АВТОМАТИЗАЦІЯ ДИНАМІЧНОГО АНАЛІЗУ ЗАХИЩЕНОСТІ ВЕБ-ДОДАТКІВ У КОНВЕЄРІ БЕЗПЕРЕРВНОЇ ІНТЕГРАЦІЇ ЗА СТАНДАРТОМ OWASP WSTG

Шмагун В.І., Балагура Д.С.
Харківський Національний Університет Радіоелектроніки, Харків, Україна

Інтеграція засобів автоматизованого контролю безпеки в процеси безперервної інтеграції та доставки програмного забезпечення (CI/CD) є одним із ключових завдань підходу DevSecOps при захисті сучасних веб-орієнтованих систем. Значна частина критичних уразливостей веб-додатків, таких як порушення контролю доступу, помилки керування сесіями, серверна підробка запитів (SSRF) або некоректна конфігурація заголовків безпеки, проявляється виключно під час виконання програми та не може бути надійно виявлена статичними аналізаторами коду [1, 2]. Водночас класичне динамічне тестування за принципом «чорної скриньки» (DAST) розраховане на тривале ручне або напівавтоматичне сканування, що сповільнює випуск оновлень і ускладнює перевірку кожного релізу згідно з методологією OWASP Web Security Testing Guide (WSTG) [2].

Метою доповіді є розробка підходу до автоматизації динамічного аналізу захищеності веб-додатків у конвеєрі CI/CD за стандартом OWASP WSTG, а також визначення ефективних способів підвищення точності виявлення вразливостей під час автоматизованого сканування.

В доповіді наводяться результати дослідження щодо ефективності впровадження автоматизованих DAST-сканерів для перевірки веб-орієнтованих систем на етапі розгортання тестового середовища [2, 3]. Проведений аналіз показав, що головними перешкодами при повній автоматизації динамічних перевірок є неповне виявлення маршрутів односторінкових додатків (SPA) і програмних інтерфейсів (REST та GraphQL API), втрата стану автентифікованої сесії під час активного фазингу, а також надмірна тривалість повного циклу сканування [1, 3]. Підвищити результативність виявлення вразливостей в автоматизованому режимі можна, якщо структурувати процес динамічного тестування за категоріями стандарту OWASP WSTG, що можна реалізувати кількома способами:

1. Автоматизована побудова карти поверхні атаки на основі машиночитаних специфікацій та браузерного краулінгу. Замість пасивного перебору посилань конвеєр імпортує схеми OpenAPI/Swagger і GraphQL безпосередньо зі сховища коду, доповнюючи їх обходом клієнтського інтерфейсу через безголовий браузер (Headless Browser) для перехоплення динамічних DOM-подій та асинхронних AJAX-запитів відповідно до вимог розділу інформаційної розвідки WSTG-INFO [2].

2. Конфігурування захищеного контексту виконання та автоматичної підтримки автентифікованих сесій (WSTG-ATHN, WSTG-SESS). Для перевірки закритих функціональних модулів у проксі-сканер інтегруються сценарії отримання та автоматичного оновлення токенів доступу (JWT, OAuth

2.0) або сесійних ідентифікаторів, а для тестування рольової моделі розмежування прав (WSTG-ATHZ) виконується автоматичне повторення HTTP-запитів із підміною параметрів доступу, заголовків авторизації та ідентифікаторів об'єктів (IDOR) [1, 2].

3. Розділення динамічного аналізу на швидкий базовий та поглиблений асинхронний профілі в середовищі CI/CD. Під час кожного коміту виконується пасивний аналіз трафіку функціональних автотестів та швидка перевірка конфігурації безпеки (заголовки CSP, HSTS, CORS, атрибути кукі HttpOnly та Secure за категорією WSTG-CONF), тоді як ресурсоємний активний фазинг векторів ін'єкцій (SQLi, NoSQLi, XSS, OS Command Injection за категорією WSTG-INPV) запускається паралельно на ізольованому стенді для змінених модулів [2, 4].

4. Автоматична верифікація результатів сканування, дедуплікація звітів та блокування вразливих збірок (Quality Gate). Отримані звіти проксі-сканера OWASP ZAP у стандартному форматі SARIF передаються до централізованої системи управління вразливостями, де здійснюється фільтрація хибнопозитивних спрацювань за кодами та структурою відповідей сервера, зіставлення знахідок із результатами статичного аналізу коду та розрахунок балів критичності за метрикою CVSS; при виявленні підтверджених уразливостей високого рівня ризику подальше розгортання релізу автоматично блокується [3, 5].

Абсолютно безпечних веб-орієнтованих систем не існує, проте перехід від періодичних ручних перевірок до безперервного динамічного сканування в конвеєрі CI/CD дозволяє суттєво скоротити вікно вразливості програмного продукту.

Поєднання автоматичної інвентаризації API, контролю автентифікованих сесій та розділення профілів фазингу за методологією OWASP WSTG забезпечує глибоке охоплення бізнес-логіки веб-додатка, мінімізує кількість пропусків і дозволяє усувати критичні дефекти безпеки ще до введення системи в промислову експлуатацію.

Список літератури

1. OWASP Top 10: The Ten Most Critical Web Application Security Risks. – OWASP Foundation, 2021. URL: https://owasp.org/www-project-top-ten/

2. Meucci M., Muller A. OWASP Web Security Testing Guide (WSTG) v4.2. – OWASP Foundation, 2020. – 364 p.

3. Hsu T. Hands-On Security in DevOps: Continuous Security, Deployment, and Delivery with DevSecOps. – Birmingham: Packt Publishing, 2018. – 422 p.

4. Halde J. Basics of SQL Injection Analysis, Detection and Prevention. – LAP LAMBERT Academic Publishing, 2014. – 80 p.

5. Коваленко А. А., Кучук Г. А. Методи синтезу інформаційної та технічної структур системи управління об'єктом критичного застосування. Сучасні інформаційні системи. 2025. Т. 2, № 1. С. 22–27.