МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет комп'ютерних наук,
управління та адміністрування
Кафедра інформаційних
технологій

Кваліфікаційна робота бакалавра

на тему: Розробка технології перевірки на уразливості web-ресурсів

Виконав студент групи К-19
спеціальності 122 Комп'ютерні науки
Джумаєв Сулейман

Керівник к. техн. наук, доцент
Фразе-Фразенко О.О.

Рецензент к.т.н.,
Домаскін О.М.

Одеса 2023

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

СУБД – Система управління базами даних

Хостинг – послуга з надання простору для розміщення сайтів в Інтернеті.

ABAC – Attribute-based access control – Керування доступом на основі атрибутів

CRLF – Carriage Return Line Feed – передача рядка повернення каретки.

CSRF – Cross-Site Request Forgery – міжсайтова підробка запиту.

CSS – Cascading Style Sheets – каскадні таблиці стилів.

DDoS – DDoS attack, (distributed) denial-of-service attack – розподілена атака типу «відмова в обслуговуванні»

HTML – HyperText Markup Language – мова гіпертекстової розмітки.

IDS – Intrusion Detection System – Система виявлення вторгнень

IPS – Intrusion Prevention System – Система запобігання вторгненням

MAC – Message Authentication Code – код автентифікації повідомлення.

MySQL – вільна система управління базами даних (СУБД).

ODBC – Open Database Connectivity Standard – стандарт підключення до відкритих баз даних.

PHP – Hypertext Preprocessor – препроцесор гіпертекста.

RBAC – Role Based Access Control – Керування доступом на основі ролей

SIEM – Security information and event management – Інформація про безпеку та керування подіями

SSL – Secure Sockets Layer – рівень захищених сокетів

TLS – Transport Layer Security – захист на транспортному рівні

UML – Unified Modeling Language – мова візуального моделювання.

URI – Uniform Resource Identifier – уніфікований ідентифікатор ресурсу.

WAF – Web Application Firewall – міжмережний екран вебзастосунків

XSS – Cross-Site Scripting – міжсайтовий скриптинг.

ВСТУП

Веб-ресурси стали невід'ємною складовою сучасного інформаційного середовища. Однак, разом зі зростанням їх популярності з'явилися й загрози, пов'язані з уразливістю веб-ресурсів. Відомо, що зловмисники активно використовують вразливості веб-сайтів для здійснення атак, таких як виток конфіденційної інформації, впровадження шкідливого коду та порушення цілісності систем.

Мета даної дипломної роботи полягає у розробці технології дослідження на уразливість веб-ресурсів з метою виявлення потенційних вразливостей та запобігання можливим атакам. Використання такої технології дозволить забезпечити безпеку веб-додатків, підвищити рівень захищеності інформації та запобігти можливим наслідкам від експлуатації вразливостей.

Дослідження на уразливість веб-ресурсів є складним завданням, оскільки вони можуть включати різноманітні компоненти, такі як веб-сервери, бази даних, скриптові мови та інші, кожен з яких може мати свої власні потенційні вразливості. Тому, для ефективного виявлення вразливостей, потрібна комплексна методологія та інструменти, які дозволять здійснювати систематичний аналіз та перевірку безпеки веб-ресурсів.

У рамках дипломної роботи будуть вивчені сучасні підходи та методики аналізу вразливостей веб-додатків, а також розроблено технологію дослідження на уразливість.

Дипломна робота містить 52 сторінок, 6 рисунків та 10 посилань.

1 ПОНЯТТЯ ТА ПРИНЦИПИ БЕЗПЕКИ WEB-РЕСУРСІВ

Поняття веб-ресурсів стосуються різних елементів і компонентів веб-сайту або веб-програми, які потребують захисту від загроз безпеки та вразливостей [1]. Ці ресурси включають:

Веб-сторінки: веб-сторінки є основними компонентами веб-сайту або веб-програми. Вони складаються з файлів HTML, CSS і JavaScript, які визначають інтерфейс користувача, макет і інтерактивні елементи. Захист веб-сторінок включає захист від несанкціонованого доступу, модифікації або впровадження шкідливого коду.

Дані та бази даних: веб-ресурси часто покладаються на бази даних для зберігання та керування конфіденційними даними, такими як інформація про користувачів, платіжні відомості або конфіденційна інформація. Захист даних і баз даних передбачає впровадження належного контролю доступу, шифрування та механізмів резервного копіювання для запобігання несанкціонованому доступу, витоку даних або маніпулюванню даними.

Автентифікація та авторизація користувачів: веб-ресурси часто вимагають автентифікації користувачів для перевірки ідентичності користувачів і авторизації їх доступу до певних функцій або даних. Захист автентифікації користувача передбачає впровадження надійних механізмів автентифікації, таких як багатофакторна автентифікація, і захист облікових даних користувача від крадіжки або несанкціонованого використання.

API (інтерфейси прикладного програмування): веб-ресурси можуть взаємодіяти із зовнішніми службами або програмами через API. Захист API передбачає впровадження належної автентифікації, авторизації та перевірки введених даних, щоб запобігти несанкціонованому доступу, витоку даних або зловмисному використанню кінцевих точок API.

Введення користувачами та обробка форм: веб-ресурси зазвичай покладаються на введення користувачами через форми, вікна пошуку або завантаження файлів. Забезпечення безпеки користувацьких даних передбачає

впровадження перевірки введених даних і очищення, щоб запобігти таким атакам, як міжсайтовий сценарій (XSS), впровадження SQL або вразливості завантаження файлів.

Керування сеансами: веб-ресурси часто використовують механізми керування сеансами для підтримки сеансів користувачів і зберігання даних, пов'язаних із сеансами. Захист керування сеансом передбачає використання безпечних маркерів сеансу, реалізацію належного завершення сеансу та запобігання захопленню сеансу або атакам фіксації.

Брандмауери веб-додатків (WAF): WAF відіграють вирішальну роль у безпеці веб-ресурсів. Вони відстежують і фільтрують вхідний веб-трафік, виявляють і блокують зловмисні запити або атаки, такі як міжсайтовий сценарій, впровадження SQL або розподілені атаки на відмову в обслуговуванні (DDoS).

Аудити безпеки та оцінки вразливостей: регулярні перевірки безпеки та оцінки вразливостей необхідні для виявлення та усунення слабких місць безпеки веб-ресурсів. Ці оцінки включають виявлення вразливостей, неправильних конфігурацій або застарілих версій програмного забезпечення, якими можуть скористатися зловмисники.

Реагування на інциденти та моніторинг: безпека веб-ресурсів вимагає постійного моніторингу та можливостей реагування на інциденти. Моніторинг передбачає відстеження та аналіз системних журналів, мережевого трафіку та подій безпеки для виявлення інцидентів безпеки та оперативного реагування на них.

Відповідність вимогам і нормативні вимоги: залежно від галузі чи географічного розташування, веб-ресурси можуть мати потребу в дотриманні певних стандартів безпеки та правил. Забезпечення відповідності передбачає впровадження необхідних заходів контролю, проведення аудитів і ведення документації для демонстрації відповідності застосовним стандартам безпеки.

Загалом безпека веб-ресурсів охоплює захист веб-сторінок, даних, механізмів автентифікації, API, введення користувачами, керування сесіями та

міркування щодо відповідності. Впроваджуючи надійні заходи безпеки, регулярно оцінюючи вразливості та оперативно реагуючи на інциденти безпеки, організації можуть захистити свої веб-ресурси від потенційних загроз і забезпечити безпечну роботу користувачів.

1.1 Вразливості веб-ресурсів

Безпека веб-ресурсів - це заходи і практики, реалізовані для захисту веб-сайтів, веб-додатків та інших онлайн-активів від несанкціонованого доступу, порушень даних, маніпуляцій та інших зловмисних дій. Це передбачає захист конфіденційності, цілісності та доступності веб-ресурсів, щоб гарантувати, що вони функціонують належним чином і забезпечують безпечний досвід користувача.

Безпека веб-ресурсів охоплює широкий спектр компонентів, технологій і процесів, спрямованих на пом'якшення загроз і вразливостей. Нижче наведені деякі ключові аспекти безпеки веб-ресурсів [2]:

Автентифікація та контроль доступу. Автентифікація – це процес перевірки ідентичності користувачів і надання їм відповідних привілеїв доступу. Впровадження надійних механізмів автентифікації, таких як імена користувачів, паролі, біометрія або багатофакторна автентифікація, допомагає гарантувати, що лише авторизовані особи можуть отримати доступ до веб-ресурсів. Механізми контролю доступу, включаючи контроль доступу на основі ролей (RBAC) або контроль доступу на основі атрибутів (ABAC), визначають, до яких дій і даних користувачі можуть отримати доступ у межах веб-ресурсу.

Шифрування та безпечний зв'язок. Шифрування відіграє важливу роль у безпеці веб-ресурсів, захищаючи дані, що передаються через Інтернет. Протоколи Secure Sockets Layer/Transport Layer Security (SSL/TLS) шифрують зв’язок між клієнтами та веб-серверами, запобігаючи несанкціонованому перехопленню або втручанню. Впровадження HTTPS (HTTP через SSL/TLS)

гарантує, що дані, якими обмінюються користувачі та веб-ресурс, залишаються конфіденційними та не можуть бути легко розшифровані зловмисниками.

Безпечне кодування та безпека програм. Розробка безпечних веб-додатків має вирішальне значення для запобігання вразливостям, якими можуть скористатися зловмисники. Захищені методи кодування, такі як перевірка вхідних даних, кодування вихідних даних і належна обробка помилок, зменшують ризик поширених уразливостей веб-додатків, таких як міжсайтовий сценарій (XSS), впровадження SQL або віддалене виконання коду. Регулярне оновлення та виправлення веб-додатків, фреймворків і плагінів допомагає усунути відомі вразливості безпеки.

Брандмауери веб-додатків (WAF). WAF діють як захисний шар між веб-ресурсами та потенційними загрозами. Вони відстежують вхідний веб-трафік, аналізують моделі та поведінку та блокують підозрілу чи зловмисну діяльність. WAF можуть виявляти та пом’якшувати різні атаки, включаючи атаки XSS, впровадження SQL і DDoS-атаки. Вони забезпечують додаткову лінію захисту, відфільтровуючи потенційно шкідливі запити та надаючи систему раннього попередження про потенційні загрози.

Аудит безпеки та оцінка вразливостей. Регулярні перевірки безпеки та оцінки вразливостей необхідні для виявлення та усунення слабких місць у веб-ресурсах. Ці оцінки передбачають систематичні перевірки веб-додатків, мереж і серверів для виявлення потенційних уразливостей і неправильних налаштувань. Тестування на проникнення, яке проводять спеціалісти з безпеки, намагається використати виявлені вразливості для перевірки надійності заходів безпеки веб-ресурсу.

Моніторинг безпеки та реагування на інциденти. Постійний моніторинг веб-ресурсів дозволяє своєчасно виявляти і реагувати на інциденти безпеки. Системи безпеки та керування подіями (SIEM), системи виявлення вторгнень (IDS) і системи запобігання вторгненням (IPS) можуть допомогти контролювати мережевий трафік, виявляти підозрілі дії та генерувати

сповіщення. Створення плану реагування на інциденти забезпечує структурований підхід до обробки інцидентів безпеки, мінімізуючи потенційну шкоду та скорочуючи час простою.

Навчання та обізнаність користувачів. Обізнаність і освіта користувачів відіграють важливу роль у безпеці веб-ресурсів. Навчання користувачів щодо потенційних ризиків, звичок безпечного перегляду, розпізнавання спроб фішингу та дотримання гігієни паролів може значно зменшити ймовірність успішних атак.

1.2 Огляд типів та технік перевірки на уразливості

У сучасному цифровому середовищі, де кіберзагрози постійно розвиваються, необхідно вживати активних заходів для виявлення та пом’якшення вразливостей у системах. Інструменти та методи тестування вразливостей відіграють вирішальну роль у цьому процесі, виявляючи слабкі місця, неправильні конфігурації та потенційні точки входу, якими можуть скористатися зловмисники. У цьому всебічному огляді досліджуються різні типи інструментів і методів тестування вразливостей, які використовуються фахівцями з безпеки для оцінки та покращення стану безпеки систем, мереж і програм.

Тестування вразливостей (оцінка вразливостей або сканування вразливостей) – це процес виявлення слабких місць безпеки та вразливостей в інформаційній системі. Він передбачає систематичне сканування та тестування мереж, систем, програм та інфраструктури для виявлення потенційних уразливостей, якими можуть скористатися зловмисники.

1.3 Методи тестування уразливостей

Існують різні типи перевірки вразливості, кожна з яких служить певній меті виявлення та оцінки вразливостей [3]. До них належать:

- тестування вразливості мережі: цей тип тестування зосереджений на виявленні вразливостей у мережевій інфраструктурі, включаючи брандмауери, маршрутизатори, комутатори та інші мережеві пристрої. Інструменти та методи перевірки вразливості мережі допомагають виявити неправильні конфігурації, слабкі протоколи шифрування, відкриті порти та інші недоліки, пов’язані з мережею;
- тестування вразливості веб-додатків: веб-додатки є звичайною мішенню для зловмисників. Тестування вразливостей веб-додатків спрямоване на виявлення вразливостей, таких як міжсайтовий сценарій (XSS), впровадження SQL, віддалене виконання коду та незахищені механізми автентифікації. Ці тести гарантують, що веб-додатки стійкі до типових векторів атак;
- тестування вразливості мобільних додатків: мобільні пристрої та додатки стають все більш популярними об’єктами для кібератак. Інструменти та методи тестування вразливостей мобільних додатків зосереджені на виявленні вразливостей, характерних для мобільних платформ, таких як незахищене зберігання даних, слабке шифрування, неадекватна автентифікація або підробка коду;
- тестування вразливості бездротової мережі: бездротові мережі вразливі до різних уразливостей, включаючи слабке шифрування, фальшиві точки доступу та неавторизований доступ. Інструменти та методи перевірки вразливості бездротової мережі допомагають виявити ці слабкі місця, оцінити безпеку бездротових мереж і забезпечити відповідність найкращим практикам безпеки;
- перевірка вразливості баз даних: бази даних зберігають важливу та конфіденційну інформацію, що робить їх привабливими цілями для зловмисників. Інструменти та методи тестування вразливості бази даних спрямовані на виявлення вразливостей у системах керування

базами даних (СУБД), включаючи слабкий контроль доступу, недоліки ін’єкцій, підвищення привілеїв і неправильні конфігурації.

Для виявлення складних уразливостей і забезпечення комплексної оцінки безпеки використовуються методи ручного тестування вразливостей.

Серед цих методів:

- перевірка вихідного коду: перевірка вихідного коду вручну передбачає перевірку вихідного коду програм для виявлення вразливостей, які можуть бути пропущені інструментами автоматичного сканування. Ця техніка дозволяє фахівцям із безпеки отримати глибше розуміння логіки програми та виявити потенційні слабкі місця;
- тестування на проникнення. Тестування на проникнення, також відоме як етичний хакерство, передбачає імітацію реальних атак для виявлення вразливостей і оцінки ефективності засобів контролю безпеки. Тестери проникнення використовують комбінацію автоматизованих інструментів і ручних методів для використання вразливостей і надання рекомендацій щодо покращення;
- тестування соціальної інженерії: Тестування соціальної інженерії оцінює вразливість організації до маніпуляцій та обману шляхом імітації фішингових атак, видавання себе за іншу особу або інших форм тактики соціальної інженерії. Ця методика допомагає оцінити ефективність навчання з питань безпеки та виявити потенційні недоліки в поведінці співробітників;
- Фаззинг передбачає надсилання недійсних або неочікуваних вхідних даних до програми для виявлення вразливостей, пов’язаних з обробкою вхідних даних і керуванням помилками. Це допомагає виявити недоліки, такі як переповнення буфера, помилки перевірки введення та несподівані збої в програмі.

Тестування вразливостей – це не одноразова діяльність, а постійний процес. Вкрай важливо створити програму управління вразливістю, щоб визначити пріоритети та усунути виявлені вразливості. Це включає:

- пріоритезація вразливостей: не всі вразливості мають однаковий рівень критичності. Пріоритезація включає оцінку серйозності та потенційного впливу вразливостей, щоб зосередити ресурси на пом’якшенні найбільш критичних з них;
- керування виправленнями: швидке застосування виправлень безпеки та оновлень має вирішальне значення для усунення відомих вразливостей. Організації повинні встановити надійні процеси керування виправленнями, щоб підтримувати системи та програми в актуальному стані;
- планування виправлення: структурований підхід до виправлення передбачає розробку плану усунення виявлених вразливостей, розподіл відповідальності, встановлення часових рамок і відстеження прогресу. Це забезпечує систематичний і ефективний процес рекультивації;
- поінформованість про безпеку та навчання: навчання співробітників загальним слабким місцям, методам безпечного кодування та найкращим практикам безпеки допомагає запобігти появі нових слабких місць. Постійне навчання з питань безпеки має важливе значення для створення в організації культури безпеки;
- безперервний моніторинг: впровадження безперервного моніторингу та контролю безпеки допомагає виявляти нові вразливості, відстежувати зміни в середовищі та оперативно реагувати на нові загрози.

2 АНАЛІЗ ІСНУЮЧИХ МЕТОДІВ ПЕРЕВІРКИ НА УРАЗЛИВОСТІ WEB-РЕСУРСІВ

2.1 Інструменти автоматизованої перевірки на уразливості

Інструменти автоматичного тестування вразливостей призначені для сканування систем, мереж і програм на наявність уразливостей без ручного втручання. Ці інструменти використовують різні техніки та принципи для виявлення потенційних слабких місць і надання розуміння стану безпеки в організації. Існують різні типи інструментів для автоматичного тестування вразливостей за принципом їхньої роботи.

Сканери вразливостей – це автоматизовані інструменти, які сканують мережі, системи та програми для виявлення відомих уразливостей і неправильних конфігурацій. Ці сканери зазвичай працюють на основі бази даних відомих уразливостей, яка регулярно оновлюється, щоб включити останні загрози. Під час сканування сканери вразливостей надсилають певні запити або зонди до цільової системи та аналізують відповіді, щоб виявити вразливості. Вони часто використовують такі методи, як сканування портів, ідентифікація служби та перевірка вразливості, щоб оцінити безпеку цілі. Приклади популярних сканерів вразливостей включають Nessus, OpenVAS і QualysGuard.

Сканери веб-додатків зосереджуються саме на оцінці безпеки веб-додатків. Вони автоматично сканують веб-сторінки, імітують взаємодію користувачів і виявляють такі вразливості, як міжсайтовий сценарій (XSS), впровадження SQL і незахищені механізми автентифікації. Сканери веб-додатків зазвичай використовують комбінацію статичного аналізу (вивчення вихідного коду) і динамічного аналізу (взаємодія з програмою) для виявлення вразливостей. Деякі добре відомі сканери веб-додатків включають Acunetix, Burp Suite і OWASP ZAP.

Мережеві сканери перевіряють мережеву інфраструктуру та пристрої, щоб виявити вразливі місця та неправильні налаштування. Вони сканують мережі на наявність відкритих портів, служб і потенційних слабких місць, якими можуть скористатися зловмисники. Мережеві сканери допомагають виявити вразливі пристрої, слабкі протоколи шифрування або неправильно налаштовані брандмауери чи маршрутизатори. Їх можна використовувати як для внутрішньої, так і для зовнішньої оцінки мережі. Серед популярних інструментів мережевого сканування Nmap, OpenVAS і Nexpose.

Інструменти фаззингу, також відомі як фаззери, спрямовані на виявлення вразливостей шляхом надсилання неочікуваних або неправильних вхідних даних до цільової програми. Фаззинг допомагає виявити такі недоліки, як переповнення буфера, помилки підтвердження введення або збої в програмі. Інструменти фаззингу генерують великий обсяг випадкових або мутованих вхідних даних і відстежують реакцію програми на наявність аномалій або збоїв. Спостерігаючи за поведінкою програми при різних вхідних даних, фаззери можуть визначити потенційні вразливості. American Fuzzy Lop (AFL), Peach Fuzzer і Sulley є широко використовуваними інструментами фаззингу.

Інструменти керування конфігурацією безпеки зосереджені на оцінці параметрів безпеки та конфігурацій систем, пристроїв і програм. Вони порівнюють поточну конфігурацію з попередньо визначеними політиками безпеки чи найкращими практиками, висвітлюючи будь-які відхилення чи недоліки. Ці інструменти допомагають організаціям виявляти незахищені конфігурації, слабкі засоби контролю доступу або інші неправильні конфігурації, які можуть наражати системи на вразливі місця. Приклади інструментів керування конфігурацією безпеки включають Tripwire, OpenSCAP і Microsoft Baseline Security Analyzer (MBSA).

Деякі засоби тестування вразливостей інтегруються з конвеєрами CI/CD для автоматизації оцінки безпеки протягом життєвого циклу розробки програмного забезпечення. Ці інтеграції дозволяють організаціям виявляти

вразливості на ранніх етапах процесу розробки та забезпечувати безпечне розгортання коду. Такі інструменти, як Jenkins, GitLab CI/CD і Travis CI, можна налаштувати для включення сканування вразливостей як частини безперервної інтеграції та робочого процесу доставки.

Інструменти автоматичного тестування вразливостей зазвичай працюють на основі таких принципів [4]:

1. Виявлення: Інструменти використовують різні методи для ідентифікації цільових систем, пристроїв або програм, які потрібно сканувати, за допомогою ручного введення або автоматичних механізмів виявлення.
2. Сканування: інструменти сканують ціль за допомогою комбінації методів, таких як сканування портів, ідентифікація служби, перевірка вразливості або розмитість вхідних даних. Інструменти надсилають певні запити або зонди до цілі, аналізуючи відповіді на наявність індикаторів вразливості.
3. Виявлення вразливостей: інструменти порівнюють зібрані дані з базою даних відомих вразливостей або попередньо визначених політик безпеки, щоб виявити потенційні слабкі місця. Це передбачає зіставлення сигнатур, шаблонів або відомих індикаторів уразливості.
4. Звітування: після завершення процесу сканування інструменти створюють вичерпні звіти, у яких висвітлюються виявлені вразливості, їх рівні серйозності, системи, на які впливає, і рекомендовані кроки для усунення. Звіти допомагають визначити пріоритети та ефективно усунути вразливості.

2.2 Аналіз інструментів для перевірки на уразливості

Популярними інструментами тестування вразливостей є Nessus, OpenVAS, Burp Suite, OWASP ZAP та SQLMap.

Nessus – це широко використовуваний інструмент сканування вразливостей, розроблений Tenable Network Security [5].

Рисунок 1.1 – Скріншоти застосунку Nessus

Він призначений для виявлення вразливостей у системах, мережах і програмах, допомагаючи організаціям оцінити рівень безпеки та визначити пріоритети для усунення.

Переваги Nessus:

– комплексне виявлення вразливостей: Nessus пропонує величезну базу даних перевірки вразливостей, яка постійно оновлюється останніми загрозами безпеці. Він може ефективно визначати широкий спектр вразливостей, включаючи типові слабкі місця, неправильні конфігурації, застарілі версії програмного забезпечення та відомі експлойти;

– масштабованість і гнучкість: Nessus підтримує сканування як малих, так і великих середовищ, що робить його придатним для організацій будь-якого розміру. Він пропонує гнучкі варіанти розгортання,

включаючи локальні та хмарні рішення, що дозволяє користувачам адаптувати інструмент до своїх конкретних потреб;

- розширені можливості сканування: Nessus надає різні параметри сканування, наприклад сканування на основі агента, сканування з автентифікацією та сканування відповідності. Автентифіковане сканування дозволяє проводити більш глибоку перевірку, використовуючи облікові дані для доступу до системи, забезпечуючи точнішу та детальнішу оцінку вразливостей;
- детальні звіти та аналіз: Nessus створює вичерпні звіти з детальною інформацією про виявлені вразливості, включаючи рівні серйозності, уражені системи та рекомендації щодо усунення. Звіти можна налаштовувати та експортувати в різні формати, що полегшує спілкування із зацікавленими сторонами та допомагає в процесі відновлення;
- інтеграція та співпраця: Nessus інтегрується з іншими інструментами та платформами безпеки, такими як системи безпеки інформації та керування подіями (SIEM), системи продажу квитків та рішення для керування вразливими місцями. Це дозволяє оптимізувати робочі процеси, автоматизувати обмін даними та покращити співпрацю між командами безпеки;
- аудит відповідності: Nessus містить попередньо визначені політики аудиту та шаблони для різних стандартів відповідності, таких як PCI DSS, HIPAA та контрольні показники CIS. Він може сканувати системи на наявність порушень відповідності та надавати вказівки щодо виконання конкретних вимог, допомагаючи організаціям підтримувати відповідність нормативним вимогам.

Недоліки Nessus:

- вартість: Nessus є комерційним інструментом і має вартість ліцензування. Хоча він пропонує безкоштовну версію з обмеженою

функціональністю, організаціям, яким потрібні розширені функції та комплексне керування вразливістю, можливо, доведеться інвестувати в комерційну версію, яка може бути відносно дорогою;

- складність: Nessus – це багатофункціональний інструмент, який пропонує численні параметри конфігурації та можливості налаштування. Ця складність може бути надзвичайною для початківців користувачів або організацій з обмеженими ресурсами чи досвідом управління вразливістю. Для ефективного використання інструменту необхідні відповідна підготовка та досвід;
- хибно-позитивні та помилково-негативні результати: як і будь-який інструмент сканування вразливостей, Nessus може генерувати хибні спрацьовування (виявлення вразливості, якої не існує) або хибно-негативні (нездатність визначити справжню вразливість). Користувачам необхідно уважно переглядати та перевіряти результати сканування, щоб уникнути непотрібних зусиль щодо виправлення або не помітити критичні вразливості;
- тривалість сканування. Сканування Nessus може зайняти багато часу, особливо у великих середовищах або під час виконання автентифікованих сканувань. Тривалість сканування залежить від таких факторів, як пропускна здатність мережі, швидкість реагування системи та глибина сканування. Велика тривалість сканування може вплинути на ефективність роботи та доступність ресурсів під час сканування.

OpenVAS – це інструмент сканування вразливостей і керування ними з відкритим вихідним кодом, який допомагає організаціям виявляти й оцінювати вразливості в системах, мережах і програмах [6].

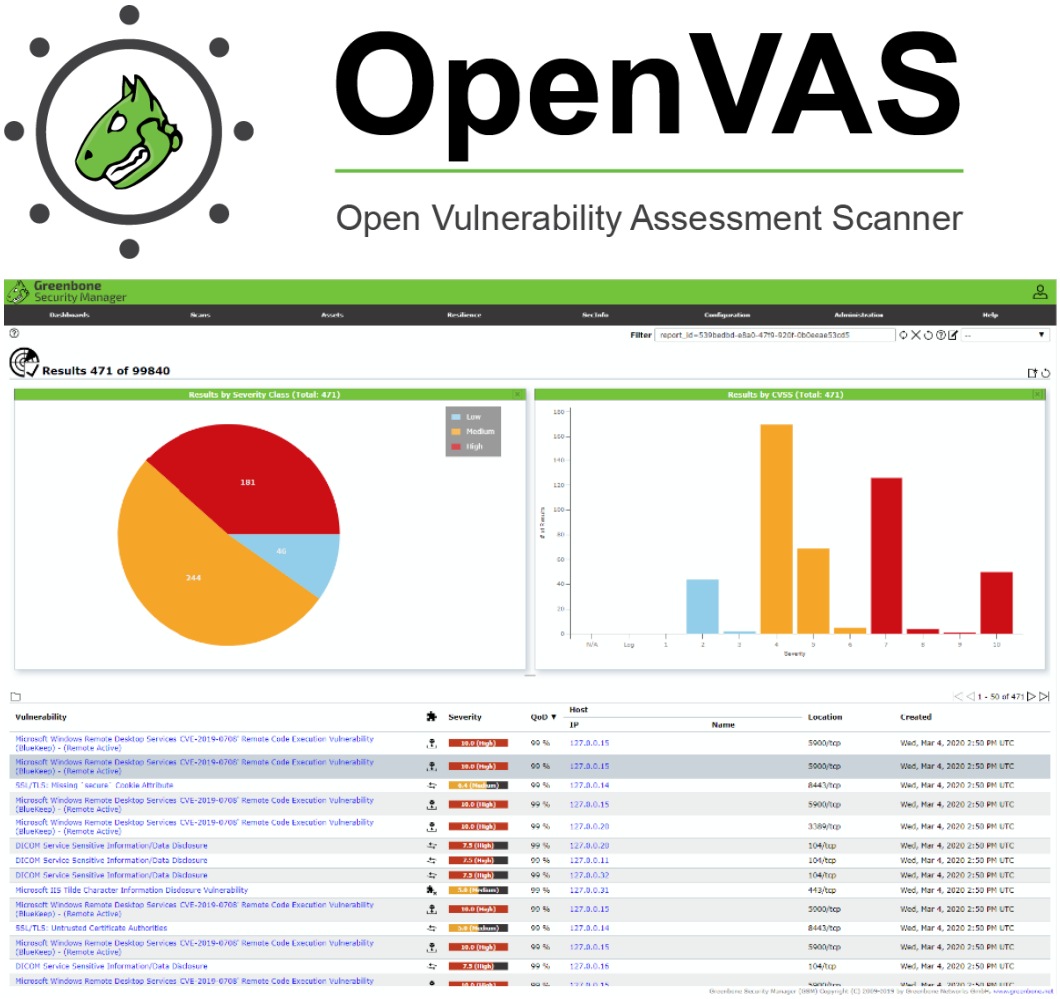

Рисунок 1.2 – Скріншоти застосунку OpenVAS

Він надає повний набір функцій для сканування вразливостей, звітування та виправлення.

Переваги OpenVAS:

– відкритий і безкоштовний: OpenVAS – це інструмент із відкритим вихідним кодом, доступний за ліцензією GNU General Public License (GPL), що означає, що його можна вільно використовувати та змінювати. Це робить його доступним для організацій з обмеженим бюджетом, сприяючи ширшому впровадженню та внеску спільноти;

– розширена база знань про вразливості: OpenVAS підтримує велику та регулярно оновлювану базу даних про відомі вразливості. Він включає перевірки на загальні слабкі місця, неправильні конфігурації та

проблеми безпеки в різних системах, програмах і платформах. Велика база знань забезпечує комплексне виявлення вразливостей;

- гнучкі параметри розгортання: OpenVAS пропонує гнучкі варіанти розгортання, дозволяючи користувачам вибирати між локальною інсталяцією або використанням попередньо налаштованих віртуальних пристроїв. Ця гнучкість полегшує інтеграцію OpenVAS в існуючу інфраструктуру та адаптацію її до конкретних організаційних вимог;
- настроювані профілі сканування: OpenVAS дозволяє користувачам створювати індивідуальні профілі сканування відповідно до їхніх конкретних потреб. Це дає змогу детально контролювати процес сканування, включаючи вибір плагінів, інтенсивність сканування та цільові перевірки вразливостей;
- комплексна звітність: OpenVAS створює докладні та вичерпні звіти, у яких висвітлюються виявлені вразливості, їх рівні серйозності та рекомендації щодо виправлення. Звіти можна налаштовувати та експортувати в різні формати, що полегшує спілкування із зацікавленими сторонами та допомагає в процесі відновлення;
- можливості інтеграції: OpenVAS забезпечує інтеграцію з іншими інструментами та платформами безпеки, такими як системи SIEM (інформаційна система безпеки та управління подіями), системи продажу квитків і рішення для керування вразливими місцями. Це забезпечує безперебійний обмін даними, спрощені робочі процеси та покращену співпрацю між командами безпеки.

Недоліки OpenVAS:

- складність навчання: OpenVAS може бути складним, особливо для користувачів, які тільки починають роботу з уразливостями. Початкове налаштування та конфігурація можуть потребувати певних

технічних знань. Для ефективного використання OpenVAS необхідні адекватне навчання та розуміння концепцій управління вразливістю;

- обмежена швидкість сканування: OpenVAS може мати меншу швидкість сканування порівняно з деякими комерційними інструментами сканування вразливостей. Процес сканування може зайняти багато часу, особливо у великих мережах або складних середовищах. Тривалість сканування може змінюватися залежно від таких факторів, як пропускна здатність мережі, швидкість реагування системи та конфігурація сканування;
- помилкові спрацьовування та помилково негативні результати: як і будь-який інструмент сканування вразливостей, OpenVAS може давати хибні спрацьовування (визначаючи вразливість, якої не існує) або хибно негативні (не виявляти справжню вразливість). Користувачам необхідно уважно переглядати та перевіряти результати сканування, щоб уникнути непотрібних зусиль щодо виправлення або не помітити критичні вразливості;
- відсутність офіційної технічної підтримки: будучи інструментом з відкритим кодом, OpenVAS не надає офіційної технічної підтримки безпосередньо від команди розробників. Однак підтримка спільноти доступна через форуми, списки розсилки та онлайн-ресурси. Організаціям може знадобитися покладатися на підтримку спільноти або звернутися за професійними послугами для більш розширених вимог підтримки;
- залежність від оновлень бази даних: OpenVAS покладається на регулярні оновлення своєї бази даних уразливостей, щоб бути в курсі останніх загроз і вразливостей. Користувачі повинні переконатися, що вони мають активну підписку, щоб отримувати своєчасні оновлення та скористатися перевагами останніх перевірок уразливостей.

Burp Suite – популярний і широко використовуваний інструмент тестування безпеки веб-додатків, розроблений PortSwigger [7].

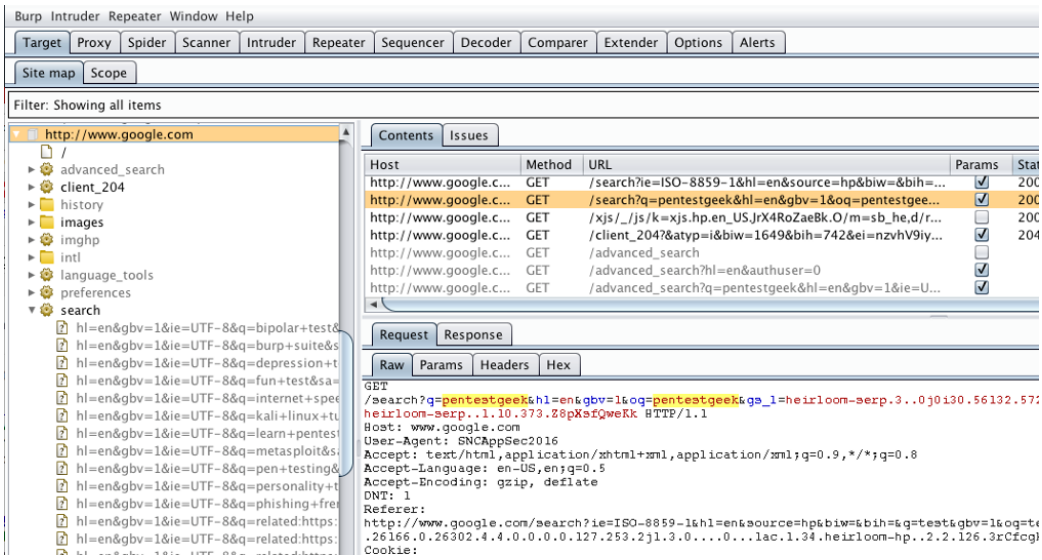

Рисунок 1.3 – Скріншоти застосунку Burp Suite

Він призначений для допомоги фахівцям із безпеки у виявленні вразливостей та оцінці безпеки веб-додатків. Burp Suite пропонує повний набір функцій для ручного та автоматичного тестування, що дозволяє користувачам виконувати різні оцінки безпеки.

Переваги Burp Suite:

– широкі функціональні можливості: Burp Suite надає широкий спектр інструментів і можливостей для тестування безпеки веб-додатків. Він містить такі функції, як веб-проксі, сканер, порушник, повторювач, секвенсор тощо. Ці інструменти дозволяють комплексно тестувати різні аспекти веб-додатків, наприклад перевірку введених даних, керування сеансами, механізми автентифікації тощо;

- проксі-перехоплювач: однією з ключових особливостей Burp Suite є проксі-перехоплювач. Це дозволяє користувачам перехоплювати та змінювати HTTP-запити та відповіді між браузером і веб-програмою. Ця функція є цінною для ідентифікації та маніпулювання вхідними параметрами, заголовками, файлами cookie та іншими елементами для тестування елементів керування безпекою програми;
- сканування веб-додатків: Burp Suite містить автоматичний сканер уразливостей, який може сканувати веб-додатки, виявляти потенційні вразливості та надавати докладні звіти. Сканер перевіряє загальні вразливості, такі як XSS (міжсайтовий сценарій), впровадження SQL, впровадження команд тощо. Це допомагає заощадити час, автоматизувавши ідентифікацію вразливостей;
- налаштування та розширення: Burp Suite пропонує широкі можливості налаштування та підтримує використання розширень. Користувачі можуть створювати власні сценарії, плагіни або доповнення, щоб розширити функціональність інструменту. Це дозволяє користувачам адаптувати інструмент до своїх конкретних потреб і розширити його можливості для спеціалізованих сценаріїв тестування;
- спільне тестування: Burp Suite підтримує співпрацю між членами команди, дозволяючи кільком користувачам працювати над одним проектом одночасно. Ця функція полегшує командну роботу, обмін знаннями та ефективне тестування складних веб-додатків;
- активна спільнота та оновлення: Burp Suite має активну спільноту користувачів, яка ділиться знаннями, порадами та розширеннями через форуми та онлайн-ресурси. PortSwigger, розробник Burp Suite, регулярно оновлює інструмент для усунення нових вразливостей, підвищення продуктивності та впровадження нових функцій на основі відгуків користувачів.

Недоліки Burp Suite:

- складність навчання: Burp Suite – це потужний інструмент із численними функціями та функціональними можливостями, які можуть зробити його непосильним для початківців або користувачів із обмеженим досвідом тестування безпеки веб-додатків. Для ефективного використання інструменту та інтерпретації його результатів потрібен певний рівень знань і досвіду;
- вартість професійної версії: у той час як Burp Suite пропонує безкоштовну версію (Burp Suite Community Edition) з обмеженими можливостями, професійна версія, яка включає розширені можливості, має вартість ліцензування. Вартість може бути обмежуючим фактором для окремих осіб або організацій з обмеженим бюджетом;
- помилкові спрацьовування та помилково негативні результати: як і будь-який автоматизований інструмент сканування, сканер уразливостей Burp Suite може генерувати помилкові спрацьовування або помилково негативні результати. Точність результатів сканування залежить від таких факторів, як складність програми, налаштування конфігурації, а також знання та досвід користувача. Ретельна перевірка та перевірка вручну необхідні, щоб не покладатися лише на автоматизовані результати;
- вплив на продуктивність: інтенсивне сканування або тестування за допомогою Burp Suite може потенційно вплинути на продуктивність цільової веб-програми або мережі. Користувачі повинні пам’ятати про ресурси, які споживає інструмент, і розглянути можливість відповідного налаштування швидкості або обсягу сканування, щоб мінімізувати вплив на продуктивність.

Nikto – це сканер уразливостей веб-сервера з відкритим вихідним кодом, який сканує веб-сервери на відомі вразливості, неправильні конфігурації та застарілі версії програмного забезпечення [8].


```
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          87.229.144.34
+ Target Hostname:    google.ru
+ Target Port:        80
+ Start Time:         2015-02-26 14:51:38 (GMT6)
---------------------------------------------------------------------------
+ Server: gws
+ Uncommon header 'alternate-protocol' found, with contents: 80:quic,p=0.08
+ The X-Content-Type-Options header is not set. This could allow the user agent
to render the content of the site in a different fashion to the MIME type
+ Root page / redirects to: http://www.google.ru/
+ Server banner has changed from 'gws' to 'sffe' which may suggest a WAF, load b
alancer or proxy is in place
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ OSVDB-5737: WebLogic may reveal its internal IP or hostname in the Location he
ader. The value is "http://87.229.144.34/".
```

Рисунок 1.4 – Скріншоти застосунку Nikto

Він призначений для виконання комплексного сканування та надання докладних звітів про потенційні вразливості та неправильні налаштування.

Переваги Nikto:

- відкритий і безкоштовний: Nikto – це інструмент із відкритим кодом, доступний безкоштовно за GNU General Public License (GPL). Це робить його доступним для організацій з обмеженим бюджетом і заохочує внески спільноти та покращення;
- широкий спектр тестів: Nikto пропонує широкий спектр тестів і перевірок для виявлення типових проблем безпеки та вразливостей у веб-серверах і веб-додатках. Він шукає відомі вразливості, застарілі

версії програмного забезпечення, неправильні конфігурації та інші потенційні недоліки, якими можуть скористатися зловмисники;

- комплексні можливості сканування: Nikto виконує комплексне сканування, яке включає перевірки на наявність неправильних конфігурацій сервера, незахищених конфігурацій, файлів і каталогів за замовчуванням, застарілих версій програмного забезпечення тощо. Він забезпечує цілісне уявлення про безпеку веб-серверів і програм;
- простий у використанні: Nikto має простий і зручний інтерфейс командного рядка, що робить його доступним навіть для користувачів з обмеженим досвідом тестування веб-безпеки. Він надає чіткі інструкції та оновлення прогресу під час процесу сканування, що дозволяє користувачам легко контролювати хід сканування;
- швидкість та ефективність: Nikto відомий своєю швидкістю та ефективністю сканування веб-серверів і програм. Він може швидко сканувати кілька цілей одночасно, що робить його придатним для великомасштабних проектів сканування або чутливих до часу оцінок;
- регулярні оновлення: Nikto активно підтримується та оновлюється спільнотою розробників. Регулярні оновлення гарантують, що інструмент залишається в курсі нових загроз і включає перевірки на наявність останніх вразливостей і проблем безпеки.

Недоліки Nikto:

- відсутність графічного інтерфейсу користувача: Nikto – це в першу чергу інструмент командного рядка, який може створити проблеми для користувачів, які віддають перевагу графічному інтерфейсу користувача (GUI) або не знайомі з інтерфейсами командного рядка. Однак різні сторонні інструменти та інтеграції надають оболонки GUI для Nikto, пропонуючи більш інтуїтивно зрозумілий інтерфейс;
- помилкові спрацьовування та помилково негативні результати: як і будь-який автоматизований сканер уразливостей, Nikto може видавати

хибні спрацьовування (неправильне визначення вразливостей, яких не існує) або хибні негативні результати (відсутні фактичні вразливості). Користувачам необхідно підтвердити та перевірити результати сканування, щоб уникнути непотрібних зусиль щодо виправлення або не помітити критичні проблеми безпеки;

- обмежене охоплення розширених уразливостей: Хоча Nikto охоплює широкий спектр поширених уразливостей і неправильних конфігурацій, він може не охоплювати більш складні або менш поширені вразливості. Організаціям із особливими вимогами до безпеки або додаткам із унікальними конфігураціями може знадобитися доповнити Nikto додатковими інструментами тестування або оцінками вручну;
- обмежена підтримка та документація: Ресурси підтримки та документація Nikto не такі великі, як у деяких комерційних інструментів. Хоча спільнота розробників надає допомогу через форуми та онлайн-ресурси, користувачі можуть зіткнутися з труднощами під час пошуку конкретних відповідей або рішень на свої запити;
- відсутність активної розробки. Незважаючи на те, що Nikto все ще широко використовується та надійний, останнім часом було порівняно менше оновлень порівняно з деякими іншими інструментами сканування вразливостей. Це може призвести до потенційних затримок у вирішенні нещодавно виявлених вразливостей або впровадженні нових функцій.

OWASP ZAP (Zed Attack Proxy) – ще один широко використовуваний інструмент тестування безпеки веб-програм із відкритим кодом [9].

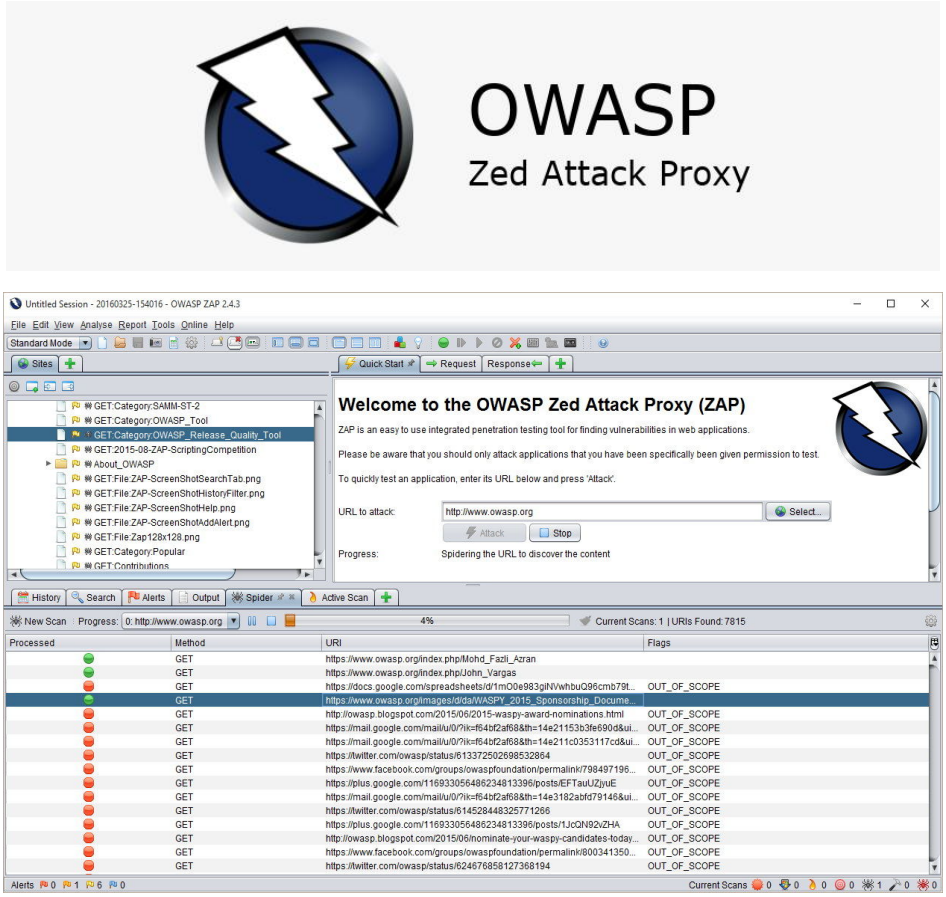

Рисунок 1.5 – Скріншоти застосунку OWASP ZAP

OWASP ZAP (Zed Attack Proxy) – це широко використовуваний інструмент тестування безпеки веб-додатків із відкритим кодом [9]. Він розроблений, щоб допомогти розробникам і фахівцям із безпеки визначати вразливості у веб-додатках і API. OWASP ZAP пропонує широкий спектр функцій як для ручного, так і для автоматичного тестування, що робить його популярним вибором для оцінки безпеки.

Переваги OWASP ZAP:

– відкритий і безкоштовний: OWASP ZAP – це інструмент з відкритим кодом, випущений за ліцензією Apache 2.0, що робить його вільно доступним для використання, модифікації та розповсюдження. Його природа з відкритим кодом сприяє внеску спільноти, постійному вдосконаленню та прозорості;

- активна спільнота OWASP: OWASP ZAP підтримується жвавою та активною спільнотою OWASP, яка забезпечує регулярні оновлення, виправлення помилок і включення нових функцій. Спільнота надає підтримку, ресурси та платформу для обміну знаннями між користувачами;
- широкі можливості сканування: OWASP ZAP підтримує широкий спектр тестів безпеки, включаючи сканування вразливостей, автоматичне сканування та ручне тестування. Він може ідентифікувати типові вразливості, такі як впровадження SQL, міжсайтовий сценарій (XSS), неправильні налаштування безпеки, незахищені прямі посилання на об’єкти тощо;
- зручний інтерфейс: OWASP ZAP надає зручний інтерфейс, який робить його доступним як для спеціалістів із безпеки, так і для розробників. Його інтуїтивно зрозумілий графічний інтерфейс дозволяє користувачам легко переміщатися між різними функціями та конфігураціями. Він також пропонує REST API, що забезпечує інтеграцію з іншими інструментами та робочими процесами;
- активне сканування та спайдерінг: OWASP ZAP включає активне сканування та спайдерінг, які автоматично досліджують і тестують веб-програми. Ці функції допомагають виявляти та аналізувати різні частини програми, визначати потенційні вразливості та надавати докладні звіти;
- сценарії та автоматизація: OWASP ZAP підтримує сценарії та автоматизацію, що дозволяє користувачам розширювати його функціональні можливості та налаштовувати свої тести безпеки. Користувачі можуть створювати власні сценарії або використовувати наявні для автоматизації повторюваних завдань, налаштування шаблонів сканування та інтеграції з робочими процесами тестування.

Недоліки OWASP ZAP:

- важкість навчання для початківців: Для досягнення максимальної ефективності потрібен певний рівень знайомства з концепціями веб-безпеки, методологіями тестування та особливостями OWASP ZAP;
- вплив на продуктивність. Виконання комплексного сканування безпеки за допомогою OWASP ZAP може споживати значні ресурси, особливо для великих програм або складних сценаріїв тестування. Користувачі повинні враховувати потенційний вплив на продуктивність цільової програми та відповідно виділяти достатні ресурси;
- хибно-позитивні та хибно-негативні результати: як і будь-який автоматизований інструмент тестування безпеки, OWASP ZAP може давати хибні спрацьовування або хибно-негативні результати;
- відсутність формальної підтримки: хоча OWASP ZAP отримує переваги від активної спільноти OWASP, він не надає офіційних каналів підтримки, як комерційні інструменти. Користувачі в основному покладаються на форуми спільноти, списки розсилки та онлайн-ресурси для підтримки. Отримання своєчасної спеціалізованої підтримки може вимагати додаткових зусиль або звернення за професійною допомогою;
- обмежене охоплення розширених уразливостей: OWASP ZAP покриває багато поширених уразливостей веб-додатків, але він може мати обмежене охоплення для більш складних або менш поширених проблем безпеки. Організаціям із особливими вимогами безпеки або унікальними конфігураціями додатків може знадобитися доповнити OWASP ZAP додатковими інструментами тестування або оцінками вручну.

SQLMap – це інструмент тестування на проникнення з відкритим кодом, який спеціалізується на виявленні та використанні вразливостей SQL-ін’єкцій у веб-додатках [10].

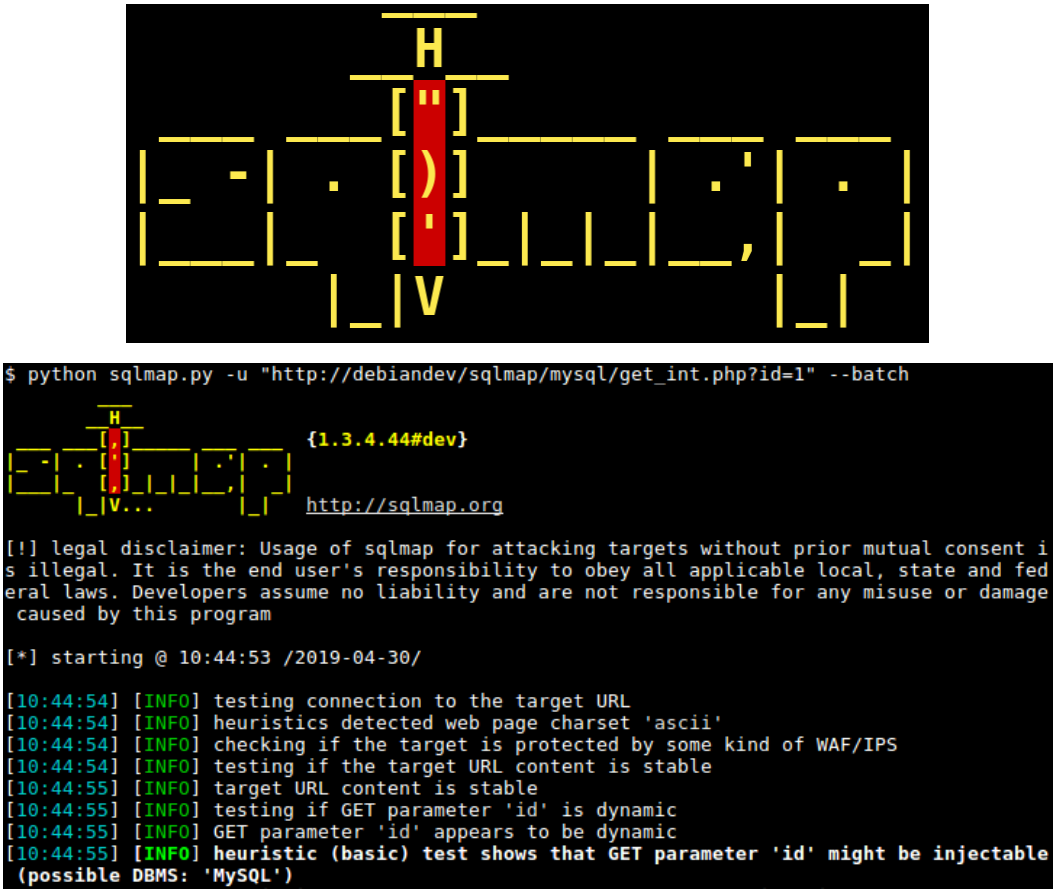

Рисунок 1.6 – Скріншоти застосунку SQLMap

Він автоматизує процес виявлення та використання недоліків впровадження SQL, дозволяючи фахівцям із безпеки ефективніше оцінювати безпеку веб-додатків.

Переваги SQLMap:

– комплексне тестування впровадження SQL: SQLMap спеціально розроблено для виявлення та використання вразливостей впровадження SQL у веб-додатках. Він виконує широкий спектр тестів і методів для виявлення та використання недоліків

впровадження SQL, допомагаючи фахівцям із безпеки точно оцінити стан безпеки веб-додатків;

- автоматизація та ефективність: SQLMap автоматизує процес тестування на вразливості SQL-ін’єкції. Він автоматично виконує різні тести та корисні навантаження, заощаджуючи час і зусилля порівняно з ручним тестуванням. Він також надає варіанти налаштування та розширені сценарії тестування, що дозволяє користувачам адаптувати процес тестування до своїх конкретних потреб;
- можливості використання: SQLMap не тільки визначає вразливості SQL-ін’єкцій, але й надає варіанти для використання цих уразливостей. Він може отримувати інформацію про схему бази даних, отримувати дані з баз даних і навіть виконувати команди операційної системи на базовому сервері. Ця функція допомагає власникам і розробникам додатків продемонструвати вплив і серйозність уразливостей впровадження SQL;
- підтримка кількох систем керування базами даних: SQLMap підтримує широкий спектр популярних систем керування базами даних (СУБД), включаючи MySQL, Oracle, PostgreSQL, Microsoft SQL Server та інші. Він коригує свої методи та корисне навантаження на основі цільової СУБД, забезпечуючи точне тестування та використання вразливостей SQL-ін’єкції на різних платформах;
- звітування: SQLMap створює детальні звіти та результати після завершення процесу тестування. Він надає інформацію про виявлені вразливості, використані недоліки, отримані дані тощо. Ці звіти можна використовувати для передачі результатів, визначення пріоритетності заходів з усунення проблем і допомоги в процесах управління вразливістю;

- активна спільнота та оновлення: SQLMap отримує переваги від активної спільноти користувачів і учасників. Інструмент регулярно оновлюється та вдосконалюється на основі відгуків користувачів і нових методів. Активна спільнота гарантує, що SQLMap залишається в курсі найновіших методів виявлення та використання SQL-ін’єкцій.

Недоліки SQLMap:

- обмежений обсяг: SQLMap зосереджується саме на вразливостях SQL-ін’єкцій. Хоча впровадження SQL є поширеною та критичною проблемою безпеки, це лише один аспект безпеки веб-додатків. Використання лише SQLMap може не забезпечити повну оцінку загального стану безпеки програми;
- складність навчання: SQLMap має інтерфейс командного рядка та численні параметри, які можуть ускладнити роботу для користувачів, які не знайомі з інструментами командного рядка чи технікою впровадження SQL. Для ефективного використання інструменту та інтерпретації його результатів може знадобитися деяке навчання та експерименти;
- можливість несанкціонованого доступу та пошкодження: хоча SQLMap розроблено для законного тестування безпеки, його можливості використання становлять ризик у разі використання без відповідного дозволу або неналежним чином. Щоб уникнути несанкціонованого доступу або ненавмисної шкоди, надзвичайно важливо використовувати SQLMap відповідально та з відповідним дозволом від власників програми.

3 РОЗРОБКА ТЕХНОЛОГІЇ ПЕРЕВІРКИ НА УРАЗЛИВОСТІ WEB-РЕСУРСІВ

3.1 Вибір мови програмування та технологій

Для розробки технології перевірки на уразливість Web-ресурсу була використана мова програмування Python з інтегрованим пакетом Anaconda.

Python – це популярна високорівнева мова програмування, яка відома своєю простотою та легкістю вивчення. Вона має широке застосування у різних сферах, включаючи веб-розробку, наукові обчислення, штучний інтелект, аналіз даних та багато іншого.

Anaconda, з іншого боку, є інтегрованим пакетом для Python, який спрощує установку та управління різноманітними пакетами та залежностями, необхідними для розробки програм на Python. Він включає в себе інсталятор Python, віртуальне середовище та набір популярних пакетів, таких як NumPy, Pandas, Matplotlib, SciPy та інші, які часто використовуються в наукових та аналітичних проектах. Основні переваги використання Anaconda для розробки на Python включають [11]:

- керування залежностями – Anaconda дозволяє легко встановлювати, оновлювати та керувати пакетами та залежностями Python, що забезпечує рівень сумісності та уникнення конфліктів між різними версіями пакетів;
- віртуальне середовище – ви можете створювати ізольовані віртуальні середовища для різних проектів, що дозволяє уникнути конфліктів між різними версіями пакетів та забезпечує чистоту установки для кожного проекту;
- популярні пакети – Anaconda поставляється з великою кількістю популярних пакетів та бібліотек для наукових обчислень та аналізу

даних, що спрощує їх використання і дозволяє швидше розпочати роботу над проектами;

- кросплатформеність – Anaconda підтримується на різних операційних системах, таких як Windows, macOS та Linux, що дозволяє розробникам працювати відповідно до своїх вподобань.

Загалом, Python з інтегрованим пакетом Anaconda надає зручне та потужне середовище для розробки на Python, особливо для наукових та аналітичних проектів, де вимагається використання різноманітних пакетів та залежностей.

Середою програмування була обрана PyCharm – це інтегроване середовище розробки (IDE) для мови програмування Python, розроблене компанією JetBrains. Воно пропонує широкий набір інструментів та функцій, спеціально розроблених для покращення продуктивності розробників Python. Основні особливості та переваги PyCharm включають:

- редактор коду – PyCharm має потужний редактор коду зі схемою розмітки, підсвічуванням синтаксису, автодоповненням, вбудованими інструментами для переходу до визначень функцій та багато іншого. Він підтримує також інспекцію коду та автоматичну корекцію помилок;
- розширена підтримка віртуальних середовищ – PyCharm дозволяє легко створювати та керувати віртуальними середовищами для ізоляції проектів та залежностей. Це допомагає уникнути конфліктів між різними версіями пакетів та забезпечує чистоту установки для кожного проекту;
- підтримка систем контролю версій – PyCharm інтегрується з популярними системами контролю версій, такими як Git, Mercurial, Subversion та іншими. Він надає можливості для комітів, синхронізації, перегляду історії змін та розв'язання конфліктів;

- відладка та профілювання – PyCharm надає інструменти для відладки коду, включаючи можливість встановлювати точки зупинки, крокування у коді, перегляду значень змінних та стеку викликів. Він також має функцію профілювання, яка допомагає виявляти та виправляти швидкісні проблеми в програмі;
- підтримка рефакторингу – PyCharm надає набір інструментів для автоматичного перейменування змінних, методів, класів та інших елементів коду. Він також допомагає виконувати рефакторинг коду для поліпшення структури та читабельності;
- інтеграція з іншими інструментами – PyCharm інтегрується з різними засобами тестування, включаючи популярні фреймворки, такі як unittest, pytest, nose та інші. Він також підтримує інструменти для аналізу коду, контролю якості та забезпечення виконання стандартів програмування.

PyCharm пропонує різні версії, включаючи безкоштовну відкриту версію Community та комерційну версію Professional з додатковими функціями та підтримкою [12].

В якості web-серверу були обрані такі модулі:

- Apache є одним з найпопулярніших веб-серверів, що доступних на сьогоднішній день. Він є вільним та відкритим програмним забезпеченням, розповсюджується під ліцензією Apache, і відомий своєю надійністю та стабільністю. Apache може обробляти запити HTTP від клієнтів, такі як веб-браузери, і надсилати їм сторінки та ресурси відповідно до налаштувань сервера. Він підтримує багато розширень та модулів, які дозволяють розширювати його функціональність, наприклад, встановлення SSL-шифрування, створення віртуальних хостів та багато іншого. Apache є основною складовою багатьох стеків веб-розробки, таких як LAMP (Linux,

Apache, MySQL, PHP/Python/Perl), і він широко використовується для розгортання веб-додатків та веб-сайтів.

– MySQL є однією з найпопулярніших відкритих реляційних систем управління базами даних (RDBMS). Він надає механізми для зберігання, організації та управління великими обсягами даних у структурованому форматі. MySQL пропонує мову запитів SQL (Structured Query Language), яку можна використовувати для створення, зміни та видалення даних у базі даних. Він підтримує велику кількість функцій, включаючи транзакції, індексацію, з'єднання таблиць, представлення та інші. MySQL часто використовується у веб-додатках, де потрібно зберігати та обробляти дані, таких як інформація користувачів, замовлення, коментарі тощо. Він також може бути частиною стеку LAMP для розробки веб-додатків.

У поєднанні, Apache та MySQL можуть використовуватись разом для розгортання та роботи з веб-додатками. Apache виконує роль веб-сервера, що обслуговує HTTP-запити від клієнтів, а MySQL - бази даних, де зберігаються та обробляються дані, необхідні для роботи додатків.

3.2 Проектування архітектури програмного модуля

Для проектування і розробки програмного забезпечення були обрані такі моделі:

– інтерактивна модель;
– операційна модель програми;
– архітектура програми;
– побудова тестових компонентів.

Інтерактивна модель орієнтована на взаємодію з користувачем та враховує його потреби та вимоги. Ця модель фокусується на розробці

ітераційних версій програмного продукту, де користувач може вносити зміни та віддається перевага швидкому отриманню зворотного зв'язку. Це дозволяє користувачам бачити результати в ранніх стадіях розробки і активно брати участь у процесі.

Операційна модель програми фокусується на описі, як програма буде взаємодіяти з операційною системою та зовнішніми компонентами. Ця модель описує структуру та поведінку програми, визначає потрібні операційні середовища, розміщення та конфігурацію компонентів програми.

Архітектура програми описує високорівневу структуру та організацію програмного продукту. Вона визначає основні компоненти, їх взаємозв'язок та взаємодію, а також забезпечує модульність, розширюваність та підтримуваність програми. Архітектура програми може бути представлена у вигляді діаграми компонентів, діаграми пакетів або іншого архітектурного опису.

Побудова тестових компонентів включає створення окремих модулів або компонентів програми, які будуть використовуватись для тестування. Це можуть бути тестові сценарії, функції для автоматичного тестування, інструменти для модульних або інтеграційних тестів. Побудова тестових компонентів допомагає перевірити правильність роботи програми, виявити та виправити помилки та забезпечити якість програмного забезпечення.

Ці моделі допомагають визначити структуру та поведінку програми, взаємодію з користувачем та забезпечити якість та надійність програмного продукту. Вони є важливими етапами у процесі проектування та розробки програмного забезпечення.

3.2.1 Інтерактивна модель

Використання інтерактивної моделі (рис. 3.1) дозволяє залучити користувачів та зацікавлені сторони в процес виявлення вразливостей, швидко

вносити зміни та отримувати зворотний зв'язок для покращення безпеки веб-ресурсу.

Використайте клієнтську програму, написану на Python, для перевірки вразливостей веб-сайту, підключивши сайт до тестового серверу. Клієнтська програма підключається до серверу, завантажує відповідну веб-сторінку і аналізує її для отримання всіх пов'язаних посилань. Потім виконує відповідну атаку на веб-сайт і отримує результат. Вона аналізує, чи містить веб-сайт дефекти, і повідомляє користувача [13].

Серверна програма отримує з'єднання від клієнта та відповідає на відповідні запити. Вона налаштована для тестування сервера, який містить будь-який веб-сайт, що може мати дефекти або не мати.

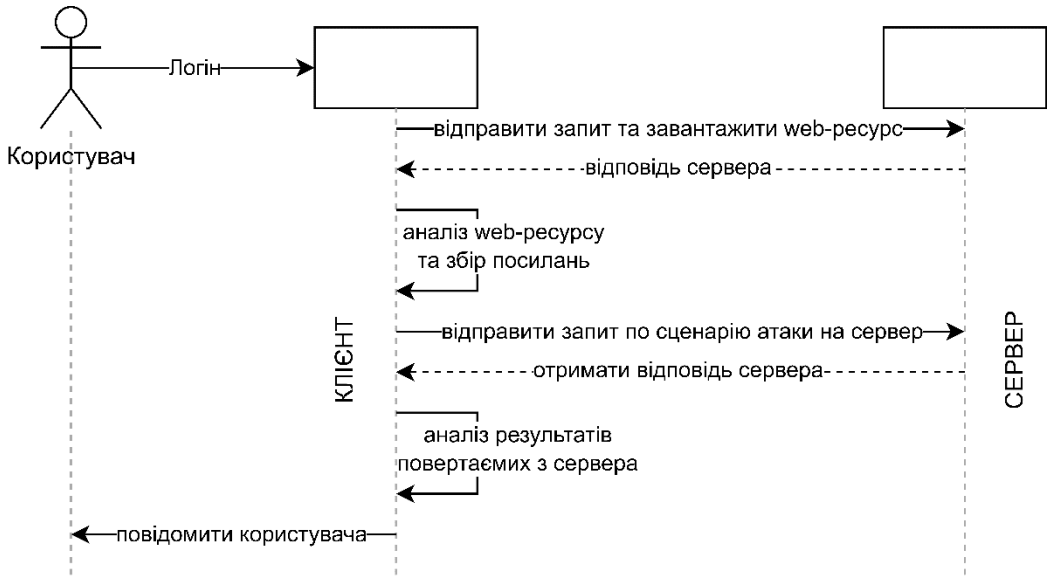

Рис. 3.1 – Схема процесу перевірки

3.2.2 Операційна модель програми

За допомогою аналізу інтерактивної моделі можемо отримати уявлення про продуктивність програми (рис. 3.2) в залежності від взаємодій користувачів з нею. Це дозволяє виявляти можливі проблеми та вчасно

вносити виправлення для поліпшення продуктивності програмного забезпечення.

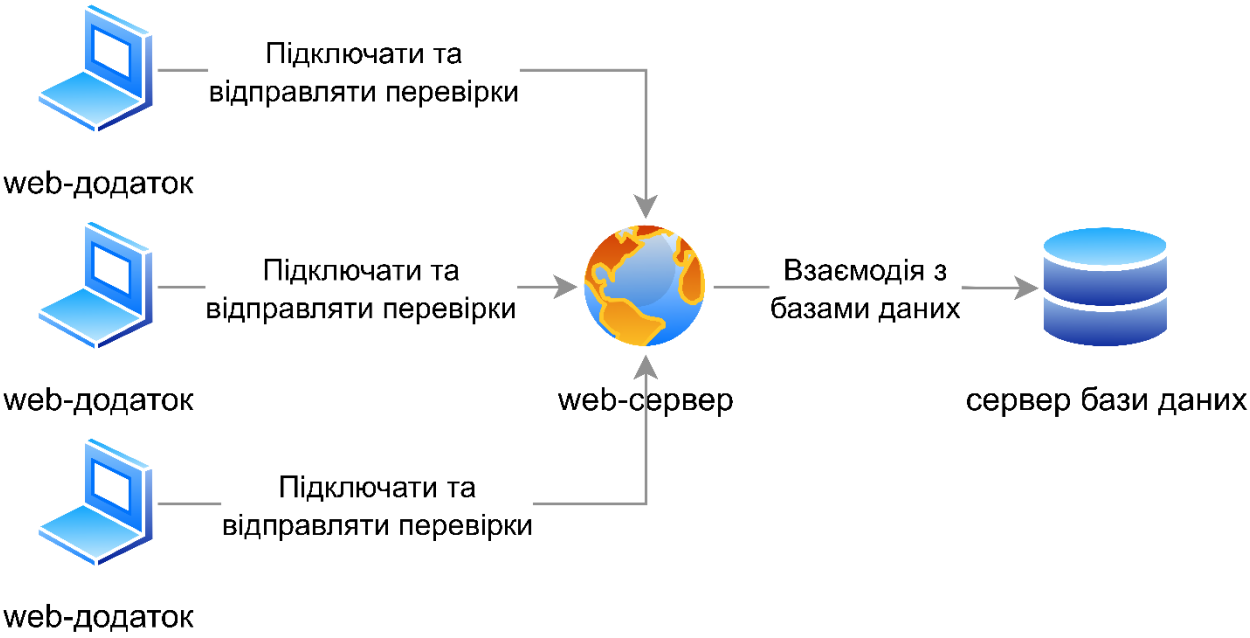

Рис. 3.2 – Модель взаємодії програми

Механізм сканування виконує з'єднання з сервером за допомогою посилання, наданого користувачем через параметр, і аналізує вхідні параметри для виявлення відповідного механізму сканування або атаки. Якщо параметри не використовуються, виконується сканування всіх компонентів.

На стороні сервера використовується веб-додаток, що містить певні мережеві вразливості, які можуть стосуватися бази даних або пов'язаних з нею вразливостей. Для перевірки вхідних даних та інших аспектів безпеки сканер може виявити і видаляти попередження, і програміст повинен перевірити безпеку для цього сайту.

Загалом, механізм сканування допомагає виявляти потенційні вразливості веб-додатків шляхом аналізу вхідних параметрів та інших елементів, що можуть впливати на безпеку системи. Це дозволяє програмістам

та адміністраторам вжити відповідних заходів для поліпшення безпеки і захисту веб-ресурсу.

3.2.3 Архітектура програми

Архітектура програми розробки програмного забезпечення для виявлення вразливостей веб-ресурсів (рис. 3.3) може мати наступні компоненти [14]:

- базова частина;
- модуль «Основа»;
- модуль «Атаки».

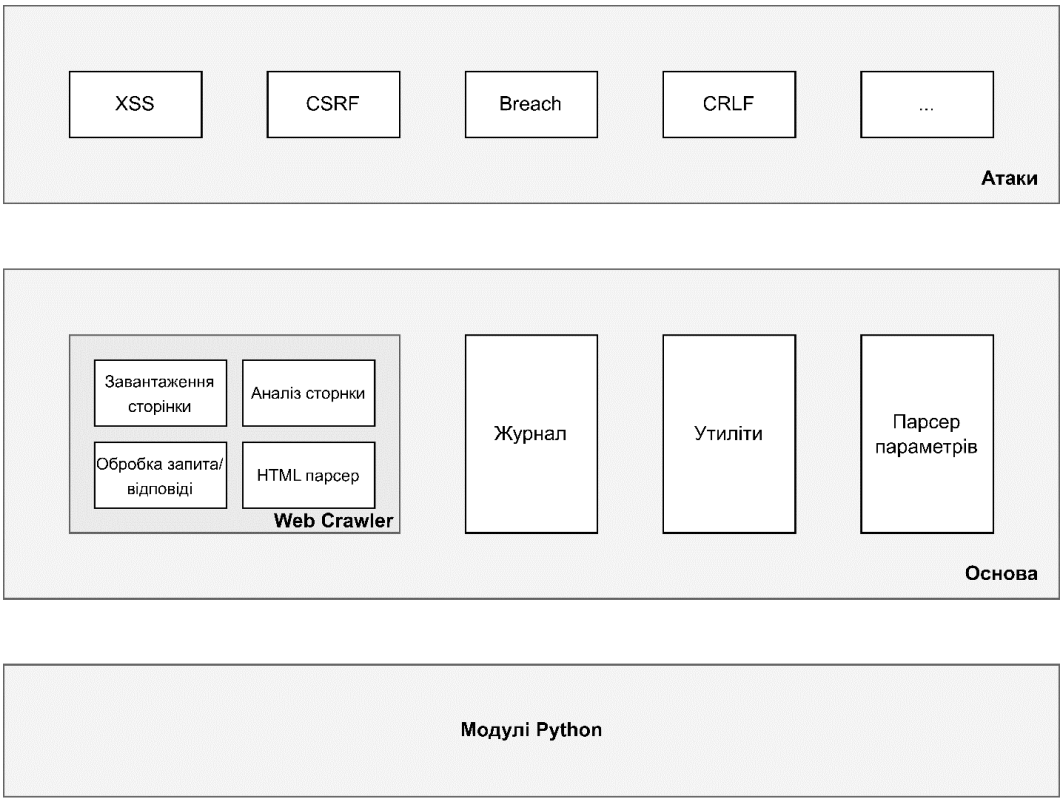

Рис. 3.3 – Архітектура програми вразливості web-сайту

Базова частина архітектури програми включає модулі мови програмування Python з відповідними бібліотеками.

Модуль «Основа» містить менші модулі, які виконують різні завдання. Наприклад, модуль збирає посилання на веб-сайт і перетворює їх в різні формати згідно з типовою структурою веб-сайту, такими як форми, текст, посилання, текстові поля і т.д. Модулі також надають інструменти для ведення журналу, параметрів аналізу та інших допоміжних функцій.

Модуль «Атаки» містить атаки, які виконують перевірки на веб-сайті. Він може включати атаки, такі як XSS, CSRF, Breach і Clickjack. Ці атаки використовуються для перевірки наявності вразливостей на веб-сайті та виявлення можливостей для зловживання.

Ці модулі допомагають програмі збирати і аналізувати інформацію з веб-сайту, виконувати атаки та перевіряти наявність вразливостей. Вони можуть бути розширені та налаштовані залежно від потреб проекту та вимог безпеки.

3.2.4 Побудова тестових компонентів

В модулі Attacks розпочинається розробка деяких компонентів атак для перевірки вразливостей веб-сайту:

- уразливість XSS;
- уразливість CSRF;
- уразливість Clickjacking;
- уразливість сканування файлів cookie;
- уразливість CRLF.

Уразливість XSS – цей компонент дозволяє впроваджувати шкідливі скрипти на веб-сторінку та перевіряти, чи вони виконуються коректно, що може призвести до витоку конфіденційної інформації або виконання шкідливого коду на браузері користувача.

Уразливість CSRF – цей компонент відправляє підставлені запити на веб-сайт в ім'я авторизованого користувача, що може призвести до несанкціонованого виконання дій від імені користувача.

Уразливість Clickjacking – цей компонент намагається визначити, чи існує можливість впроваджування стороннього контенту на веб-сторінку, який може бути використаний для обману користувача та отримання несанкціонованого доступу до даних.

Уразливість сканування файлів cookie – цей компонент аналізує механізм збереження та передачі файлів cookie і перевіряє наявність потенційних проблем безпеки, таких як витік конфіденційних даних або можливість підробки.

Уразливість CRLF – ця уразливість пов'язана з некоректною обробкою символів CRLF у введених даних, що може призвести до виконання несанкціонованих дій або зміни поведінки веб-сайту.

Ці компоненти атак допомагають виявляти різні типи вразливостей веб-сайту і дозволяють програмі проводити тестування безпеки для переконання у надійності та захищеності системи.

3.3 Результат та тестування

Для проведення тестування використовувався веб-сайт "Приклади Веб-вразливостей" (рис. 3.4). Це додаток, написаний на Python, який працює на порті 8666 і містить низку вразливостей для використання:

- перевірка авторизації на стороні клієнта;
- розширення довжини MAC;
- підробка запитів Cross-Site Request Forgery;
- вразливість відображення вразливого скрипту Cross-Site Scripting ;
- збережені вразливі скрипти Cross-Site Scripting;
- SQL Injection;

– обхід шляху.

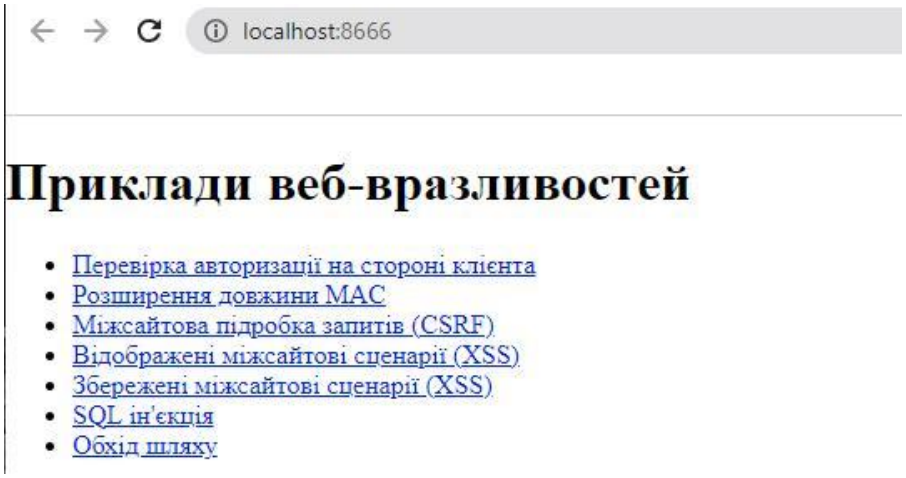

Рис. 3.4 – Інтерфейс web-сайту вразливостей

Перевірка авторизації на стороні клієнта – дозволяє обхід механізму авторизації і отримання несанкціонованого доступу до обмежених ресурсів або функцій.

Розширення довжини MAC – стосується криптографічних алгоритмів і може дозволити зловмиснику підмінити частину повідомлення, використовуючи вразливості в обробці MAC.

Підробка запитів CSRF – дозволяє зловмиснику виконувати несанкціоновані дії від імені авторизованого користувача, використовуючи підроблені запити [15].

Вразливість відображення вразливого скрипту XSS – дозволяє зловмиснику впроваджувати та виконувати шкідливий код на веб-сторінці, що може призвести до виконання небажаних дій у браузері користувача.

Збережені вразливі скрипти XSS – дозволяють зловмиснику впливати на вміст веб-сторінки, який зберігається на сервері, і виконувати шкідливі дії при відображенні цього вмісту.

SQL Injection – дозволяє зловмиснику впливати на базу даних, виконуючи шкідливі SQL-запити, що може призвести до витоку, зміни або видалення конфіденційної інформації.

Обхід шляху – дозволяє зловмиснику отримати доступ до обмежених файлів або ресурсів, обхідши механізми контролю доступу.

Ці уразливості були використані для проведення тестування програми і перевірки її здатності виявляти та реагувати на ці вразливості.

3.3.1 Сканування усіх вразливостей

Для сканування усіх уразливостей на веб-сайті, потрібно ввести команду: python webscan.py http://localhost:8666/.

Результатом даного сканування буде наступний текст:

```
It might be XSS vulnerability:
http://localhost:8666/reflected_xss/?username=%3Cscript%3Ealert%28%22XSS_STRING%22%29
%3B%3C%2Fscript%3E XSS with parameter “username”

It might be vulnerability CSRF: http://localhost:8666/csrf/send Lo hong CSRF

Summary report
Warning: http://localhost:8666/stored_xss/send HTML Error Encountered </script>,
expected
</body>

Warning: http://localhost:8666/stored_xss/ HTML Error Unclosed tag <form> (and 20
similar)
```

Висновком даного сканування можна зробити наступне:

– XSS (міжсайтовий скриптинг) уразливість:

Потенційна вразливість XSS за посиланням http://localhost:8666/reflected_xss/?username=%3Cscript%3Ealert%28%22XSS_STRING%22%29%3B%3C%2Fscript%3E з параметром "username". Це може вказувати на наявність уразливості XSS, яка може бути використана для виконання шкідливого коду на веб-сайті;

– CSRF (міжсайтова підробка запиту) уразливість:

Потенційна вразливість CSRF за посиланням http://localhost:8666/csrf/send. Це може вказувати на наявність уразливості CSRF, яка може дозволити зловмиснику виконати небажані дії в ім'я авторизованого користувача;

- попередження: http://localhost:8666/stored_xss/send – зустрінута помилка HTML </script>, очікувався </body>;
- попередження: http://localhost:8666/stored_xss/ – незакритий тег HTML <form> (і ще 20 схожих).

Ці результати вказують на можливі вразливості веб-сайту, пов'язані з XSS, CSRF та некоректним відображенням HTML. Рекомендується вжити відповідних заходів для усунення цих вразливостей, щоб забезпечити безпеку веб-сайту.

3.3.2 Сканування вразливостей XSS

Для сканування уразливостей XSS на веб-сайті, потрібно ввести команду: python webscan --xss http://localhost:8666/.

Результатом даного сканування буде наступний текст:

```
It might be XSS vulnerability:
http://localhost:8666/reflected_xss/?username=%3Cscript%3Ealert%28%22XSS_STRING%22%29
%3B%3C%2Fscript%3E XSS with parameter “username”

Summary report
Warning: http://localhost:8666/stored_xss/send HTML Error Encountered </script>,
expected
</body>

Warning: http://localhost:8666/stored_xss/ HTML Error Unclosed tag <form> (and 4
similar)
```

Висновком даного сканування можна зробити наступне:

- вразливість XSS за посиланням http://localhost:8666/reflected_xss/?username=%3Cscript%3Ealert%28%22XSS_STRING%22%29%3B%3C%2Fscript%3E з параметром

"username". Це може вказувати на наявність уразливості XSS, яка може бути використана для виконання шкідливого коду на веб-сайті;

- попередження: http://localhost:8666/stored_xss/send – виявлено помилку HTML </script>, очікувався </body>;
- попередження: http://localhost:8666/stored_xss/ – незакритий тег HTML <form> (і ще 4 схожих).

Ці результати вказують на можливі уразливості веб-сайту, пов'язані з XSS та некоректним відображенням HTML. Рекомендується вжити відповідних заходів для усунення цих вразливостей та забезпечення безпеки веб-сайту.

3.3.3 Сканування вразливостей CSRF

Для сканування уразливостей CSRF на веб-сайті, потрібно ввести команду: python webscan --csrf http://localhost:8666/.

Результатом даного сканування буде наступний текст:

```
It might be CSRF vulnerability: http://localhost:8666/csrf/send -- CSRF vulnerability

Summary report
Warning: http://localhost:8666/stored_xss/ HTML Error Unclosed tag <form> (and 3 similar)
```

Висновком даного сканування можна зробити наступне:

- вразливість CSRF за посиланням http://localhost:8666/csrf/send. Це може вказувати на наявність уразливості CSRF, яка дозволяє зловмиснику виконувати небажані дії в ім'я авторизованого користувача;
- попередження: http://localhost:8666/stored_xss/ – незакритий тег HTML <form> (і ще 3 схожих).

Ці результати вказують на можливі вразливості веб-сайту, пов'язані з CSRF та некоректним відображенням HTML. Рекомендується вжити

відповідних заходів для усунення цих вразливостей і забезпечення безпеки веб-сайту.

3.3.4 Виявлення вразливості Clickjacking

Для виявлення вразливості Clickjacking на веб-сайті, потрібно ввести команду: python webscan --clickjack http://localhost:8666/.

Результатом даного сканування буде наступний текст:

```
Summary report:
Warning: http://localhost:8666/stored_xss/ HTML Error Unclosed tag <form>

Warning: http://localhost:8666/stored_xss/?username=Benutzer%21 HTML Error Unclosed tag
<form>
```

Висновком даного сканування можна зробити наступне:

- попередження: http://localhost:8666/stored_xss/ – зустрінуто помилку HTML з незакритим тегом <form>;
- попередження: http://localhost:8666/stored_xss/?username=Benutzer%21 – зустрінуто помилку HTML з незакритим тегом <form>.

Ці результати вказують на можливі вразливості веб-сайту, пов'язані з CSRF та некоректним відображенням HTML. Рекомендується вжити відповідних заходів для усунення цих вразливостей і забезпечення безпеки веб-сайту.

На підставі проробленої роботи можна зробити такий висновок. Веб-сайт "Приклади Веб-вразливостей", що був використаний для тестування, виявив ряд уразливостей, таких як XSS, CSRF та помилки HTML. Було успішно виконано сканування веб-сайту за допомогою розробленої програми на Python, що дозволило виявити потенційні вразливості та помилки веб-сайту. В цілому, проробленa робота демонструє важливість виявлення та виправлення уразливостей веб-ресурсів для забезпечення безпеки та захисту інформації.

ВИСНОВКИ

У ході дипломної роботи було розроблено технологію дослідження на уразливість веб-ресурсів з метою виявлення потенційних вразливостей та запобігання можливим атакам. Дослідження на уразливість веб-додатків є важливим завданням у забезпеченні безпеки інтернет-ресурсів і захисту конфіденційної інформації.

В роботі було розглянуте поняття веб-ресурсів та здійснено огляд типів та технік перевірки їх на уразливості. Інструменти автоматичного тестування вразливостей призначені для сканування систем, мереж і програм на наявність уразливостей без ручного втручання. Популярними інструментами тестування вразливостей є Nessus, OpenVAS, Burp Suite, OWASP ZAP та SQLMap. В роботі було наведені переваги та недоліки кожного з цих інструментів.

Для розробки власної технології перевірки на уразливість веб-ресурсів була використана мова програмування Python з інтегрованим пакетом Anaconda. Середою програмування було обране PyCharm – інтегроване середовище розробки (IDE) для мови програмування Python, розроблене компанією JetBrains.

В результаті вдалого застосування розробленої програми на мові Python було проведено успішне сканування веб-сайту. Це дозволило виявити потенційні вразливості та помилки на веб-сайті.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Різновиди веб-систем та їх основні вразливості. URL: http://websecurity.com.ua/security/chapter1 (дата звернення 03.05.2023)
2. Захист веб-додатків: чому це важливо? URL: https://itbiz.ua/statti-ta-obzori/zaxist-veb-dodatkiv-chomu-ce-vazhlivo (дата звернення 03.05.2023)
3. Тест на проникнення. Вікіпедія. URL: https://uk.wikipedia.org/wiki/Тест_на_проникнення (дата звернення 05.05.2023)
4. Стеценко, І. В., & Савчук, В. В. (2021). Метод автоматизації тестування на проникнення веб-атак. Технічні науки та технології.
5. Nessus: Malware and Vulnerability Assessment. URL: https://techexpert.ua/ru/products/nessus (дата звернення 29.04.2023)
6. Greenbone OpenVAS. URL: https://openvas.org (дата звернення 08.05.2023)
7. Burp Suite. URL: https://portswigger.net/burp (дата звернення 10.05.2023)
8. CIRT.net. Suspicion Breeds Confidence. URL: https://cirt.net/Nikto2 (дата звернення 10.05.2023)
9. OWASP Zed Attack Proxy. URL: https://www.zaproxy.org (дата звернення 11.05.2023)
10. sqlmap. URL: https://sqlmap.org (дата звернення 11.05.2023)
11. Елізабет, Ф. Створення набору тестів для сканерів веб-додатків. Елізабет Ф., Ромен Г., Пол Б.
12. Емре, Е. Веб-сканери вразливостей: практичне дослідження. Емре, Е., Енджел, Р. 2017.
13. Фонсека, Дж., Вієйра, М., і Мадейра, Х. (2014). Оцінка механізмів веб-безпеки з використанням вразливості та впровадження атак. Надійні та безпечні обчислення, IEEE Transactions on, 11(5), 440-453.

14. Кіннерд, Мак. Веб-сканери вразливостей із відкритим вихідним кодом: економічно ефективний вибір? 2014 Матеріали конференції прикладних досліджень інформаційних систем, Балтімор, Меріленд, США. ISSN: 2167-1508.
15. М. Парвез, Р. Заварський і N. Khoury, «Аналіз ефективності сканерів веб-додатків чорної скриньки у виявленні збережених SQL-ін’єкцій і збережених уразливостей XSS», 2015, 10-та Міжнародна конференція з Інтернет-технологій і безпечних транзакцій (ICITST), Лондон , 2015, С. 186-191. doi: 10.1109/ICITST.2015.7412085