УДК 004.056.53:004.42

О. В. Салієва
А. І. Білоус

ПОРІВНЯЛЬНИЙ АНАЛІЗ АЛГОРИТМІВ АВТОМАТИЗОВАНОГО ВИЯВЛЕННЯ ІН'ЄКЦІЙНИХ ВРАЗЛИВОСТЕЙ У СУЧАСНИХ ВЕБЗАСТОСУНКАХ

Вінницький національний технічний університет

Анотація

У роботі досліджується проблематика автоматизованого виявлення вразливостей вебзастосунків до ін'єкційних атак (SQL Injection, Cross-Site Scripting, OS Command Injection). У зв'язку зі стабільним домінуванням цих загроз у глобальних рейтингах безпеки, виникає критична потреба у вдосконаленні методів динамічного тестування безпеки (DAST). Проведено порівняльний аналіз сигнатурних та евристичних алгоритмів сканування, реалізованих у межах розробленої спеціалізованої програмної вебплатформи. Визначено переваги та недоліки кожного підходу за критеріями швидкодії та рівня хибних спрацювань. Обґрунтовано доцільність застосування гібридної моделі сканування для підвищення точності ідентифікації векторів атак.

Ключові слова: кібербезпека, тестування на проникнення, DAST, SQL Injection, XSS, Command Injection, сканер вразливостей, інформаційна безпека.

Abstract

The paper investigates the problem of automated vulnerability detection in web applications against injection attacks (SQL Injection, Cross-Site Scripting, OS Command Injection). Due to the consistent dominance of these threats in global security rankings, there is a critical need to improve Dynamic Application Security Testing (DAST) methods. A comparative analysis of signature-based and heuristic scanning algorithms, implemented within a developed specialized web platform, is conducted. The advantages and disadvantages of each approach are identified based on performance criteria and the rate of false positives. The feasibility of using a hybrid scanning model to increase the accuracy of attack vector identification is substantiated.

Keywords: cybersecurity, penetration testing, DAST, SQL Injection, XSS, Command Injection, vulnerability scanner, information security.

Вступ

Еволюція архітектури сучасних вебзастосунків та їхня глибока інтеграція з базами даних і системними оболонками супроводжується зростанням кількості критичних векторів кіберзагроз. Відповідно до консенсусу фахівців з кібербезпеки, зафіксованого у звітах OWASP Top 10, ін'єкційні атаки стабільно визнаються одними з найбільш критичних ризиків для вебсистем [1]. Успішна експлуатація таких вразливостей, як SQL Injection чи OS Command Injection, дозволяє зловмиснику порушити цілісність баз даних або здійснити несанкціоноване виконання команд на рівні операційної системи сервера [2, 3]. Водночас атаки класу Cross-Site Scripting (XSS) залишаються основним вектором компрометації клієнтських сесій [4].

Згідно з рекомендаціями Національного інституту стандартів і технологій США (NIST SP 800-115) та методологією OWASP Web Security Testing Guide (WSTG), ефективна стратегія захисту вимагає регулярного технічного тестування та валідації механізмів управління вхідними даними [5, 6]. Зважаючи на високу трудомісткість ручного тестування на проникнення, критичного значення набувають системи динамічного тестування безпеки, які аналізують застосунок у стані виконання [7]. Подібні автоматизовані інструменти працюють за методологією «чорного ящика», імітуючи поведінку зовнішнього зловмисника без необхідності доступу до вихідного коду. Впровадження таких рішень є невід'ємною складовою сучасних практик DevSecOps, оскільки дозволяє масштабувати процеси аудиту та виявляти логічні помилки на ранніх етапах розгортання. Метою даної роботи є порівняльний аналіз ефективності алгоритмів автоматизованого виявлення ін'єкційних вразливостей, імплементованих у розроблену спеціалізовану вебплатформу.

Результати дослідження

У процесі проєктування вебплатформи для виявлення вразливостей було програмно реалізовано та досліджено кілька базових алгоритмів автоматизованого сканування. Головним критерієм ефективності алгоритму DAST є баланс між повнотою покриття, швидкістю виконання операцій та мінімізацією рівня хибних спрацювань. Загалом алгоритми сканування класифікуються на сигнатурні та евристичні.

Сигнатурний алгоритм базується на відправленні заздалегідь сформованого тестового корисного навантаження та лексичному аналізі HTTP-відповіді сервера на наявність специфічних маркерів помилок. Наприклад, для детектування Error-based SQL Injection система ін'єктує спеціальні символи та використовує регулярні вирази для ідентифікації діагностичних повідомлень СУБД. Для виявлення Reflected XSS сканер відправляє унікальний буквено-цифровий ідентифікатор та перевіряє факт його повернення у структурі документа без належного HTML-екранування [4]. Основною перевагою цього методу є висока швидкість обробки мережевих запитів. Проте він демонструє низьку ефективність під час тестування систем, що приховують виведення помилок, або захищені міжмережевими екранами (Web Application Firewall, WAF).

Евристичний алгоритм фокусується на аналізі часових та структурних аномалій у реакції цільової системи на спеціально згенеровані запити. Найбільш показовим прикладом є виявлення Time-based Blind SQL Injection або сліпих ін'єкцій команд операційної системи (Blind OS Command Injection). Алгоритм платформи відправляє тестовий вектор із командою затримки виконання потоку. Факт наявності вразливості фіксується у випадку виявлення статистично значущої кореляції між заданим часом затримки та фактичним часом очікування HTTP-відповіді від сервера [2, 3]. Цей підхід дозволяє ідентифікувати критичні вектори навіть за умови відсутності видимого виведення даних, однак суттєво збільшує тривалість сеансу сканування.

Для системного відображення ефективності досліджуваних підходів їхні характеристики узагальнено в таблиці 1.

Таблиця 1 – Порівняльна характеристика алгоритмів автоматизованого виявлення ін'єкцій

Тип алгоритму	Вектор атаки	Принцип роботи	Переваги	Обмеження та недоліки
Сигнатурний	Error-based SQLi, Reflected XSS	Пошук синтаксичних помилок та маркерів у тілі HTTP-відповіді	Висока швидкодія, мінімальне навантаження на мережевий стек	Високий відсоток хибних негативних результатів, вразливість до WAF
Евристичний (часовий)	Blind SQLi, OS Command Injection	Аналіз часових затримок HTTP-відповідей на ін'єктовані команди	Висока точність виявлення прихованих вразливостей (сліпі атаки)	Знижена швидкість сканування, ризик хибних спрацювань через нестабільність мережі
Аналіз об'єктної моделі	DOM-based XSS, Stored XSS	Емуляція середовища виконання (браузера), синтаксичний аналіз DOM-дерева	Ефективність проти складних клієнтських атак та асиметричних ін'єкцій	Вимагає значних обчислювальних ресурсів для рендерингу сторінок

Результати аналізу свідчать, що використання ізольованих алгоритмів є недостатнім для забезпечення надійного аудиту безпеки. Оптимальним інженерним рішенням для сучасних DAST-рішень є впровадження гібридної моделі, за якої платформа ініціює перевірку із застосуванням швидких сигнатурних тестів, і лише у разі відсутності явних маркерів вразливості переходить до глибокого евристичного аналізу із використанням методів затримки часу [7].

Висновки

У результаті проведеного дослідження проаналізовано алгоритмічні підходи до автоматизованого виявлення вразливостей клієнтського та серверного рівнів у сучасних вебзастосунках. Встановлено, що сигнатурні алгоритми гарантують високу швидкість обробки цільових ресурсів, проте не здатні детектувати приховані вектори атак типу OS Command Injection та сліпих SQL-ін'єкцій. Евристичні методи, зі свого боку, забезпечують точну ідентифікацію вразливостей наосліп, але вимагають оптимізації обробки асинхронних мережевих запитів. Доведено, що інтеграція гібридного алгоритму в архітектуру розробленої вебплатформи дозволяє оптимізувати процес динамічного тестування, мінімізувати рівень хибних спрацювань та забезпечити відповідність процедур аудиту сучасним галузевим стандартам інформаційної безпеки.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. OWASP Top 10. The Open Worldwide Application Security Project (OWASP). URL: https://owasp.org/Top10/ (дата звернення: 09.03.2026).

2. SQL Injection. Web Security Academy. PortSwigger. URL: https://portswigger.net/web-security/sql-injection (дата звернення: 09.03.2026).

3. OS command injection. Web Security Academy. PortSwigger. URL: https://portswigger.net/web-security/os-command-injection (дата звернення: 09.03.2026).

4. Cross-site scripting (XSS). Web Security Academy. PortSwigger. URL: https://portswigger.net/web-security/cross-site-scripting (дата звернення: 09.03.2026).

5. Web Security Testing Guide (WSTG) v5.0. OWASP Foundation. URL: https://github.com/OWASP/wstg/tree/master/document (дата звернення: 10.03.2026).

6. Scarfone K., Souppaya M. Technical Guide to Information Security Testing and Assessment (NIST SP 800-115). National Institute of Standards and Technology. URL: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-115.pdf (дата звернення: 10.03.2026).

7. Dynamic Application Security Testing. OWASP DevSecOps Guideline. OWASP Foundation. URL: https://owasp.org/www-project-devsecops-guideline/latest/02b-Dynamic-Application-Security-Testing (дата звернення: 10.03.2026).

Білоус Артем Ігорович – студент групи 1КІТС-22Б, Факультет менеджменту та інформаційної безпеки, Вінницький національний технічний університет, м. Вінниця, e-mail: artembilous7@gmail.com

Науковий керівник: ***Салієва Ольга Володимирівна*** – доктор філософії (PhD) за спеціальністю 125 «Кібербезпека», доцент кафедри менеджменту та безпеки інформаційних систем, Вінницький національний технічний університет, Вінниця, e-mail: salieva8257@vntu.edu.ua

Bilous Artem I. – student of group 1KITS-22B, Faculty of Management and Information Security, Vinnytsia National Technical University, Vinnytsia, e-mail: artembilous7@gmail.com

Supervisor: ***Saliieva Olha V.*** – Doctor of Philosophy (PhD) in specialty 125 "Cybersecurity", Associate Professor of the Department of Management and Security of Information Systems, Vinnytsia National Technical University, Vinnytsia, email: salieva8257@ vntu.edu.ua