потребують тривалих прогонів; академічні ML-моделі демонструють гарні метрики на контрольних наборах, однак рідко змінюють реальну поведінку сканера без належної інтеграції у конвеєр. Звідси випливає принципова теза: ML має працювати як диспетчер навантаження, що визначає пріоритети дослідження поверхні атаки, скорочуючи обсяг активних перевірок і підвищуючи цінність кожного запиту [6].

Архітектура системи спроєктована як трирівневий організм зі сквозним потоком даних. На вхідному рівні акумулюються журнали веб-трафіку, результати краулінгу, історичні пейлоади атак і записи з відкритих реєстрів вразливостей; ці джерела утворюють корпус для навчання та контекстуалізації

майбутніх знахідок. Центральне ядро виконує ETL-перетворення, проводить ML-аналіз і керує фаззингом, забезпечуючи взаємодію між препроцесорами, моделями та засобами зберігання. На вихідному рівні реалізовано прикладну інфраструктуру з дашбордом, базою знахідок і механізмами генерації звітів, придатних для використання розробниками й аналітиками безпеки (рис.1).

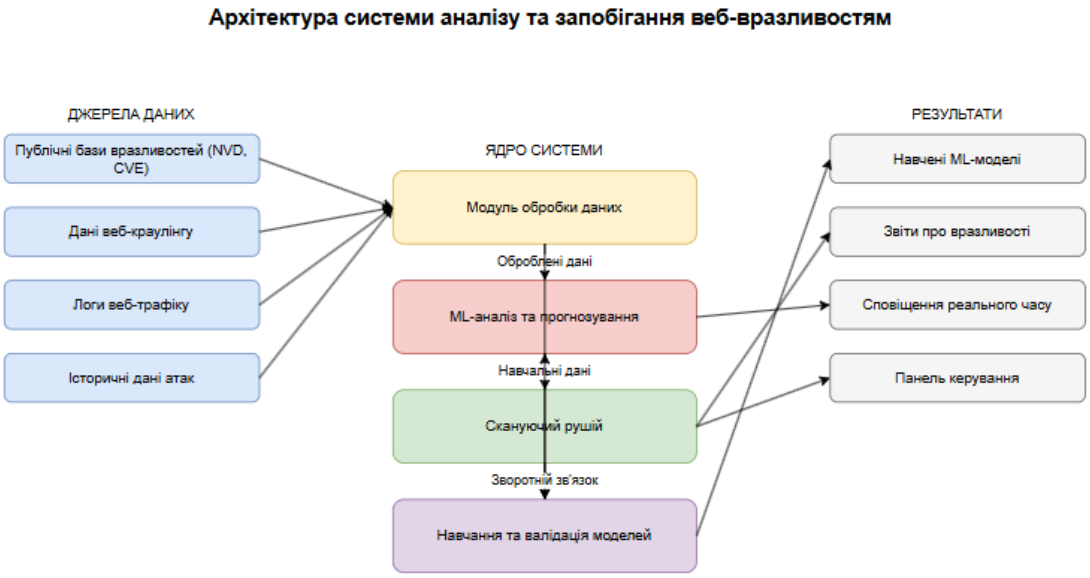

Рис. 1. Архітектура системи для аналізу та запобігання веб-вразливостям

Ключовим рішенням є подвійний конвеєр підготовки даних із подальшим ансамблюванням. «Класична» гілка використовує TF-IDF із символьними n-грамами та інженерію ознак (довжина параметра, частка спеціальних символів, наявність маркерів SQL/XSS, оцінка ентропії), що формує компактні вектори для Random Forest, лінійної SVM, XGBoost і Naive Bayes (рис.2).

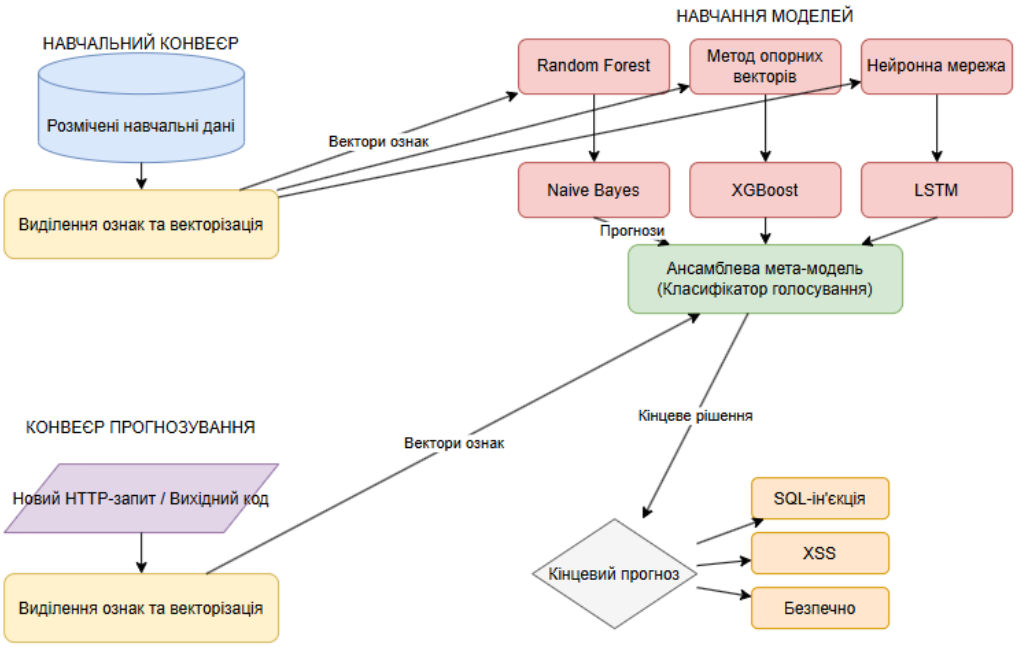

Рис. 2. Конвеєр моделі машинного навчання для виявлення вразливостей

«Нейромережева» гілка розглядає запити як символьні послідовності фіксованої довжини; LSTM у поєднанні з вбудовуваннями та регуляризацією вловлює контекстні та структурні патерни, які важко відобразити ручними ознаками. Над обома гілками працює метапередбачувач із «м'яким» голосуванням, що усереднює апостеріорні ймовірності й повертає єдину, стабілізовану оцінку ризику для кожного об'єкта аналізу.

Процес функціонування системи організовано як чотири взаємопов'язані фази. На етапі краулінгу формується карта поверхні атаки з урахуванням автентифікації та параметрів форм. Далі реалізується пасивний ML-аналіз, який ранжує URL/параметри без ін'єкцій і маркує перспективні напрямки для подальших дій. На фазі інтелектуального фаззингу великий пул пейлоадів передається на батч-скоринг ансамблем; у результаті до активної перевірки переходить лише вузька вибірка найризиковіших кандидатів. Завершальна фаза виконує багатопоточну активну валідацію з оновленням CSRF-токенів і диференційним порівнянням сторінок, формуючи доказову базу для звітів.

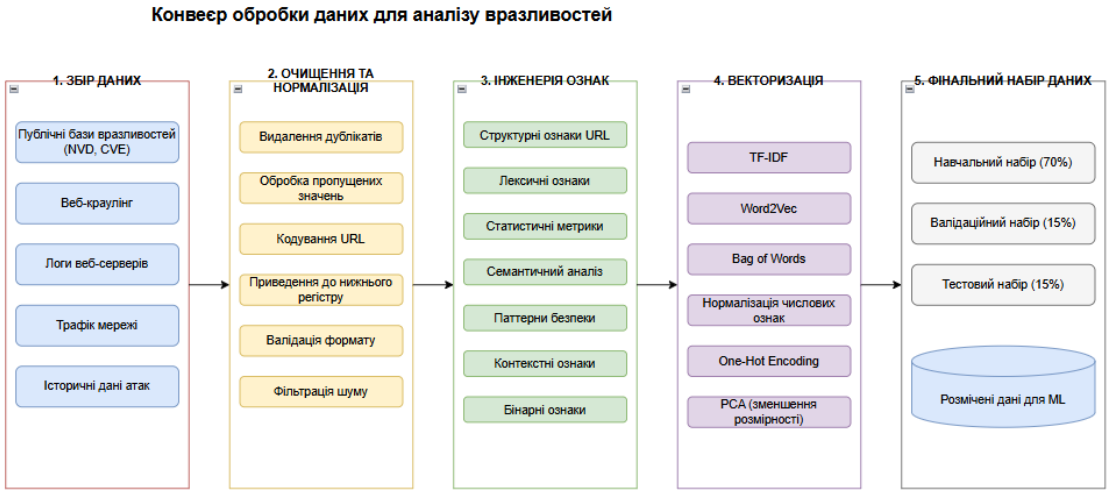

Рис. 3. Конвеєр обробки даних для аналізу вразливостей

Запропонована організація потоків забезпечує баланс між швидкістю та акуратністю. Пасивний аналіз виконує функцію раннього відсіву, зменшуючи ризик «шуму» на наступних етапах; ML-фаззинг замінює грубу силу кількості на цілеспрямованість, а активна валідація фіксує результати у вигляді відтворюваних доказів . Завдяки версіонуванню артефактів кожен прогін залишається зіставним із попередніми, що є критично важливим для оцінювання якості, регресійного тестування та поступового удосконалення моделей.

Окреслена архітектура адресує традиційні компроміси. Суперечність між локальною точністю класифікації та загальною пропускною здатністю конвеєра нівелюється за рахунок ансамблю та пакетної інференції. Конфлікт між вимогами до «обережності» щодо продуктивних середовищ і потребою в переконливих доказах долається завдяки перевазі пасивних етапів і чітким стоп-умовам на активних. Напруга між гнучкістю та керованістю послаблюється модульністю, контрактами між фазами та стандартизованими форматами артефактів.

Узагальнюючи, проєктування системи веб-захисту подається як побудова цілісної екосистеми прийняття рішень, де дані послідовно очищуються, осмислюються моделями та перевіряються рівно настільки, наскільки цього потребує ризик. Машинне навчання у такій екосистемі виконує роль керуючого елемента, що спрямовує увагу на найбільш перспективні ділянки, зменшуючи обсяг активних дій і підвищуючи доказовість звітів. Запропонована архітектура, подвійний конвеєр підготовки даних і чотирифазний процес утворюють надійну основу для подальшого розширення на інші класи вразливостей і для інтеграції у промислові режими з вимогами до відтворюваності, масштабованості та пильного контролю якості.

Список використаних джерел

1. OWASP Foundation. (2021). *OWASP Top 10: 2021*. Retrieved November 7, 2025, from https://owasp.org/Top10/
2. OWASP Cheat Sheet Series. (n.d.). *SQL Injection Prevention Cheat Sheet*. Retrieved November 1, 2025, from https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html
3. PortSwigger Ltd. (n.d.). *What is SQL injection?* PortSwigger Web Security Academy. Retrieved November 2, 2025, from https://portswigger.net/web-security/sql-injection
4. National Institute of Standards and Technology. (n.d.). *National Vulnerability Database (NVD)*. Retrieved November 1, 2025, from https://nvd.nist.gov/
5. Куперштейн, Л. М., Притула, А. В., & Маліновський, В. І. (2024). Аналіз технологій тестування на проникнення web-додатків. *Наукові праці Вінницького національного технічного університету*, (2). https://doi.org/10.31649/2307-5376-2024-2-45-53
6. Притула, А., & Куперштейн, Л. (2025). Аналіз підходів тестування на проникнення з використанням машинного навчання з підкріпленням. *Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка»*, *4*(28), 259–271. https://doi.org/10.28925/2663-4023.2025.28.789