УДК 004.056.5

Максим ЧУХНІЙ, Надія Гавришків[2], Віктор ДЗЯДИК[2]

[1]Західноукраїнський національний університет
[2]Галицький фаховий коледж ім. В'ячеслава Чорновола

СУЧАСНІ МЕТОДИ ДОСЛІДЖЕННЯ БЕЗПЕКИ ВЕБ-ДОДАТКІВ

Вступ. Веб-додатки є ключовими інструментами для взаємодії між користувачами та інформаційними системами, що робить їх привабливою ціллю для кіберзлочинців. Забезпечення безпеки таких додатків стало критично важливим завданням для розробників та фахівців з кіберзахисту. Сучасні методи дослідження безпеки включають автоматизоване сканування вразливостей, пентестинг, аналіз коду та поведінкові моделі. Використання цих підходів дозволяє виявити та усунути потенційні загрози ще на ранніх етапах розробки.

Мета роботи: проаналізувати сучасні методи дослідження безпеки веб-додатків, оцінити їх ефективність та практичну застосовність для виявлення й усунення вразливостей з метою підвищення рівня кіберзахисту веб-систем.

1. Методи тестування

Для успішного тестування веб-застосунків необхідно застосовувати систематизований підхід або методологію. Найбільш відомі це OWASP та WASC. Вони є найбільш повними та формалізованими методологіями на сьогоднішній день.

Далі необхідно визначиться з веб-додатком - для дослідження можна взяти останню версію однієї з безкоштовних CMS, і встановити в неї вразливий плагін (вразливі версії можна завантажити з сайту exploit-db.com).

Є кілька принципів тестування, які ми можемо застосувати:

DAST - динамічний (тобто вимагає виконання) аналіз програми без доступу до вихідного коду та серверної частини, по суті BlackBox.

SAST – статичний (тобто не вимагає виконання) аналіз програми з доступом до вихідного коду веб-додатка та до веб-сервера, по суті це аналіз вихідного коду за формальними ознаками наявності вразливостей та аудит безпеки сервера.

IAST – динамічний аналіз безпеки веб-додатку, з повним доступом до вихідного коду, веб-серверу – по суті є WhiteBox тестуванням.

Аналіз вихідного коду – статичний чи динамічний аналіз із доступом до вихідного коду без доступу до серверного оточення.

Ці методи повністю підійдуть для тренування навичок виявлення вразливостей веб-програми за наявності доступу до веб-додатку, або частинок, якщо ви досліджуєте веб-додаток, наприклад, за участю в програмі BugBounty.

2. Основні етапи тестування

Для повноти тестування необхідно намагатися дотримуватися наведених нижче рекомендацій кастомізую ті чи інші етапи в залежності від веб-додатку.

Розвідка включає наступні етапи: сканування портів та піддоменів; дослідження видимого контенту; пошук прихованого контенту (директорій, файлів); визначення платформи та веб-оточення та визначення форм введення.

Контроль доступу передбачає перевірку засобів автентифікації та авторизації; визначення вимог парольної політики; проведення наступних видів тестування: підбору облікових даних, відновлення облікового запису, функцій збереження сесії, функцій ідентифікації облікового запису, перевірку повноважень та прав доступу, перевірка CSRF, а також дослідження сесії (час життя, сесійні токени, ознаки, спроби одночасної роботи і т.д.);

Фазинг параметрів включає тестування додатків до різного виду ін'єкцій (SQL, SOAP, LDAP, XPATH тощо) та тестування додатків до XSS-уразливостей. На даному етапі відбувається перевірки заголовків HTTP; редиректів та переадресацій; виконання команд ОС; локального та віддаленого включення; впровадження XML-сутностей; темплейт-ін'єкцій та взаємодії веб-сокетів.

Перевірки логіки роботи веб-програми передбачають перевірку можливості дублювання чи поділу даних а також тестування логіки роботи програми за клієнта на так званий "Стан гонки" - race condition, каналу передачі та доступності інформації, виходячи з прав доступу або його відсутності.

Перевірка серверного оточення включає:

– перевірку архітектури сервера та серверних облікових записів (служби та послуги), а також прав доступу;

– пошук та виявлення публічних уразливостей;.

– визначення параметрів сервера або компонентів (SSL тощо).

Маючи план тестування програми, ми можемо крок за кроком дослідити всі його компоненти на наявність тих чи інших вразливостей. Виходячи з веб-програми, ті чи інші пункти можуть бути доповнені специфічними для цієї програми перевірками.

Висновок. У результаті проведеного дослідження було проаналізовано основні сучасні методи дослідження безпеки веб-додатків, такі як автоматизоване сканування вразливостей, тестування на проникнення, аналіз вихідного коду та моніторинг поведінки системи. На основі отриманих даних було складено узагальнений план тестування безпеки, який може бути використаний для систематичної перевірки веб-додатків на наявність критичних вразливостей. Застосування такого плану дозволяє підвищити ефективність виявлення загроз і забезпечити більш високий рівень захисту інформаційних систем. Отже, комплексний підхід до тестування безпеки є важливою складовою сучасної практики розробки безпечного програмного забезпечення.

Перелік використаних джерел.

1. R. A. Muzaki, O. C. Briliyant, M. A. Hasditama and H. Ritchi, "Improving Security of Web-Based Application Using ModSecurity and Reverse Proxy in Web Application Firewall," 2020 International Workshop on Big Data and Information Security (IWBIS), Depok, Indonesia, 2020, pp. 85-90, doi: 10.1109/IWBIS50925.2020.9255601.

2. M. Agreindra Helmiawan, E. Firmansyah, I. Fadil, Y. Sofivan, F. Mahardika and A. Guntara, "Analysis of Web Security Using Open Web Application Security Project 10," 2020 8th International Conference on Cyber and IT Service Management (CITSM), Pangkal, Indonesia, 2020, pp. 1-5, doi: 10.1109/CITSM50537.2020.9268856.