**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ**

Факультет Магістерської підготовки

Кафедра Інформаційних технологій

# МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему: Програмні засоби пошуку уразливостей в веб ресурсах

Виконав студент 2 курсу групи МІС-19
спеціальності 122 Комп'ютерні науки

Сагуйченко Олексій Володимирович

Керівник д.т.н., професор
Казакова Надія Феліксівна

Рецензент засновник Компанії «UALinux»
Попов Володимир Леонідович

Одеса 2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет _Магістерської підготовки_____
Кафедра _Інформаційних технологій_____
Рівень вищої освіти _магістр_____
Спеціальність 122 Комп'ютерні науки_____
(шифр і назва)

**ЗАТВЕРДЖУЮ**
**Завідувач кафедри**_____
_____
" 26 " жовтня 2020 р.

# З А В Д А Н Н Я
## НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

_____Сагуйченку Олексію Володимировичу_____
(прізвище, ім'я, по батькові)

1. Тема роботи __«Програмні засоби пошуку уразливостей в веб ресурсах»__
_____
_____

керівник роботи __Казакова Надія Феліксівна, д.т.н., професор_____,
( прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від " 16 " жовтня № 194 «с»

2. Строк подання студентом роботи 7 грудня 2020р.

3. Вихідні дані до роботи 1. Огляд уразливостей web-ресурів_____
__2. Аналіз методів та засобів виявлення уразливістей системи._____
__3.Існуючі продукти для пошуку уразливостей_____
__4. Проведення тестування на уразливість._____
__5. Рекомендації про підвищення захищеності від уразливостей._____

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)_____
__1. Огляд уразливостей Web-Ресурсів._____
__2. Існуючі продукти для пошуку уразливостей_____
__3. Засоби для проведення тестування та робота з OWASP ZAP_____
__4. Проведення тестування на уразливість_____
__5. Рекомендації для підвищення захищеності від уразливостей_____
_____

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
____Слайди презентації_____
_____
_____

6. Консультанти розділів роботи

| Розділ | Прізвище, ініціали та посада консультанта | Підпис, дата | |
|---|---|---|---|
| | | завдання видав | завдання прийняв |
| | | | |
| | | | |
| | | | |
| | | | |

7. Дата видачі завдання « 26 » жовтня 2020 р.

## КАЛЕНДАРНИЙ ПЛАН

| № з/п | Назва етапів дипломної роботи | Термін виконання етапів роботи | Оцінка виконання етапу | |
|---|---|---|---|---|
| | | | у % | за 4-х бальною шкалою |
| 1. | Огляд уразливостей web-ресурів | **26.10.2020** | **70** | **задов.** |
| 2. | Аналіз методів та засобів виявлення уразливостей системи | 30.10.2020 | 70 | задов. |
| 3. | Існуючі продукти для пошуку уразливостей | 4.10.2020 | 70 | задов. |
| 4. | Опис популярних методологій та стандартів тестування на проникнення | 10.10.2020 | 70 | задов. |
| 5. | Порівняння програмних продуктів тестування | 12.10.2020 | 70 | задов. |
| | Рубіжна атестація | **19.11.2020** | **70** | **задов.** |
| 6. | Проведення тестування на уразливість ОС | 21.11.2020 | 70 | задов. |
| 7. | Дослідження мережі та перевірки безпеки за допомогою OWASP ZAP | 22.11.2020 | 70 | задов. |
| 8. | Проведення тестування на уразливість | 25.11.2020 | 70 | задов. |
| 9. | Міри для підвищення захищеності від уразливостей | 3.12.2020 | 70 | задов. |
| | | | | |
| | Подання роботи на кафедру | **07.12.2020** | **вик** | |
| | Перевірка на плагіат | **08.12.2020** | **вик** | |
| | Рецензування | **16.12.2020** | **вик** | |
| | | | | |
| | **Інтегральна оцінка виконання етапів календарного плану (як середня по етапам)** | | **70** | **задов.** |

**Студент** _____ **Сагуйченко О.В.**
(підпис) (прізвище та ініціали)

**Керівник роботи** _____ **Казакова Н.Ф.**
(підпис) (прізвище та ініціали)

# АНОТАЦІЯ

на магістерську кваліфікаційну роботу

«Програмні засоби пошуку уразливостей в веб ресурсах»,

студента Сагуйченко Олексій Володимировича

Актуальність теми магістерської кваліфікаційної роботи обумовлюється необхідністю підвищення рівня захищеності доступу до веб русурсу управління, в якому циркулює інформація. В першу чергу це стосується персональних даних, та іншої інформації, необхідність захисту якої регламентована законодавчими та нормативними документами України.

Дипломна робота присвячена аналізу засобів пошуку уразливостей Web-сайтів та серверів. Було розглянуто поширені уразливості Web-сайтів та серверів, обрано програмні засоби для їх сканування. Після цього було розглянуто їх інтерфейс та функціональні можливості.

Виконаний ретельно проведений тест існуючих уразливостей двох веб-ресурсів , створений звіт цих проблем та сформульована рекомендація щодо їх уникнення.

Магістерська кваліфікаційна робота містить 76 сторінки, 35 рисунків, 2 таблиці та 18 джерел.

Ключові слова: АНАЛІЗ, ВЕБ-РЕСУРС, ЕКСПЛОЙТ, ЕТИЧНИЙ ХАКІНГ, МЕТОДИ ПРОНИКНЕННЯ, МЕРЕЖЕВЕ СКАНУВАННЯ, ПЕНТЕСТИНГ, ТЕСТУВАННЯ НА ПРОНИКНЕННЯ, УРАЗЛИВІСТЬ.

**SUMMARY**

for a master's degree

" Software tools for finding vulnerabilities in web resources",

student Saguichenko Alexey Vladimirovich

The relevance of the topic of the master's qualification work is due to the need to increase the level of security of access to the automated management system of the enterprise in which information circulates. First of all, this applies to personal data and other information, the need for protection of which is regulated by laws and regulations of Ukraine.

Thesis is devoted to the analysis of vulnerabilities for Web-sites and servers. Common vulnerabilities in Web sites and servers were examined, and software for scanning them was selected. After that, their interface and functionality were considered.

A thorough test of the existing vulnerabilities of the two web resources was performed, a report on these problems was created and a recommendation was made to avoid them.

Master's thesis contains a 76 page, 35 figures, 2 tables and 18 sources.

Keywords: ANALYSIS, ACS, EXPLOIT, ETHICAL HACKING, NETWORK SCANNING, PENTESTING, PENETRATION TESTING, VULNERABILITY.

**ЗМІСТ**

# СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

Бекдор (backdoor) – програми прихованого віддаленого адміністрування, які надають шахраям можливість несанкціонованого і віддаленого управління скомпрометованим комп’ютером.

Брутфорс (Brute force) – називається метод злому різних облікових записів, шляхом підбору логіна і пароля.

Експлойт – шкідливий код, який використовує помилки або недоліки системи безпеки для поширення кіберзагроз.

Етичний (білий) хакінг – це законна форма злому, за допомогою якого можна знайти уразливості / помилки в чужих системах на добровільній основі або ж за плату з метою допомогти розробнику зробити продукт більш захищеним.

Нативні додатки (англ. Native app (lication) s) – це прикладні програми, які були розроблені для використання на певній платформі або на певному пристрої.

Пентестинг (тест на проникнення) – метод оцінювання захищеності комп'ютерної системи чи мережі шляхом часткового моделювання дій зовнішніх зловмисників з проникнення у неї (які не мають авторизованих засобів доступу до системи) і внутрішніх зловмисників (які мають певний рівень санкціонованого доступу).

Система безпеки – це сучасні надійні засоби технічного захисту будь-якого об’єкта.

Троян – шкідливе програмне забезпечення, яке приховує істинну ціль своєї діяльності за допомогою маскування (на відміну від вірусу, він не здатний самостійно копіювати чи заражати файли).

Уразливість – нездатність системи протистояти реалізації певної загрози або сукупності загроз.

Фішинг – це схема, за якої хакери змушують користувачів передавати конфіденційну інформацію, наприклад паролі та номери соціального страхування.

AJAX – підхід до побудови користувацьких інтерфейсів веб-застосунків, за яких веб-сторінка, не перезавантажуючись, у фоновому режимі надсилає запити на сервер і сама звідти довантажує потрібні користувачу дані

JavaScript – це мова програмування, що дозволяє зробити Web-сторінку інтерактивною, тобто такою що реагує на дії користувача; об'єктно-орієнтована скриптова мова програмування і є діалектом мови ECMAScript.

jQuery – це JavaScript бібліотека; набір готових функцій, які спрощують написання коду на JavaScript розробнику.

Metasploit Framework – це найбільш відомий інструмент для створення, тестування і використання експлойтів.

NSE (Nmap Scripting Engine / Скриптова движок Nmap) – використовується для отримання розширеної інформації про запущені сервіси.

Root – головний обліковий запис в системах UNIX.

Root-доступ – привілейований користувач, що має повний контроль над операційною системою не лише виділеного сервера, але і віртуального сервера VPS.

UNIX – це сімейство багатозадачних і багатокористувацьких операційних систем, які засновані на ідеях оригінального проекту AT&T.

ЗВО – заклад вищої освіти.

ІТ – інформаційні технології.

ОС – операційна система.

ПЛК – контролери з програмованої логікою.

ПС – підсистема.

AJAX – Asynchronous Javascript and XML.

API – Application Programming Interface.

ISSAF – Information System Security Assessment Framework.

NIST – National Institute of Standards and Technology.

NSE – Nmap Scripting Engine.

OSSTMM – Open Source Security Testing Methodology Manual.

OWASP – Open Web Application Security Project.

PTES – Penetration Testing Methodologies and Standards.

SQL – Structured Query Language.

WAF – Web Application Firewall.

# ВСТУП

Web-ресурси – об’єкти котрі постійно піддаються загрозам. Серйозну загрозу для них створюють хакери та віруси. Хакери можуть отримати необхідні данні для доступу для інформації, зазвичай конфіденційної, або навпаки змінити її. У їх арсеналі також є можливість виведення з ладу самого сервера що критично важливий. Віруси що вражають веб-ресурси можуть перетворити їх у носій інфекції. Такі віруси використовуються для поширення їх у мережі між користувачами для отримання даних що не існують на сервері.

Кожна програма має свої уразливості. Це дуже легко пояснюється тим що людина скоює помилки. При створенні великого програмного забезпечення використовують великі ресурси людей, різні люди створюють певні частини програм і це є підставою номер один для виникнення дразливостей у наш час.

Якщо дивитися на сьогоднішній день компанії стали замислюватися над безпекою свої інформаційних мереж через кількість атак. Але як показує статистика кількість атак лише збільшуються. Кількість кібер атак у першому кварталі 2020 року збільшилась на 22.5%, якщо порівнювати з четвертим кварталом 2019 року.

Щоб знайти та усунути веб вразливість, були створенні спеціальні веб сканери. Веб сканер – програмний або апаратний веб засіб, що сканує систему на предмет уразливостей і дозволяє виявити і оцінювати уразливості мережі.

Сканери уразливості діляться на дві основні групи:

1. Сканери корпоративних мереж, призначення яких полягає в аналізі мережі на наявність відкритих портів, а також уразливостей в операційних системах і додатках.

2. Сканери уразливості веб-додатків. На даний момент їхня популярність зростає в силу того, що більшість комерційних організацій і банків використовують у своїй діяльності інтернет ресурси, захист яких стає

важливим фактором. У цій роботі буде розглянуто більше інформації саме по цій групі.

Метою роботи є огляд і оцінка можливостей сучасних засобів пошуку уразливостей. Кінцевою метою є збір та аналіз результатів, наведенння шляхів боротьби зі знайденими вразливастями за допомогою обраних засобів.

# 1 ОГЛЯД УРАЗЛИВОСТЕЙ WEB-РЕСУРІВ.

Для того, щоб максимально використовувати переваги веб-технологій, необхідно забезпечити доступність доступності ресурсів для цільової аудиторії, наприклад із мережі Інтернет. Немає доступу, відповідно, може отримати та зловмисник. Це і недобросовестные конкуренція, та інші категорії порушників, керівництва переступними намірениями, наприклад з метою хіщення грошових коштів, порушення доступності ресурсу або отримання чутливої інформації. Компрометація додатків може приносити як репутаційним втратам, так і фінансовим, у тому числі в відео прибули (наприклад, якщо буде втрачений важливий клієнт або сорвется угода).

При всіх при цьому розробники не завжди забезпечують достатньо уваги захищених веб-ресурси, сосредоточиваясь в першу чергу на функціональність додатків; адміністратори системи зачастують недостатньо освічені в питаннях інформаційної безпеки [1][1)] та можуть вчинити помилки, які роблять додатки уявними. Подаючи більшість власників веб-ресурсів, не слід застосовувати принципи забезпечення безпеки на всіх етапах життєвого цикла, що додається (безпечне програмне забезпечення життєвий цикл розробки, SSDL), в результаті чого уявлення не виявляються на ранніх стадіях розробки, а залишаються в додатках навіть після їх прийому в експлуатації, що грає на руку зловмисникам.

## 1.1 Уразливості ПЗ

Якщо йде мова про уразливості веб-ресурсів, то як правило мають на увазі дірки в програмному забезпеченні. Програмне забезпечення серверів досить складне і об’ємне, тому дірки в ньому є обов’язково. Такими великими програмними забезпеченнями можуть виступати  Apache, Internet Inforamation

1) [1] Актуальные киберугрози: 1 квартал 220 года. URL: https://www.ptsecurity.com/ru-ru/research/analytics/cybersecurity-threatscape-2020-q1/ (дата звернення 11.08.2020).

Server та інш.. Сучасний веб сервер обов’язково підтримує PHP, а також системи управління базами даних (наприклад MY SQL). Усе це є доступним при установці серверного програмного забезпечення, яке у свою чергу також має уразливості.

На сьогоднішній день усе програмне забезпечення має свою лінійку версій, і дірки зустрічаються не у всій лінійці таких програм, а тільки в деяких. Хоча і для тих версій що були випробувані часом можуть бути знайдені нові уразливості.

Великою сучасною перевагою є можливість установки нових патчів програми для її оновлення. Самі розробники спостерігають за виникшими дірками та багами, що мають бути усуненні. Адже при виявленні критичних для безпеки уразливостей необхідно швидко випускати оновлення.

### 1.1.1 Уразливості обладнання

Уразливості можуть виникати не лише через зовнішні загрози але і через неправильне налаштування обладнання. Сучасні сервери мають великий набір інструментів та параметрів для їх налаштування. Виходячи з цього, безпека залежить від досвіду та знань адміністраторі що обслуговують такі веб сервера. Досвід та кваліфікація адміністратора можуть зіграти важливу роль при атаці на ресурс [2][1)].

Зазвичай при оновленні програмного забезпечення усі налаштування зберігаються, тому неправильне його перше налаштування може бути непомітною вразливістю. Це лише одни із прикладів уразливостей що демонструє складність їх виявлення.

---

1) [2] Уязвимости сайтов. URL: https://www.anti-malware.ru/threats/site-vulnerability (дата звернення 12.08.2020)

### 1.1.2 Уразливості внутрішнього ПЗ

Під внутрішнім ПЗ я маю на увазі написаний код власноруч командою що займається веб-ресурсом. Наприклад скрипти веб-сайта можуть містити уразливості. Небезпека полягає в тому що більшість скриптів виконуються на самих веб серверах, а не на комп'ютерах користувачів. Та такі скрипти не завжди розроблюються гарними спеціалістами. Якщо такий софт має помилки, то наслідки можуть бути дуже серйозними, наприклад зловмисники можуть дістати доступ до самого серверу. Виявити такі дірки в скриптах можна за допомогою сканерів безпеки. Якщо слідкувати за безпекою свого сайту, то слід періодично перевіряти такі дірки. Крім того з часом знаходять нові уразливості котрі можуть бути з'єднанні зі скриптами, тобто навіть якщо раніше скрипт вважався безпечним, зараз він уявляти загрозу.

## 1.2 Класифікація уразливостей веб-ресурсу

Для більшості досліджених веб-серверів самої поширеної помилки адміністрування є «Утечка інформації»[3][1)] (витік інформації). Даний недолік був виявлений у всіх досліджених додатках під управлінням серверів Microsoft IIS. Друге місце рейтингу в 2015 році для більшості серверів зайнято недостаток «Відсутнє захист від підбору навчальних даних» (Brute Force), який став найпоширенішою уявною уявою у додатках під управлінням веб-сервера Nginx. Розроблення інформації про версію використовуваного ПО (відбитків пальців) для більшість веб-серверів розташовані лише на третьому місці за поширеністю, а у додатках під управлінням сервера WebLogic не було виявлено вовсе.

---

[1)] [3] Внедрение команд ОС. URL: https://hackware.ru/?p=1133(дата звернення 25.08.2020)

### 1.2.1 XSS

XSS (міжсайтовий скриптинг) - один з різновидів атак на веб-системи, яка має на увазі впровадження шкідливого коду на певну сторінку сайту і взаємодія цього коду з віддаленим сервером зловмисників при відкритті сторінки користувачем. Приклад зворотної дії системи зображено на рисунку.

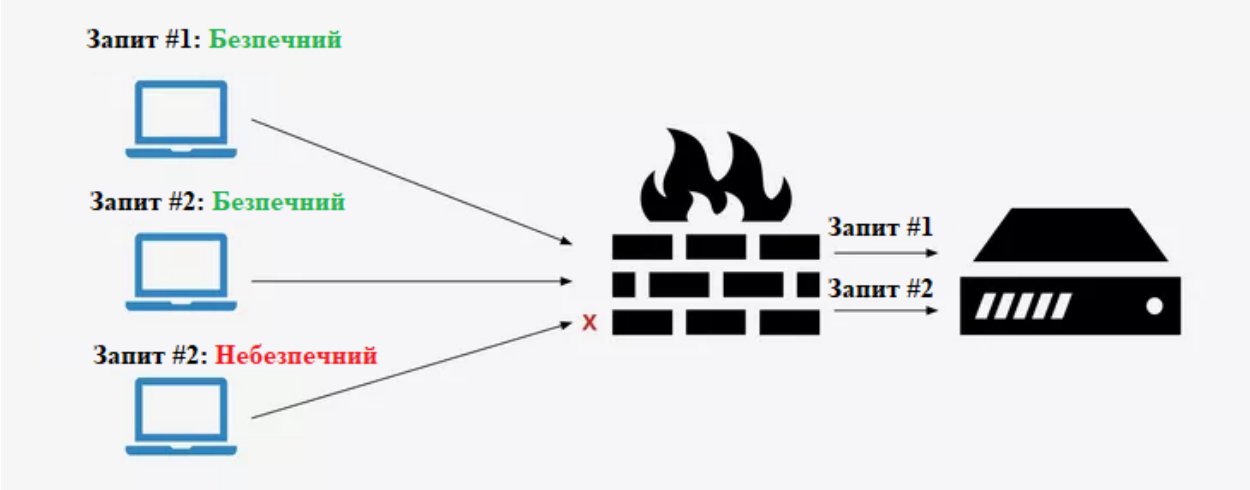



Рисунок 1 – Зворотня проти водія системи на загрозу

Основна мета міжсайтового скриптинга - крадіжка cookies[2][1)] користувачів за допомогою вбудованого на сервері скрипта з подальшою вибіркою необхідних даних і використанням їх для наступних атак і зломів. Зловмисник здійснює атаку користувачів не безпосередньо, а з використанням уразливостей веб-сайту, який відвідують жертви, і впроваджує спеціальний JavaScript. У браузері у користувачів цей код відображається як єдина частина сайту. При цьому відвідуваний ресурс за фактом є співучасником XSS-атаки.

Якщо порівнювати з SQL-ін'єкціями, то XSS безпечний для сервера, але несе загрозу для користувачів зараженого ресурсу або сторінки. Однак, якщо до зловмисника потраплять cookies адміністратора, можна отримати доступ до панелі управління сайтом і його вмісту.

---

1) [2] Уязвимости сайтов. URL: https://www.anti-malware.ru/threats/site-vulnerability (дата звернення 12.08.2020)

Запуск шкідливого коду JavaScript можливий тільки в браузері жертви, тому сайт, на який зайде користувач, повинен мати вразливість до XSS. Для здійснення атаки зловмисник спочатку перевіряє ресурси на наявність уразливостей через XSS, використовуючи автоматизовані скрипти або ручний режим пошуку. Зазвичай це стандартні форми, які можуть відправляти і приймати запити (коментарі, пошук, зворотний зв'язок).

Для пошуку «дірок» на сайті існує величезна кількість готових скриптів і запитів, і якщо жоден з них не підходить, значить ресурс надійно захищений від подібних атак.

Класифікація XSS:

- Збережені XSS (постійні).
- Відображені XSS (непостійні).
- DOM-моделі.

**Збережені XSS** (постійні). Один з найнебезпечніших типів уразливостей, так як дозволяє зловмисникові отримати доступ до сервера і вже з нього управляти шкідливим кодом (видаляти, модифікувати). Кожен раз при зверненні до сайту виконується заздалегідь завантажений код, який працює в автоматичному режимі. В основному таким уязвимостям схильні форуми, портали, блоги, де присутня можливість коментування в HTML без обмежень. Шкідливі скрипти з легкістю можуть бути вбудовані як в текст, так і в картинки, малюнки.

**Відображені XSS** (непостійні). В цьому випадку шкідлива рядок виступає в ролі запиту жертви до зараженого веб-сайту. Працює цей принцип за наступною схемою:

1. Зловмисник заздалегідь створює URL-посилання, яка буде містити шкідливий код і відправляє його своїй жертві.
2. Вона спрямовує цей URL-запит на сайт (переходить за посиланням).
3. Сайт автоматично бере дані з шкідливою рядки і підставляє у вигляді модифікованого URL-відповіді жертві.

4. У підсумку в браузері у жертви виконується шкідливий скрипт, який і міститься у відповіді, а зловмисник отримує все cookies цього користувача.

**DOM-моделі**. У цьому варіанті можливе використання як збережених XSS, так і відображених. Суть полягає в наступному:

1. Зловмисник створює URL-адресу, який заздалегідь містить шкідливий код, і відправляє його по електронній пошті або будь-яким іншим способом користувачеві.
2. Людина переходить за цим посиланням, заражений сайт приймає запит, виключаючи шкідливу рядок.
3. На сторінці у користувача виконується сценарій, в результаті чого завантажується шкідливий скрипт і зловмисник отримує cookies.

### 1.2.2 SQL ін’єкція

SQL ін'єкція - це один з найдоступніших способів злому[4][1)] сайту. Суть таких ін'єкцій - впровадження в дані (передані через GET, POST запити або значення Cookie) довільного SQL коду. Якщо сайт вразливий і виконує такі ін'єкції, то по суті є можливість творити з БД (найчастіше це MySQL) що завгодно. Прикла атаки зображений на рисунку 2.

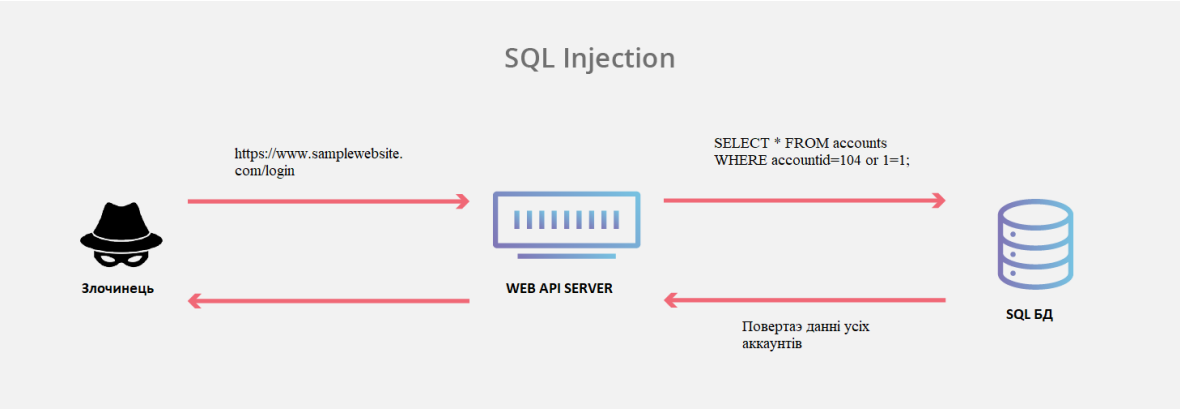



Рисунок 2 – Принцип роботи SQL ін'єкції

[1)] [4] Website Vulnerability Scanner. URL: https://pentest-tools.com/website-vulnerability-scanning/website-scanner (дата звернення 26.08.2020)

Існує п'ять основних класів SQL-ін'єкцій[5][1].

**UNION query SQL injection**. Класичний варіант впровадження SQL-коду, коли в вразливий параметр передається вираз, що починається з "UNION ALL SELECT". Ця техніка працює, коли веб-додатки безпосередньо повертають результат виведення команди SELECT на сторінку: з використанням циклу for або схожим способом, так що кожен запис отриманої з БД вибірки послідовно виводиться на сторінку. Sqlmap може також експлуатувати ситуацію, коли повертається тільки перший запис з вибірки (Partial UNION query SQL injection).

**Error-based SQL injection.** У разі цієї атаки сканер замінює або додає в вразливий параметр синтаксично неправильне вираз, після чого парсит HTTP-відповідь (заголовки і тіло) в пошуку помилок DBMS, в яких містилася б заздалегідь відома ін'єктувати послідовність символів і десь "поряд" висновок на цікавий для нас підзапит. Ця техніка працює тільки тоді, коли веб-додаток з якихось причин (найчастіше з метою налагодження) розкриває помилки DBMS.

**Stacked queries SQL injection.** Сканер перевіряє, чи підтримує веб-додаток послідовні запити, і, якщо вони виконуються, додає в вразливий параметр HTTP-запиту крапку з комою (;) і слідом впроваджуваний SQL-запит. Цей прийом в основному використовується для впровадження SQL-команд, відмінних від SELECT, наприклад для маніпуляції даними (за допомогою INSERT або DELETE). Примітно, що техніка потенційно може привести до можливості читання / запису з файлової системи, а також виконання команд в ОС. Правда, в залежності від використовуваної в якості бек-енду системи управління базами даних, а також призначених для користувача привілеїв.

**Boolean-based blind SQL injection.** Реалізація так званої сліпий ін'єкції: дані з БД в "чистому" вигляді вразливим веб-додатком ніде не повертаються.

---

[1] [5] 10 WEB vulnerabilities you can prevent. URL: https://www.toptal.com/security/10-most-common-web-security-vulnerabilities (дата звернення 26.08.2020)

Прийом також називається дедуктивним. Sqlmap додає в вразливий параметр HTTP-запиту синтаксично правильно складене вираз, що містить підзапит SELECT (або будь-яку іншу команду для отримання вибірки з бази даних). Для кожного отриманого HTTP-відповіді виконується порівняння headers / body сторінки з відповіддю на початковий запит - таким чином, утиліта може символ за символом визначити висновок впровадженого SQL-вирази. В якості альтернативи користувач може надати рядок або регулярний вираз для визначення "true"-сторінок (звідси і назва атаки). Алгоритм бінарного пошуку, реалізований в sqlmap для виконання цієї техніки, здатний витягти кожен символ виведення максимум сімома HTTP-запитами. У тому випадку, коли висновок полягає не тільки зі звичайних символів, сканер підлаштовує алгоритм для роботи з більш широким діапазоном символів (наприклад для unicode).

**Time-based blind SQL injection.** Повністю сліпа ін'єкція. Точно так само як і в попередньому випадку, сканер "грає" з вразливим параметром. Але в цьому випадку додає підзапит, який призводить до паузі роботи DBMS на певну кількість секунд (наприклад, за допомогою команд SLEEP () або BENCHMARK ()). Використовуючи цю особливість, сканер може посимвольно витягти дані з БД, порівнюючи час відповіді на оригінальний запит і на запит з впровадженим кодом. Тут також використовується алгоритм двійкового пошуку. Крім того, застосовується спеціальний метод для верифікації даних, щоб зменшити ймовірність неправильного вилучення символу через нестабільне з'єднання.

### 1.2.3 HTML ін'єкція.

HTML ін'єкція є вразливістю для сайтів і розробників, оскільки може бути іспольщована для того, щоб обдурити користувачів і змусити їх відсилати

приватні дані або відвідати сайти зловмисників. Це вже називається фішинговою атакою [6][1)].

Виявлення цих уразливостей не завжди вимагає відправки простого HTML, воно може включати в себе дослідження того, як сайт може рендерить введені значення, такі як закодовані символи URI. І хоча це не зовсім та сама HTML ін'єкція, підміна контента подібна їй в тому, що включає в себе деякий відбитий на отриману користувачем сторінку введення. Хакери повинні бути уважні до можливостей маніпулювання параметрами URL і тому, як вони відображаються на сайті.

Оскільки HTML є мовою, використовуваним для визначення структури веб-сторінки, якщо зловмисник може впровадити HTML, він може повністю змінити те, що відображає браузер. Іноді результатом цього може стати повна зміна зовнішнього вигляду сторінки або, в інших випадках, створення форми для обману користувачів. Наприклад, якщо ви можете впровадити HTML, ви можете додати тег "'<form>' на сторінки, просячи користувача заново ввести його логін і пароль. Однак, при відправці така форма передасть інформацію зловмисникові.

### 1.2.4 Виконання команд ОС.

Зазвичай застосовується термін «впровадження коду», він відноситься до SQL-ін'єкцій, Міжсайтовий Скриптінг (XSS), PHP-ін'єкціям та великої кількості інших ін'єкцій і їх різновидів. Впровадження команд ОС (іноді пишуть «ін'єкція команд ОС», «інжект команд ОС») - це одна з різновидів впровадження коду. Її особливістю є виконання несанкціонованих команд операційної системи на віддаленому сервері через вразливе веб-додаток.

Більшість веб-серверів підтримують функціональність, яка дозволяє даними взаємодіяти з серверної операційної системою. Ці функції можуть

---

[1)] [6] OWASP top 10 security vulnerabilities. URL: https://sucuri.net/guides/owasp-top-10-security-vulnerabilities-2020/ (дата звернення 27.08.2020)

бути корисними при створенні додатків, можливості яких уже реалізовані в ОС. В результаті не потрібно писати нову програму, можна просто передавати команду з додатковими опціями в систему і відображати результати на веб-сайті.

Якщо по відношенню до призначеного для користувача введення не провадиться належна перевірка, то додатки можуть бути уразливі до атаки відомої як впровадження команд. Використовуючи цю вразливість атакуючі може формувати введення таким чином, що він буде містити команди операційної системи, які будуть виконуватися з привілеями уразливого програми. Уразливості впровадження команд зазвичай поділяються на такі види:

- Впровадження команд, засноване на результатах - Вразливе додаток виводить результати впровадженої команди.
- Сліпе впровадження команд - Вразливе додаток не виводить результати впровадженої команди.

Ефективно використовуючи цю уразливість, атакуючий може отримати чутливі дані, такі як:

- Файли паролів операційної системи
- Файли операційної системи
- Вихідний код програми.

### 1.2.5 Перебір паролів (Брутфорс).

Брут-форс (перебір паролів) на веб-сайтах викликає найбільше проблем у (початківців) пентестерів[6][1)]. Якщо перебирати паролі на різних, наприклад, FTP серверах, то команди, якими запускаються програми, будуть мало відрізнятися один від одного - тільки різні цілі. Якщо ж ми переходимо до перебору паролів в веб-формах, то тут все по-іншому: важко знайти два сайти,

1) [6] OWASP top 10 security vulnerabilities. URL: https://sucuri.net/guides/owasp-top-10-security-vulnerabilities-2020/ (дата звернення 27.08.2020)

на яких був би однаковий набір полів форми з однаковими іменами та однакове поводження при успішному або неуспішному вході.

Крім цього розмаїття, навіть без проактивного захисту веб-форма може бути створена розробником так, що в неї вже після натискання на кнопку «Відправити» додаються поля, без яких сервер не приймає форму. Аналізом статичного коду це з'ясувати іноді дуже непросто. Також на стороні веб-майстри дуже легко реалізувати такі анти-брутфорс заходи як додавання прихованих полів з випадковими значеннями, аналіз заголовка Referer та інше. Все це вимагає додаткових сил на аналіз.

Ну а якщо в справу вступають серьуоьзние проактивні захисту, такі як капча, блокування спроб входу при декількох невдалих спробах, двофакторна аутентифікація і т.д., то навіть у досвідчених пентестеров опускаються руки.

І тим не менше, брут-фос облікових даних на веб-сайтах дуже цікавий для тестерів на проникнення. Оскільки якщо ПО сервера і веб-додатки не містить відомих уразливостей, то підбір пароля залишається одним з небагатьох методів компрометації.

### 1.2.6 IDOR

Небезпечні прямі посилання на об'єкти (IDOR) - це тип уразливості управління доступом, що виникає, коли програма використовує введені користувачем дані для безпосереднього доступу до об'єктів. Термін IDOR був популярний завдяки його появі в першій десятці OWASP 2007. Однак це лише один із прикладів багатьох помилок впровадження контролю доступу, які можуть призвести до обходу елементів керування доступом. Уразливості IDOR найчастіше асоціюються з горизонтальною ескалацією привілеїв, але вони також можуть виникати стосовно вертикальної ескалації привілеїв.

Веб-сайт, який використовує таку URL-адресу для доступу до сторінки облікового запису клієнта, отримуючи інформацію із внутрішньої бази даних: https://insecure-website.com/customer_account?customer_number=132355

Тут номер замовника використовується безпосередньо як індекс запису в запитах, які виконуються у внутрішній базі даних. Якщо жодних інших елементів керування немає, зловмисник може просто змінити значення customer_number, обминаючи елементи керування доступом для перегляду записів інших клієнтів. Це приклад уразливості IDOR, що веде до горизонтальної ескалації привілеїв.

Зловмисник може виконати ескалацію привілеїв по горизонталі та вертикалі, змінивши користувача на одного з додатковими привілеями, одночасно обходячи елементи керування доступом. Інші можливості включають використання витоку пароля або модифікацію параметрів після того, як зловмисник потрапив на сторінку облікових записів користувача, наприклад.

### 1.2.7 Небезпечне встановлення паролів (Weak Password Recovery Validation).

Ця вразливість виникає, коли Web-сервер дозволяє атакуючому несанкціоновано отримувати, модифікувати або відновлювати паролі інших користувачів[7][1)].

Часто аутентифікація на Web-сервері вимагає від користувача запам'ятовування пароля або парольної фрази. Тільки користувач повинен знати пароль, причому пам'ятати його виразно. Згодом пароль забувається. Ситуація ускладнюється, оскільки в середньому користувач відвідує близько 20 сайтів, які потребують введення пароля.

Таким чином, функція відновлення пароля є важливою складовою надається Web-серверами сервісу.

---

1) [7] Website vulnerabilities and tests. URL: https://cobweb-security.com/security_lessons/website-vulnerabilities-threats/ (дата звернення 27.08.2020)

Прикладом реалізації подібної функції є використання "секретного питання", відповідь на який вказується в процесі реєстрації. Питання або вибирається зі списку або вводиться самим користувачем.

Ще один механізм дозволяє користувачеві вказати "підказку", яка допоможе йому згадати пароль. Інші способи вимагають від користувача вказати частина персональних даних, таких як номер соц. страхування, ІПН, домашню адресу поштовий індекс і т.д., які потім будуть використовуватися для встановлення особи. Після того як користувач доведе свою ідентичність, система відобразить новий пароль або перешле його поштою.

Уразливості пов'язані з недостатньою перевіркою при відновленні пароля виникають, коли атакуючий отримує можливість використовується механізм. Це трапляється, коли інформацію, використовувану для перевірки користувача, легко вгадати або сам процес підтвердження можна обійти.

Система відновлення пароля може бути скомпрометована шляхом використання підбору, уразливостей системи або через легко вгадується відповіді на секретне питання.

Багато серверів вимагають від користувача вказати його email в комбінації з домашньою адресою і номером телефону. Ця інформація може бути легко отримана з мережевих довідників. В результаті, дані, які використовуються для перевірки, не є великим секретом. Крім того, ця інформація може бути отримана зловмисником з використанням інших методів, таких як міжсайтового виконання сценаріїв або фішинг (phishing).

### 1.2.8 Переповнення буфера (Buffer Overflow).

Переповнення буфера зазвичай мають високий рейтинг серйозності, оскільки вони можуть призвести до несанкціонованого виконання коду в тих випадках, коли зловмисники можуть контролювати перезаписаний простір

[7][1)] пам'яті поза цільовим буфером і можуть перенаправляти покажчик функції на свій шкідливий код.

Навіть коли довільне виконання коду неможливе, переповнення буфера часто призводить до збою, що призводить до стану відмови в обслуговуванні (DoS), що впливає на доступність програми та оброблювані нею процеси. Це особливо погано під час розгортання серверів, де необхідна та очікується постійна доступність.

### 1.2.9 Підміна вмісту (Content Spoofing)

Спуфінг вмісту, який також називають введенням вмісту, «довільним введенням тексту» або віртуальним зіпсуванням, - це атака, спрямована на користувача, що стала можливою завдяки уразливості введення у веб-програмі. Коли програма неправильно обробляє дані, надані користувачем, зловмисник може надавати вміст веб-програмі, як правило, через значення параметра, яке відображається користувачеві. Це представляє користувачеві змінену сторінку в контексті довіреного домену. Ця атака зазвичай використовується як соціальна інженерія або разом із нею, оскільки атака використовує вразливість на основі коду та довіру користувача. Як зауваження, цю атаку широко сприймають неправильно як певну помилку, яка не приносить ніякого впливу.

Фактори ризику залежать від бізнес-типу програми. Якщо торгова марка додатків добре відома і має основних конкурентів, зловмисними конкурентами / незадоволеними працівниками / незадоволеними клієнтами можуть зловживати цією проблемою, щоб викликати масовий розповсюдження неправдивих повідомлень для нічого не підозрюючих клієнтів. Ще одним фактором, що підвищує ризик, є введення SEO-ін’єкції

---

1) [7] Website vulnerabilities and tests. URL: https://cobweb-security.com/security_lessons/website-vulnerabilities-threats/ (дата звернення 27.08.2020)

таким чином, що пошукові системи сканують та індексують створені URL-адреси з підробленими повідомленнями.

Таким чином, клієнти можуть бути змушені перейти на товари конкурентів. Це може призвести до втрати грошової вартості, доки виправлення не буде належним чином здійснено бізнесом-жертвою. Для публічних компаній, що торгуються, їх акції падатимуть, що призведе до неконтрольованих втрат мільйонів

Зловмисник скомпрометував соціальні акаунти, які мають тисячі підписників, і розповсюджує оманливий корисний набір вмісту, що підмінює вміст, через Twitter / Facebook / Instagram / подібний популярний канал. Це змусить ЗМІ вважати новини правильними та створювати заголовки.

### 1.2.10 Cross-site Scripting

Міжсайтовий сценарій (XSS) - це атака введення коду, яка дозволяє зловмиснику виконувати шкідливий JavaScript у браузері іншого користувача.

Зловмисник безпосередньо не націлює свою жертву. Натомість він використовує уразливість на веб-сайті, який відвідує жертва, щоб змусити веб-сайт доставити для нього шкідливий JavaScript. Для браузера жертви шкідливий JavaScript, здається, є законною частиною веб-сайту, і, таким чином, веб-сайт діяв як ненавмисний співучасник зловмисника. Ці атаки можна здійснювати за допомогою HTML, JavaScript, VBScript, ActiveX, Flash, але найчастіше використовується XSS - шкідливий JavaScript.

Ці атаки також можуть збирати дані про викрадення облікового запису, зміну налаштувань користувача, викрадення / отруєння файлів cookie або помилкову рекламу та створювати атаки DoS

XSS-атаки часто поділяють на три типи

- Постійний XSS, де зловмисний рядок походить із бази даних веб-сайту

- Відображений XSS, де зловмисний рядок походить від запиту жертви.
- XSS на основі DOM, де вразливість полягає в коді на стороні клієнта, а не в коді на стороні сервера.

### 1.2.11 Недостатня аутентифікація (Insufficient Authentication).

За рахунок цієї уразливості зловмисник може отримати доступ до функцій сервера і важливої інформації, не маючи відповідних прав доступу. Недостатня автентифікація відбувається, коли веб-сайт дозволяє зловмиснику отримати доступ до конфіденційного вмісту [8][1)] або функціональних можливостей без необхідності належної автентифікації. Засоби веб-адміністрування є хорошим прикладом веб-сайтів, що забезпечують доступ до чутливих функціональних можливостей. Залежно від конкретного Інтернет-ресурсу, ці веб-програми не повинні бути безпосередньо доступними, не вимагаючи від користувача належної перевірки своєї особистості.

Щоб обійти налаштування автентифікації, деякі ресурси захищаються, "приховуючи" конкретне місце розташування та не прив'язуючи місцезнаходження до основного веб-сайту чи інших громадських місць. Однак цей підхід є не що інше, як "Безпека через невідомість". Важливо розуміти, що, незважаючи на те, що ресурс невідомий зловмиснику, він все одно залишається доступним безпосередньо через певну URL-адресу. Конкретну URL-адресу можна знайти за допомогою зондування Brute Force для поширених розташувань файлів і каталогів (наприклад, / admin), повідомлень про помилки, журналів перенаправлення або документації, наприклад файлів довідки. Ці ресурси, незалежно від того, керуються вони вмістом чи функціоналом, повинні бути належним чином захищені.

---

1) [8] Common website security vulnerabilities. URL: https://www.weblite.com.my/blog/website-security-vulnerabilities. (дата звернення 4.09.2020)

### 1.2.12 DDOS-атака

DDOS-атака (з англ. Distributed Denial of Service - «відмова від обслуговування») - це атака на сайт, основною метою якої є виведення його з ладу шляхом подачі великої кількості помилкових запитів[9] [1)]. В результаті такої атаки сервера, які обслуговують сайт, змушені обробляти надмірний обсяг помилкових запитів, і сайт стає недоступним для простого користувача6.

Основною метою для зловмисників, ddos застосовую для усунення конкуренції в тій чи іншій галузі, шляхом повного припинення роботи атакується сервера за рахунок подачі на нього велику кількість помилкових запитів, з якими не буде справлятися ваш сайт.

Популярними жертвами таких атак стають комерційні та інформаційні сайти. Хацкери останнім часом використовують такий вид атак з метою вимагання, вимагаючи грошей за припинення атаки, або ведуть інформаційну війну.

Раніше коли веб-пространство було дуже простим атаку можна було організувати навіть за допомогою несправної кнопки. Можна було затиснути кнопку F5 на клавіатурі. Через частого оновлення сторінки, яка робила часті запити на сервер. Сервер просто напросто не справлявся з таким навантаженням з слабкості.

Зараз же атаки організовують за допомогою БОТ-мереж. Це сукупність заражених вірусом комп'ютерів які здатні синхронно виконувати команди передані з керуючого сервера. Наприклад, якщо бот-мережі з тисячі комп'ютерів дати команду відкрити сайт, то на цільовому сайті різко зростає навантаження і сайт отримує ДДОС атаку.

Методи DDOS атак:

Слабка лінія пропускання - даний вид атаки передбачає що на веб сайт спрямовується велика кількість запитів по протоколам TCP, UDP і ICMP і

---

[1)] [9] Website vulnerability scanner tool. URL: https://www.cybersecuritywebtest.com/website-vulnerability-scanning/website-scanner-tool. . (дата звернення 7.09.2020)

таким чином повністю заповнюють його пропускну здатність. Викликаючи при цьому відмова в обслуговуванні.

На основі протоколу сервера - даний вид атаки спрямований на конкретні сервіси сервера. І може виконуватися за допомогою TCP, UDP і ICMP. Часто такі атаки називають SYN-флуд, сенс яких в відсилання на веб сервер великої кількості SYN запитів на які сервер повинен відповісти запитом ASK. Через велику повінь таких запитів, сервер часто не справляється з навантаженням і падає.

На основі помилок конкретного веб сайту - цей вид атаки є найскладнішим в плані виконання і застосовується як правило високо-професійними хакерами. Суть його полягає в тому що на сайті-жертві знаходяться уразливості, використовуючи які створюється високе навантаження на сервер і він отримує відмову в обслуговуванні.

# 2 ІСНУЮЧІ ПРОДУКТИ ДЛЯ ПОШУКУ УРАЗЛИВОСТЕЙ

Оцінка захищеності проводиться як ручним способом методів чорного, серого і білого ящика з використанням вспомогальних автоматизованих засобів, так і автоматизовано, за допомогою аналізатора вихідних кодів. Метод чорного ящика входить в оцінку захищеності інформаційної системи від осіб зовнішнього атакуючого без попереднього отримання від власника якої-небудь додаткової інформації про неї. Метод сірого ящика аналогічен методу чорного ящика, но у якості порушитель розглядає користувача, що володіє певними привілегіями в системі. У методі білого ящика для оцінки захищеності інформаційної системи використовуються всі необхідні дані про неї, включаючи вихідний доданий код.

Безпека веб-ресурсів – є одним з найбільш гострих питань в контексті інформаційної безпеки. Як правило більшість сайтів, доступних в Інтернеті, мають різного роду уразливості і постійно піддаються атакам котрі поділяються на декілька рівнів загрози(рис.3).

Основні типи загроз для інформаційної безпеки веб-додатків (сайтів):

1. Загрози конфіденційності - несанкціонований доступ до даних.
2. Загрози цілісності - несанкціоноване спотворення або знищення даних.
3. Загрози доступності - обмеження або блокування доступу до даних.

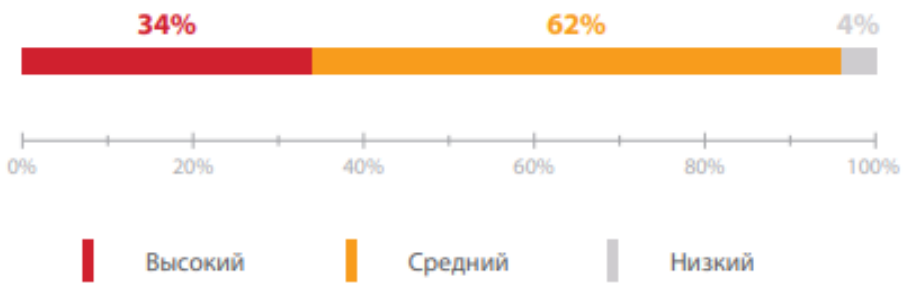



Рисунок 3 – Кількісний рівень загроз у 2015 році

На сьогоднішній день статистичні дані використовуються лише уявно, пов'язані з помилками в кодах та конфігурації веб-додатків. Інші поширені проблеми інформаційної безпеки (до прикладу, недостатності процесу управління оновленнями ПО) не розглядаються.

Доля уявних додатків, що знаходяться в експлуатації, дуже велика. Більше половины таких ресурсів (63%) підвержені критично небезпечним уявленням. Ці недоліки можуть викликати зловмисник встановити повний контроль над системою (наприклад, у разі завантаження файлів користувачів або виконаного команди), а також отримувати чутливу інформацію (наприклад, у результаті експлуатації уявних користувачів «Внедрение операторів SQL »або« Внедрення зовнішніх об'єктів XM L »).

Розподіл систем за засобами розробки можна знайти на рисунку 4.

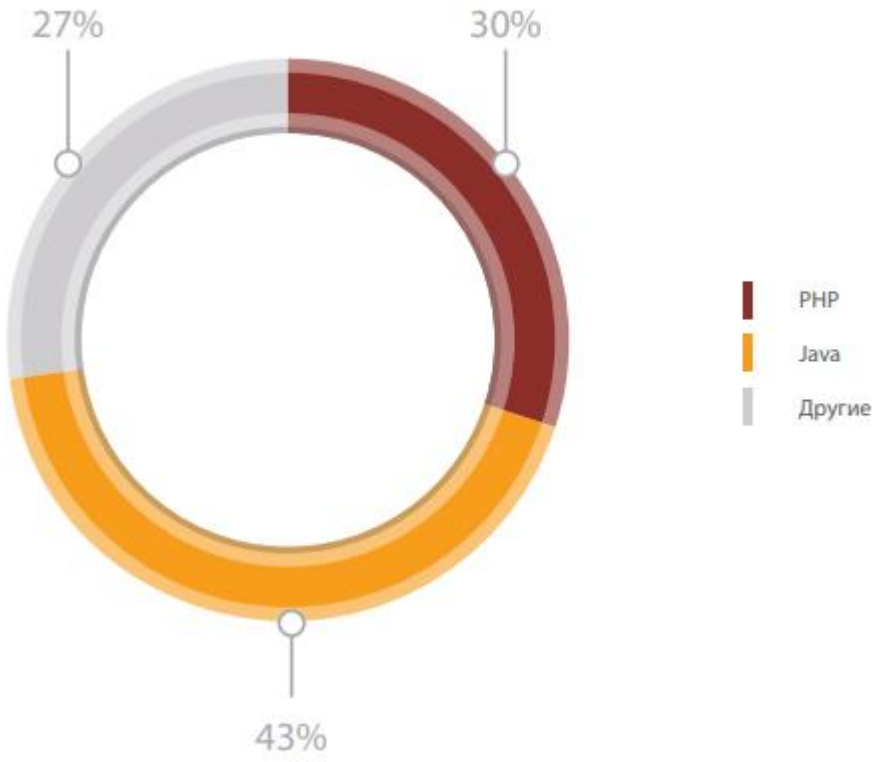



Рисунок 4 – Діаграма розподілу програм за мовою розробки

## 2.1 Потреба в програмному продукту

Немає досконалих програм, і важко знайти ту що цілком б підходила по своєму функціоналу, інтерфейсу та інструментарію для необхідних дій. Тому було прийнято рішення у оптимізації та аналізу уже існуючих продуктів для проведення аудиту. Головною проблемою таких програм є необхідність їх постійного вдосконалення. Через те що кожного дня у світі знаходять нові уразливості систем, системи безпеки мають бути значно технічні та вдосконалені. Також із появою нових технологій такі програми мають розширювати свій функціонал та багаж інструментів.

Часто програмне забезпечення для аудиту з відкритим кодом є безкоштовним у використанні. На відміну від власного програмного забезпечення, ви можете налаштувати інструменти аудиту з відкритим кодом. Малі та стартові підприємства, які мають менший бюджет, можуть використовувати безкоштовні рішення для аудиту. Поряд з цим можна навіть вибрати програмне рішення для аудиту з відкритим кодом, яке не фіксує вас за жодною ліцензією, а програмне забезпечення постійно проходить декілька вдосконалень.

### 2.1.1 Open–AudIT

Open–Audit – це програма аудиту мережі. Вона написана на мовах сценаріїв PHP , Bash і VBScript. Open–Audit може повідомити, що є у вашій мережі, як вона налаштована та чи були якісь зміни. Завдяки легкому розширенню спільнота користувачів перетворилася на понад 25 000 користувачів [10][1)].

Цей інструмент автоматично створює, звіти та аналіз. Він виявляє кожен пристрій і створює графік для автоматизації повторюваних завдань.

---

1) [10] Website vulnerability scanner online. URL: https://www.acunetix.com/vulnerability-scanner/website-vulnerability-scanner-online/ . (дата звернення 10.9.2020)

Візуалізація приладної панелі та діаграм пропонує детальну інвентаризацію та переклад даних у зручну для споживання інформацію.

Це ефективне рішення для зайнятих IT–спеціалістів, які хочуть зменшити накладні витрати та збільшити кількість даних завдяки наявності аудиту системи. Підприємство Open–Audit пропонує потужність, гнучкість та функції для організацій, щоб вони могли керувати товарно–матеріальними запасами та задовольняти свої потреби корпоративного дотримання.

Інвентаризація мережі та збір даних без агентів. Open–Audit використовує потужні сценарії аудиту, щоб уникнути встановлення агента, одночасно збираючи неймовірну кількість конфігураційних даних зі своїх пристроїв.

Програмне забезпечення для інвентаризації серверів для моніторингу змін. Open–Audit має систему виявлення змін. Будь–які атрибути пристрою, які додаються, видаляються чи змінюються, будуть виявлені та збережені.

Інструмент управління ліцензіями на програмне забезпечення. Ліцензії та гарантії на програмне забезпечення є важливим елементом для відстеження. Open–Audit дозволяє легко записувати та звітувати про гарантії.

Можна покращити:

Моніторинг цілісності файлів. Деталі, такі як ім'я, каталог, розмір, хеш, остання зміна файла не записуються.

Можливість контролю користувачів. В програмі відсутня можливість створення користувачів та обмеження їх доступу.

Покращена візуальна звітність. Оскільки для використання Open–AudIT необхідний Nmap, велику кількість інформації можна переоформити з текстового вигляду до графічного у вигляді таблиць, графіків.

### 2.1.2 ADAudit Plus

ADAudit Plus – це безкоштовне програмне рішення для аудиту, яке здійснює онлайн–зміни в Active Directory. Інколи ваш Windows Active

Directory змінюється. Зміни фіксуються цим рішенням аудиту з відкритим кодом, що допомагає своєчасно готувати аудиторські звіти[11][1)].

Це просте у використанні та ефективне внутрішнє аудиторське рішення допомагає користувачам дізнатися, хто має старий сеанс та заблокував обліковий запис. У режимі реального часу користувачі можуть контролювати та генерувати звіти про ресурси для таких елементів, як контролери домену.

Ви можете отримати інформацію про активні об'єкти каталогів, що складаються з користувачів, комп'ютерів, груп разом із змінами конфігурації. Користувачі можуть збирати аудити використання робочих станцій з історією входу, тривалістю та збоями входу. Для плавного аудиту це програмне забезпечення зберігає важливу інформацію.

У підключених системах це програмне забезпечення відстежує створення, видалення та модифікацію файлів. У разі модифікації документа та доступу до документа адміністратор отримуватиме сповіщення та сповіщення електронною поштою. Ви знайдете добре розроблений інтерфейс користувача та зможете побачити останні зміни через слід аудиту. Якщо ви не в змозі дати швидкі відповіді, ADAudit допоможе вам завдяки своїй інтуїтивній функції звітування.

### 2.1.3 Netwrix

Netwrix – це безкоштовний інструмент аудиту, який дозволяє спростити моніторинг мережевих пристроїв.

Це програмне забезпечення для управління аудитом з відкритим кодом, щоб посилити безпеку, проілюструвати відповідність та забезпечити час роботи систем. Програма служить системою аудиту змін конфігурації, яка забезпечує чітку та читану людиною інформацію про аудит. Якщо

---

[1)] [11] Best 14 open-source web application vulnerability scanners URL: https://resources.infosecinstitute.com/topic/14-popular-web-application-vulnerability-scanners/ . (дата звернення 12.09.2020)

неавторизована особа входить у вашу систему, ви можете розслідувати інциденти безпеки, дізнавшись точний час за допомогою аудиту входу.

Можна швидко виявити зовнішні та внутрішні загрози, надавши докладні аудиторські звіти та повідомити про зміни, які можуть призвести до інцидентів із безпекою. Ефективно цей інструмент усуває випадки.

Не доведеться витрачати багато днів на складання звітів або витрачати гроші на послуги підтримки відповідності, користуючись Netwrix Auditor. При менших зусиллях це програмне забезпечення доводить відповідність IT. Аудит файлового сервера Windows у цьому програмному забезпеченні дозволяє приймати кращі рішення щодо управління інформацією щодо неструктурованих даних.

Особливості:

- активна безпека та відповідність каталогу;
- звітність про поточні конфігурації;
- попередження про схеми загроз;
- аудит доступу до файлів.

### 2.1.4 OWASP ZAP

Розроблений OWASP (Open Security Security Project), ZAP або Zed Attack Proxy – це багатоплатформенний інструмент тестування безпеки веб–додатків з відкритим кодом [11][1)]. ZAP використовується для пошуку ряду вразливих місць безпеки у веб–додатку під час розробки, а також на етапі тестування. Завдяки інтуїтивно зрозумілому графічному інтерфейсу, Zed Attach Proxy можна з однаковою легкістю використовувати новачкам, як і експертам. Засіб

[1)] [11] Best 14 open-source web application vulnerability scanners URL: https://resources.infosecinstitute.com/topic/14-popular-web-application-vulnerability-scanners/ . (дата звернення 12.09.2020)

тестування безпеки підтримує доступ до командних рядків для досвідчених користувачів.

Окрім того, що є одним із найвідоміших проектів OWASP , йому присвоюють статус флагмана. ZAP написаний на Java. Окрім використання в якості сканера, ZAP також може використовуватися для перехоплення проксі для ручного тестування веб–сторінки що зображено на рисунку 5.

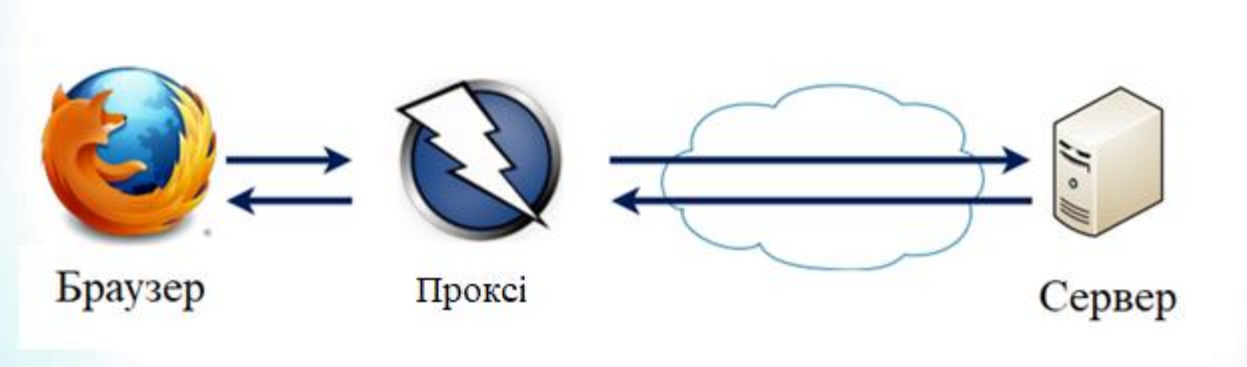



Рисунок 5 – Схема роботи OWASP ZAP з сервером

Режими OWASP ZAP[12][1)]:

1) Безпечний режим – з цим режимом можна зробити вам щось потенційно небезпечне для вашого застосування.
2) Захищений режим – за допомогою цього режиму користувач (наш бот) може виконувати тільки шкідливі дії по URL–адресами, вказаними в області браузера.
3) Стандартний режим – У цьому режимі наш користувач (Бот), може робити все, що має значення для нашого застосування.

---

[1)] [12] Тестирование на XSS и другие уязвимости с помощью OWASP ZAP. URL:https://medium.com/@svyatoslavlogyn/%D1%82%D0%B5%D1%81%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BD%D0%B0-xss-%D0%B8-%D0%B4%D1%80%D1%83%D0%B3%D0%B8%D0%B5-%D1%83%D1%8F%D0%B7%D0%B2%D0%B8%D0%BC%D0%BE%D1%81%D1%82%D0%B8-c-%D0%BF%D0%BE%D0%BC%D0%BE%D1%89%D1%8C%D1%8E-owasp-zap-a99183c32013 (дата звернення (27.09.2020)

4) Режим ATTACK – при знаходженні нових вузлів в області дії шпигуна, активно скануються, як тільки вони виявляються.

Основні можливості ZAP:

- перехоплення проксі
- автоматизований сканер
- пасивний сканер
- сканер грубої сили
- fuzzer (передача на вхід неправильних, несподіваних або випадкових даних)

Він призначений для користувачів з широким спектром досвіду в області безпеки, тому відмінно підходить для розробників і функціональних тестувальників, які погано знайомі з пентестамі.

ZAP створює проксі-сервер між клієнтом і сайтом. Поки ви переміщується по своєму веб-сайту, він фіксує всі дії, а потім атакує сайт відомими методами.

### 2.1.5 LOIC

The Low Orbit Ion Cannon (LOIC) Низько орбітальна іонна гармата. Можливо найпопулярніша DDOS програма. Вона може розсилати масові запити по протоколам ICMP, UDP тим самим забиваючи канал до сервера жертви. Найвідоміша атака за допомогою LOIC була здійснена групою Anonymous в 2009 році і спрямована проти PayPal, Visa, MasterCard в помсту за відключення WikiLeaks від системи збору пожертвувань.

JS LOIC атакує випадковими HTTP-запитами. Тут є сигнатура: варто невірний referer того сайту, звідки програма запускалася, і URI, сформований абсолютно неприродним для будь-якого веб-ресурсу чином. Ми бачимо внутрішній LOIC'овскій номер запиту і відправлене протестуючих

повідомлення. Чого вартий це відфільтрувати? Та нічого. Тут навіть поведінковий аналіз не потрібен. Невеликі атаки LOIC HTTP можна пом'якшити за допомогою локального брандмауера, дозволивши адміністратору сервера переглянути журнали та визначити IP-адреси зловмисників та скинути їх запити. Однак ця стратегія не витримає масштабної атаки, коли сотні чи навіть тисячі різних зловмисників працюють в тандемі. Місцеві брандмауери також не можуть захистити від повені TCP або UDP, останні з яких можуть навіть націлювати та порушувати роботу брандмауера. Брандмауер веб-додатків (WAF) може забезпечити надійний захист від повені HTTP, а спеціальний захист DDoS може зупинити атаки TCP та UDP.

Стандартний LOIC UDP. Нічого особливого. Утиліта шле UDP-пакети довжиною 48 байт на HTTP-порт. Які проблеми захиститися від цього? Що взагалі повинен робити UDP-пакет на вашому www-сервері? Будуйте access-листи. Інша варіація: LOIC TCP відкриває чесне TCP-з'єднання і засинає його нерелевантних сміттям. Які проблеми з цим можуть бути? Знову ж ніяких: бюджетуються ресурси, обмежуйте розміри даних і буферів. Веб-сервери nginx, Varnish все це вміють.

Наступний варіант, самий класичний, - LOIC HTTP. А тут взагалі нема чого ловити. Принцип у нього простий: GET '/' так швидко, як ми тільки зможемо, та ще й заголовки невалидность. З точки зору захисту, такі запити до вашого бекенд доходити в принципі не повинні, якщо ваше додаток правильно побудовано.

Ось ще такий цікавий інструментарій Anonymous - OWASP / SLOWPOST. Відкриваємо з'єднання і шолом дані з затримкою так довго, як тільки можна, щоб зайняти ресурси Воркер на стороні сервера. Атака стара - нічого нового тут немає.

Є ще відносно новий інструментарій HOIC (High Orbit Ion Cannon) с кумедними апгрейдами. Атаки в цьому інструменті можна скріптовать, можна додавати список юзер-агентів і реферерів, які буде проставляти гармата, щоб

збити захищає сторону з пантелику і не дати побудувати стратегію по паттернам. Користувачеві навіть пропонується зробити список URL, які потім гармата вибере випадковим чином. Втім, відбитися все одно не так складно.

На щастя, зловмисників, які використовують LOIC, досить легко виявити; його не можна використовувати через проксі-сервер, тому IP-адреси зловмисників видно цілі. Багато країн вжили законних заходів проти зловмисників, що використовують LOIC, включаючи США, Великобританію, Іспанію та Туреччину.

# 3 ЗАСОБИ ДЛЯ ПРОВЕДЕННЯ ТЕСТУВАННЯ ТА РОБОТА З OWASP ZAP

Більше половин висвітлених веб-сайтів, розроблених на PHP, містить критично небезпечні уявлення (56%). Відповідне значення для веб-ресурсів, розроблених з використанням технологій Java, виявилося вище і склало 69%. Практично все сайти, створені за допомогою інших мовних програм програмування, підвержені уявним рівням високого рівня ризику (88%).

Однією з сильних сторін ZAP є його доступність для багатьох платформ та операційних систем: ми спробували його в середовищі Windows, але його також можна встановити на Linux та macOS. Єдиною вимогою є наявність Java 7 або новішої версії для Windows та Linux, версія для macOS вже забезпечує Java 8. Будучи настільки гнучкою, ZAP може використовуватися як контейнер Docker через браузер. Процедура встановлення в Windows виконується звичайним процесом виконуваних файлів, включаючи ліцензійний договір, спеціальні параметри та доопрацювання.

## 3.1 Засоби для пентестингу

В наш час, кіберзлочинці мають безліч прийомів та методів, готових отримати доступ до мережі. Як правило, вони користуються перевагами загальних уразливостей, наприклад недопрацьоване ПЗ або паролі за замовчуванням. Саме ця причина є основною для того, щоб встановити регулярний процес пошуку уразливих місць, які піддають ризику ПК та дані, що знаходяться на ньому[13][1)].

Необхідно пам’ятати, що інформація про систему все ще може просочуватися та бути використана проти людини (наприклад, імена

[1)] [13] PentestBox. URL: https://pentestbox.org/ . (дата звернення 13.9.2020)

комп’ютерів та робочих груп). Якщо порти не фільтруються брандмауером, інформація може просочитися в Інтернет.

### 3.1.1 Павук

Павук (spider) - це програма, "повзає" по Інтернету певним чином і з певною метою. Мета може полягати в зборі інформації або в розумінні структури будь-якого Web-сайту і його корисності. На застосуванні павуків засновані сучасні пошукові машини, такі як Google і AltaVista. Такі павуки автоматично витягають дані з Web-сайту і передають їх іншим додаткам, які індексують контент цього Web-сайту з метою формування найкращого набору пошукових термінів[14][1)].

Набір веб-спайдеров, викачують з сайтів-джерел сторінки з результатами пошуку. Це ті сторінки, до яких зазвичай не добирається пошуковий бот (за винятком SEO-сторінок, але кількість оголошень в них швидше за все буде не повним і містить лише найбільш популярні категорії). Для кожного сайту доведеться написати свій плагін зі своєю логікою запитів. Іноді це GET-запит з параметрами в querystring, іноді доведеться слати POST або навіть слати параметри через cookies. Для написання такого плагіна допоможе будь-який HTTP-аналізатор[15][2)], вбудований в браузер. Завдання плагіна, завантажувати результати пошуку - охопити всі категорії оголошень на сайті, відвідати всі його сторінки і зберегти вміст у тимчасове сховище. Щоб уникнути зациклення при переміщенні по сторінкам (paging - переміщення по сторінках результату), рекомендується порівнювати вміст сторінки з попередньою.

---

1) [14] Ethical hacking and cross-site scripting. URL: https://www.tutorialspoint.com/ethical_hacking/ethical_hacking_cross_site_scripting.htm (дата звернення 13.09.2020)

2) [15] Лучшие дистрибутивы для проведения тестирования на проникнове-ние. URL: https://habr.com/ru/post/276477/ . (дата звернення 06.10.2020)

### 3.1.3 Ajax-павук

Асинхронний JavaScript + XML (Ajax) не є веб-технологією; це сукупність технологій, створених спеціально для створення динамічних веб-додатків. Завдяки набору функцій та простоти використання, Ajax на сьогодні є одним із найбільш широко використовуваних інструментів для створення веб-додатків. Усі програми, включаючи програми, створені з використанням технологій Ajax, є вразливими до експлоїтів, що компрометують веб-сайти та бази даних, які ними керують.

Програми Ajax створені для підключення до веб-сайту, на якому вони розміщені. В якості міри безпеки програма з сайту A не може підключитися до сайту B. Однак багато веб-сайтів покладаються на веб-сайти третіх сторін та джерела даних для створення змішувань. Міст служб Ajax був створений для забезпечення веб-служби через хост, який діє як проксі-сервер, який переадресовує трафік між JavaScript, що працює в браузері, і стороннім сайтом. Використовуючи мости Ajax, сайт A тепер може надавати дані або вміст своїм відвідувачам, які приходять з сайту B. Подібно до того, як Ajax не є специфічною технологією, а сукупністю технологій, мостування не є конкретною вразливістю. Мост Ajax збільшує ландшафт загрози, надаючи додатковий шлях атаки для зловмисних хакерів. Такі атаки, як ін'єкції XSS та SQL, можна передавати через службу мостів Ajax. Незважаючи на те, що сайт B міг зробити все, щоб захистити свій веб-додаток від загроз, доступних для відвідувачів, сайт A може бути використаний для нападу на сайт B за допомогою мосту Ajax, про який не помічали.

### 3.1.3 Fuzzing

Частина генерації даних складається з генераторів, а ідентифікація уразливості покладається на засоби налагодження. Генератори зазвичай

використовують комбінації статичних векторів розмивання (відомі небезпечні значення) або абсолютно випадкові дані. Розмивачі нового покоління використовують генетичні алгоритми для зв’язку введених даних та спостережуваного впливу. Такі інструменти ще не є загальнодоступними.

Кількість можливих випробуваних рішень - це простір досліджуваних рішень. Метою криптоаналізу є зменшення цього простору[16][1)], що означає пошук способу мати менше ключів для спроби, ніж чиста груба сила, щоб дешифрувати щось.

Більшість залежить від:

- протоколу
- формату файлу
- типу даних

### 3.1.4 Примусовий перегляд (Forced browsing)

Примусовий перегляд - це атака, метою якої є перерахування та доступ до ресурсів, на які програма не посилається, але все ще доступні[17][2)].

Зловмисник може використовувати прийоми грубої сили для пошуку не пов’язаного вмісту в каталозі домену, наприклад, тимчасових каталогів та файлів, а також старих файлів резервної копії та конфігурації. Ці ресурси можуть зберігати конфіденційну інформацію про веб-додатки та операційні системи, такі як вихідний код, облікові дані, адресація внутрішньої мережі тощо, що вважається цінним ресурсом для зловмисників.

Ця атака виконується вручну, коли каталоги та сторінки індексу додатків базуються на формуванні чисельності або передбачуваних значеннях, або з

---

1) [16] PentestBox. URL: https://pentestbox.org/ . (дата звернення 13.10.2020)
2) [17] Тестирование. Фундаментальная теория. URL: https://habr.com/ru/post/279535/ . (дата звернення 15.11.2020)

використанням автоматизованих інструментів для загальних імен файлів та каталогів.

Ця атака також відома як передбачуване розташування ресурсів, перерахування файлів, перелік каталогів та перелік ресурсів.

### 3.1.5 Web scraping

Веб-скрейпінг (або скрепінг, або скрапінг ← англ. Web scraping) - це технологія отримання веб-даних шляхом вилучення їх зі сторінок веб-ресурсів. Веб-скрейпінг може бути зроблений вручну користувачем комп'ютера, однак термін зазвичай відноситься до автоматизованих процесів, реалізованим за допомогою коду, який виконує GET-запити на цільовий сайт. Веб-скрейпінг використовується для синтаксичного перетворення веб-сторінок в більш зручні для роботи форми. Веб-сторінки створюються з використанням текстових мов розмітки (HTML і XHTML) і містять безліч корисних даних в коді. Однак більшість веб-ресурсів призначені для кінцевих користувачів, а не для зручності автоматичного використання, тому була розроблена технологія, яка «очищає» веб-контент. Завантаження і перегляд сторінки - найважливіші складові технології, вони є невід'ємною частиною вибірки даних

## 3.1 Робота з OWASP ZAP

ZAP може обробляти широкий спектр механізмів автентифікації. Метод автентифікації, який визначає спосіб обробки автентифікації. Аутентифікація використовується для створення веб-сеансів, які відповідають автентифікованим користувачам веб-додатків.

Стратегія перевірки автентичності, яка визначає, як ZAP повинен виявляти, коли повідомлення відповідають аутентифікованим запитам [17][1)].

### 3.2.1 Вимоги для встановлення

Нижче представлена таблиця, в якій показані основні вимоги до системи для подальшого встановлення АСУ НЗ, див.табл.1.

Таблиця 1 – Вимоги до системи для встановлення АСУ НЗ

| | Мінімальні вимоги | Рекомендовані вимоги |
|---|---|---|
| Processor | Pentium<br>або порівнянний | Intel Core i3 або порівнянний |
| RAM | 2 GB | 4 GB або більше |
| Hard drive space | 500MB | 1 GB або більше |
| Graphics | XGA (1024 x 768) | HD1080 (1920 x 1080) або вище |

Программа підтримується в наступних операційних системах (як 32-розрядних, так і 64-розрядних):

- Windows 10;
- Windows 8;
- Windows 7 SP1;
- Windows Vista SP2 [4][2)].

Повинний бути забезпечений проксі.

- від несанкціонованого доступу;

---

[1)] [17] Тестирование. Фундаментальная теория. URL: https://habr.com/ru/post/279535/ . (дата звернення 15.11.2020)

[2)] [4] Website Vulnerability Scanner. URL: https://pentest-tools.com/website-vulnerability-scanning/website-scanner (дата звернення 26.08.2020)

– від руйнування або зупинки роботи програмного забезпечення в результаті некоректних дій оператора технологічного процесу;
– від проникнення вірусів у «систему».

В «системі» повинні бути апаратні та апаратно-програмні засоби діагностики мереж, станцій, блоків і модулів.

### 3.2.2 Початкові налаштування

Після запуску сканера, всі знайдені помилки OWASP ZAP будуть відсортовані під різні серйозності уразливостей і будуть знаходиться у вкладці «Оповіщення» що показано на рисунку 6. Під червоний флажек потраплять найсерйозніші, такі як XSS, SQL injaction. Під помаранчевий менш серйозні, типу CSRF і тд. Ну і інші незначні на, які мало хто звертає увагу, хоча варто було б.

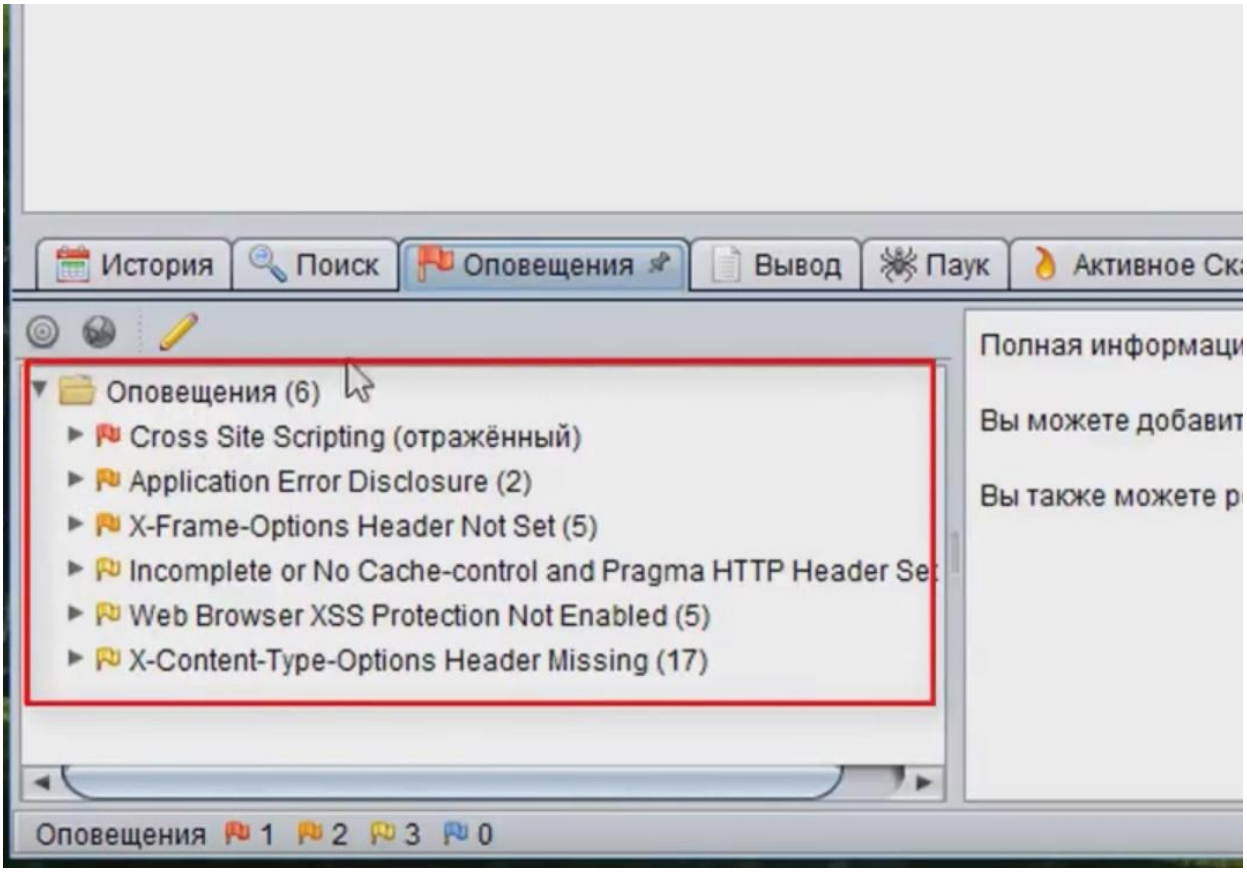



Рисунок 6 – Вікно OWASP ZAP з загрозами

Щоб подивитися більш детально знайдені уразливості, клікнувши кілька з якоїсь з уразливостей.

При цьому відкриється попап в якому буде розписано:

- що це за вразливість і на, що вона впливає
- її критичність
- де її можна відтворити
- література як її можна полагодити

Якщо у проекті присутній авторизація, то ті посилання, які доступні для авторизованих користувача, стануть недоступними для OWASP ZAP без налаштованого користувача, під яким зможе зайти наш додаток для сканування посилань всередині.

Для того, щоб швидко налаштувати цього користувача, натискаємо на іконку браузера див.рис. 7. При цьому відкриється сам браузер, в якому введені настройки проксі для прийому додатком OWASP.

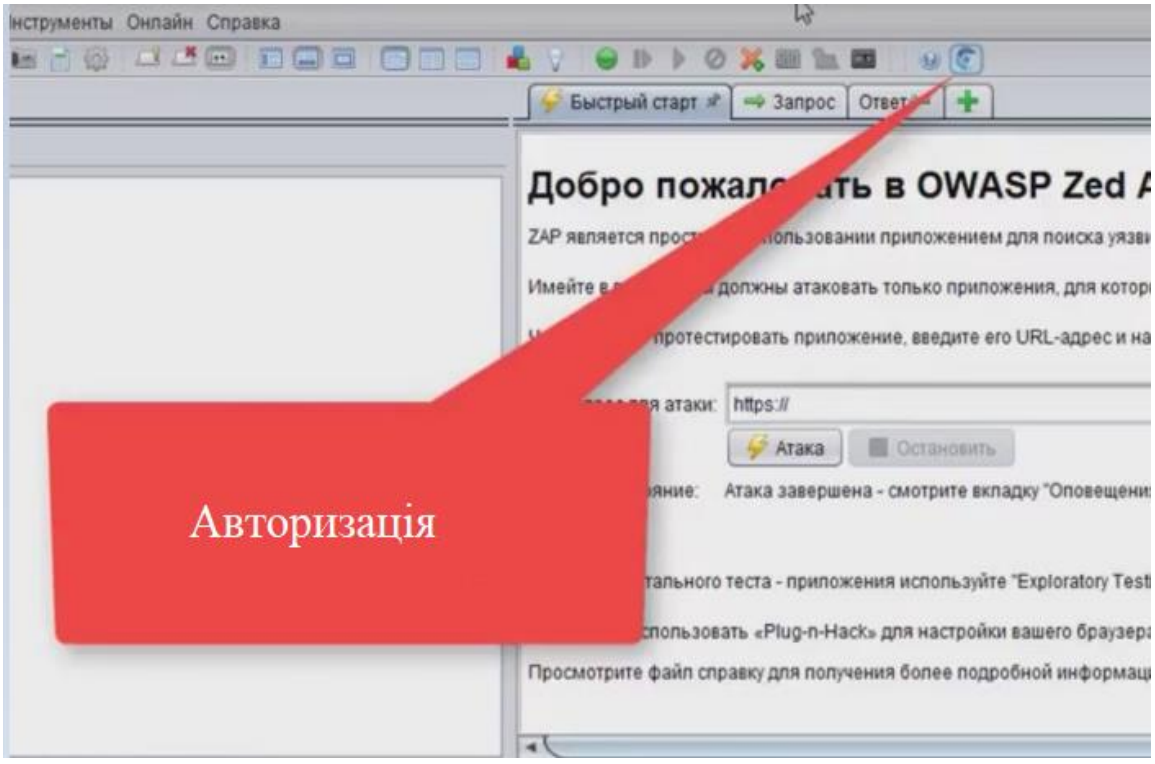



Рисунок 7 – Кнопка з авторизацією

Потім, щоб наш додаток на сканувати всі що в нього потрапить (Зайві ресурси), що не стосується нашого проекту, потрібно задати "контекст" тільки для однієї папки, яку ми хочемо перевірити. Для цього вибираємо потрібну папку і включаємо її в контекст див.рис. 8.

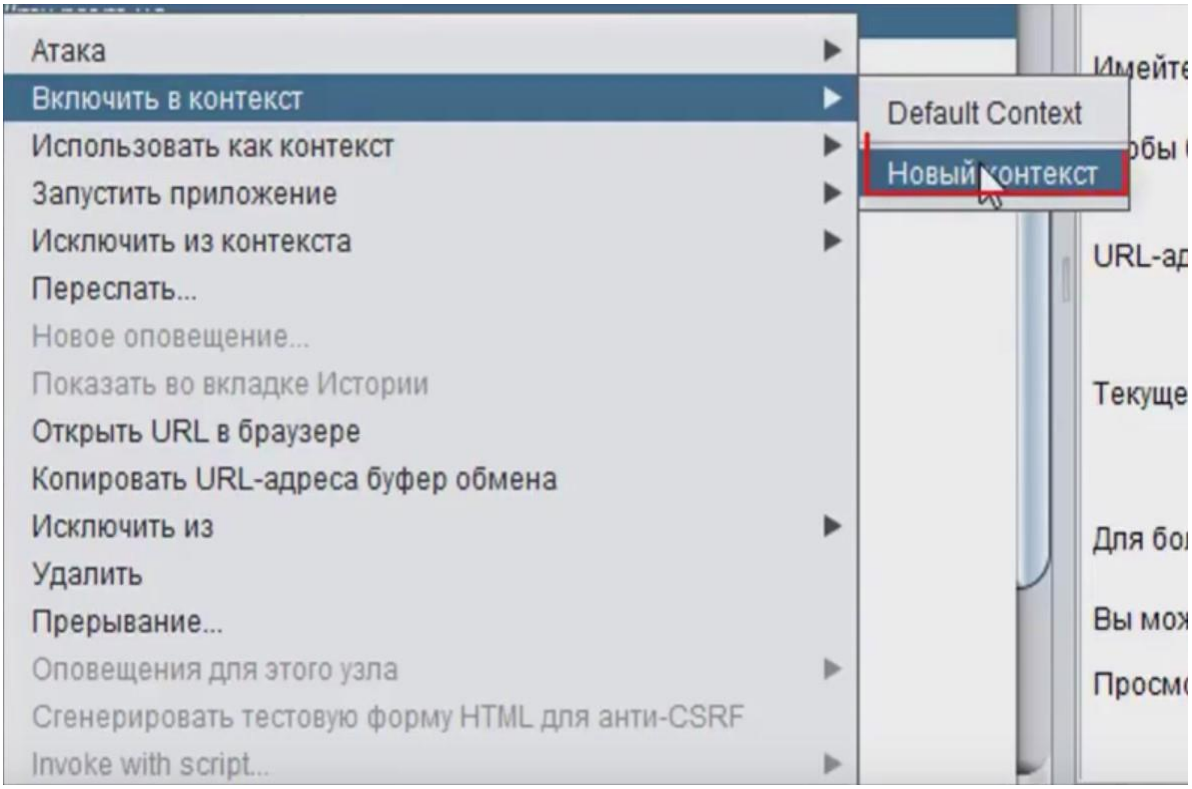



Рисунок 8 – Створеня нового контексту

Після виконаних операцій створюємо користувача з милом і паролем, щоб він міг авторезіроватся, при скануванні нашого проекту. Для цього заходимо в розділ користувачів. І в цьому розділі додаємо нашого користувача як на рисунку 9, який зможе авторизуватися.

Тепер треба показати OWASP ZAP якийсь локатор (Xpath) на сторінці авторизованого користувача, щоб вона розуміла, що авторизація пройшла успішно. Для цього нам треба зайти на запит, який ми отримали від сервера, з html розміткою, яку отримує авторизовані юзер. Потім знаходимо якийсь локатор до якого прив'яжемо наш сканер. Цей локатор (рис.10) OWASP буде

шукати після виконання авторизації і розуміти він успішно чи пройшов авторизацію на сайті.

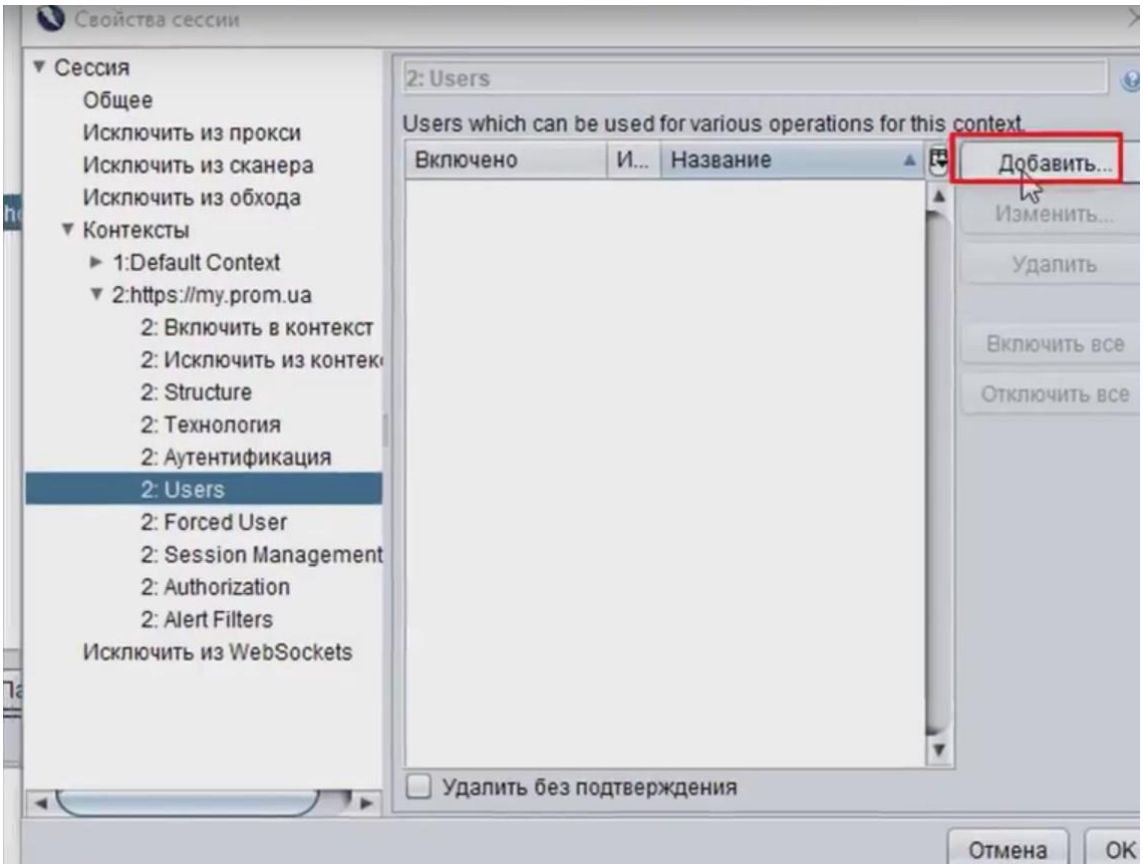



Рисунок 9 – Створення нового користувача.

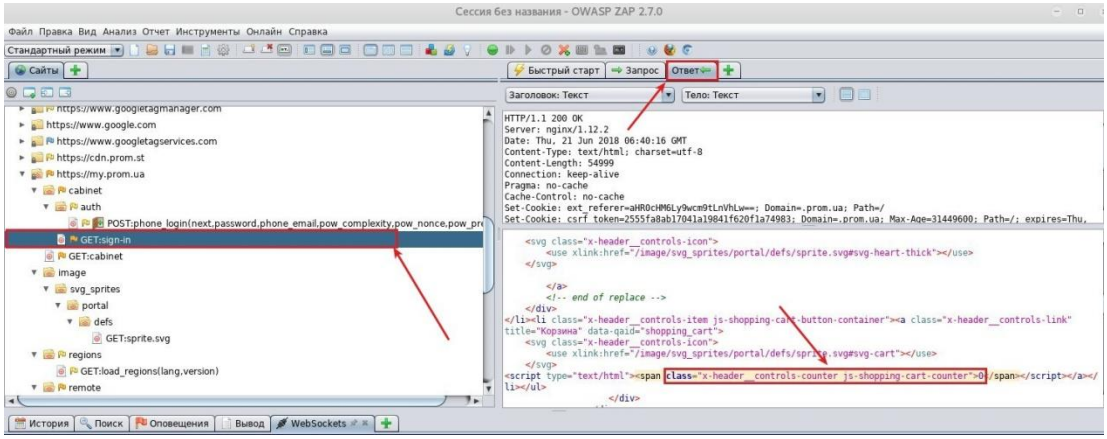



Рисунок 10 – Вікно зв’язків елементів на прикладі локатора

Цей локатор необхідно вставити в розділ аутентифікації, в якому ми задавали параметри входу. У інпут Regex pattern indentifield in Logged як на рисунку 11.

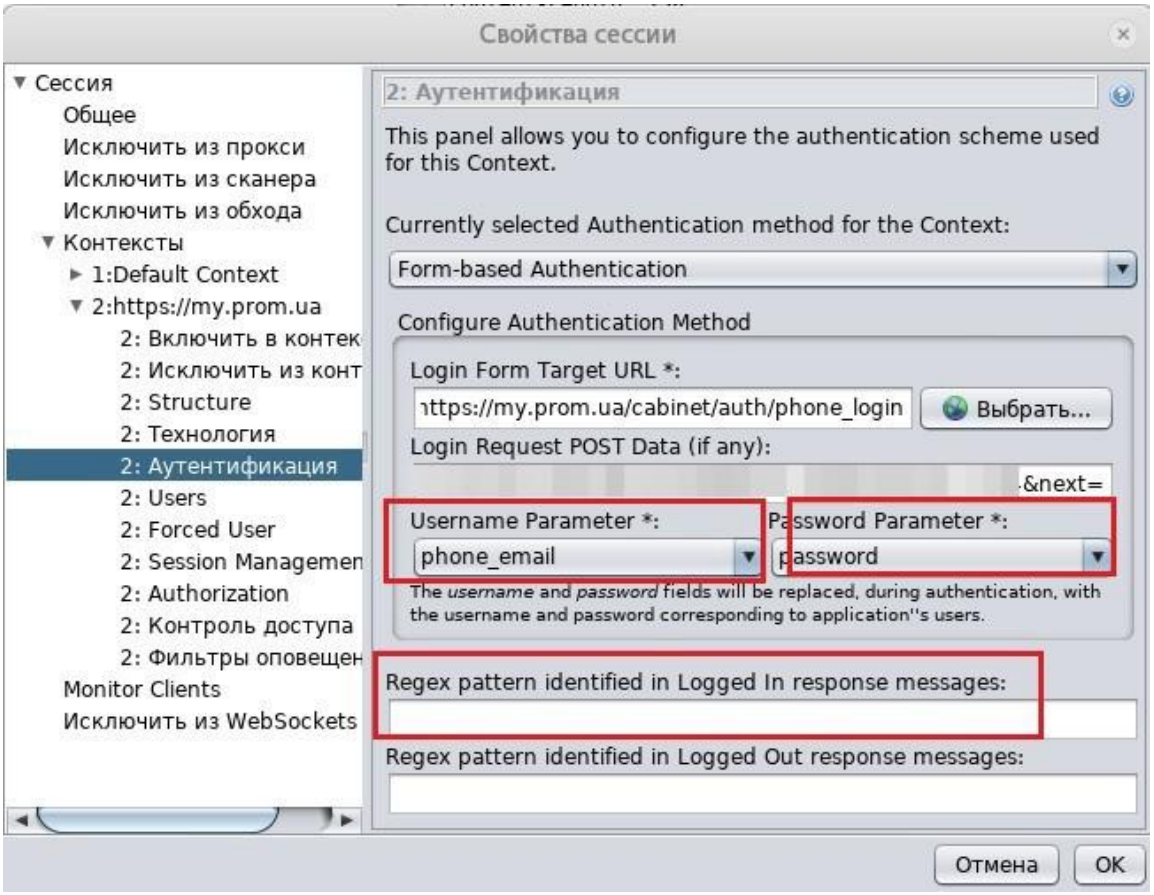



Рисунок 11 – Налаштування аунтефікації

Після цього можна починати атаку і перевірку на, що здатне наше додаток. Для цього натискаємо по нашій папці, яку задали контекст і тиснемо скан або атака, див.рис. 12.

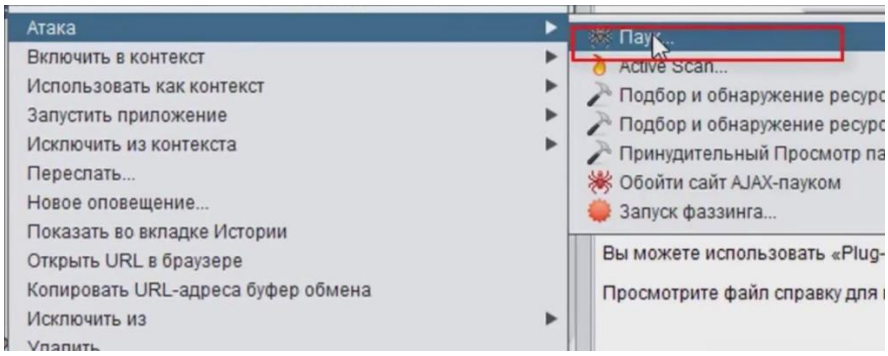



Рисунок 12 – Початок атаки павуком

Після цього в списку знаходимо необхідного користувача, див.рис. 13.

Далі починається сканування проекту. Не слід забувати що не всі уразливості можуть бути знайдені за допомогою павука, необхідно використовувати і інші методи. У наступній главі буде ретельно проведена робота пошуку уразливостей.

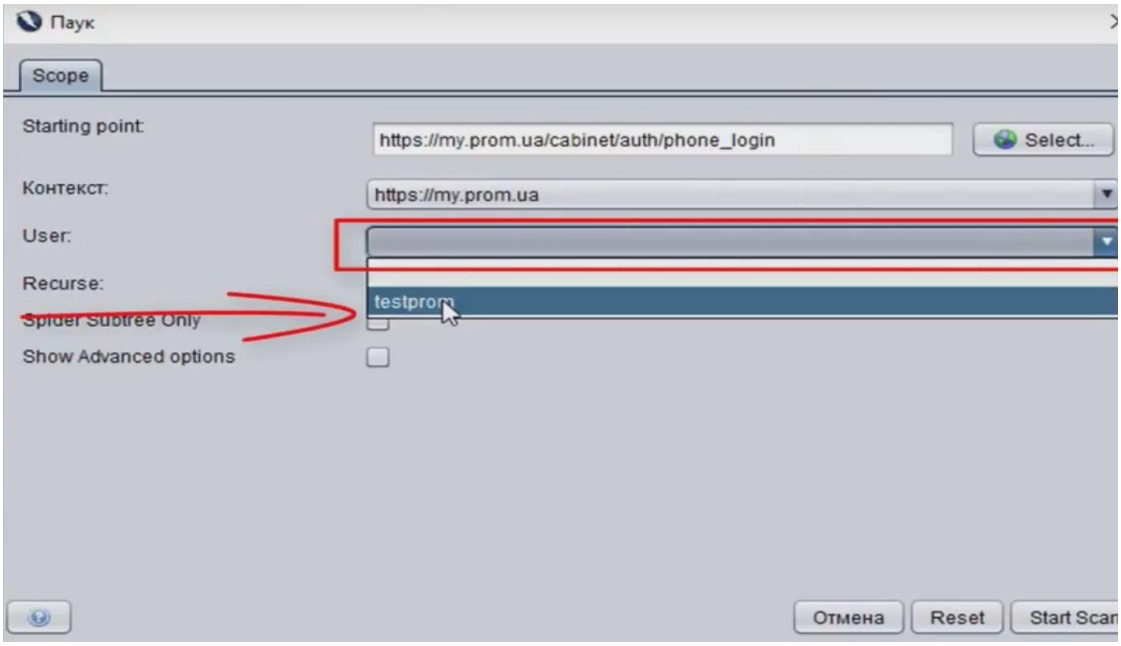



Рисунок 13 – Вибір необхідного користувача

# 4 ПРОВЕДЕННЯ ТЕСТУВАННЯ НА УРАЗЛИВІСТЬ

Етичність тестування безпеки повинна базуватись на правилах застосування (rules of engagement), яких повинен дотримуватися аудитор, котрого наймає організація для проведення тестування на проникнення до її інформаційних ресурсів, зокрема: як слід проводити тестування; визначення масштабів тестування; підготовка плану тестування; перебіг процесу тестування; забезпечення конфіденційної звітності по проведеній роботі тощо. Загальновідомі два підходи для проведення тестування на проникнення (далі – пентесту) Black-Box та White-Box. Black-Box пентест також відомий як зовнішнє тестування [18][1)].

Пентест (тестування на проникнення) – імітація реальних кібератак для перевірки комп'ютерної безпеки, пошуку та усунення уразливостей, які можуть викликати некоректну роботу або відмову системи. Під час тестування проводиться аналіз, який дозволяє оцінити захист з позицій атакуючого, виділити слабкі місця і дати рекомендації щодо їх усунення.

При застосуванні цього підходу аудитор безпеки буде оцінювати мережеву інфраструктуру організації з віддаленого місця розташування і не бути знати всі внутрішні технології, які тут використовуються. Насправді в цьому підході аудитор (Black Hat) уподібнюється поведінці зловмисників і застосовує усі відомі йому хакерські техніки та інструментальні засоби. При цьому важливо зрозуміти та класифікувати усі знайдені уразливості у відповідності з рівнем ризику (низький, середній або високий). Ризик, в цілому, може бути виміряний відповідно до загрози через виявлену вразливість і відповідні втрати, які сталися після успішного проникнення. \

[1)] [18] Про Тестинг – Тестирование – Виды Тестирования ПО – Тестирование безопасности. URL: http://www.protesting.ru/testing/types/security.html . (дата звернення 17.11.2020)

По завершенні пентесту створюється звіт з усією необхідною інформацією щодо оцінки рівня безпеки мережевої інфраструктури організації, класифікацією усіх виявлених ризиків в бізнес-контексті. White-Box пентест також відомий як внутрішнє тестування. Аудитор (White Hat) при цьому повинен бути в курсі будови інфраструктури мережі та усіх наявних сервісів організації. White-Box пентест подібний до того, як проводиться Black-Box пентест, але немає потреби проводити такі етапи як визначення меж тестування, збір інформації про цільову систему та виявлення працюючих сервісів на цільових хостах. White-Box пентест дозволяє виявити усі уразливості в системі та їх усунути, що природно підніме рівень захищеності системи в цілому[18][1)]. Крім того, цей підхід може бути легко інтегрований в звичайний цикл розробки продуктів, що випускає організація, що дозволить викорінити будь-які можливі проблеми з безпекою на ранній стадії, перш ніж вони будуть розкриті і використані зловмисниками, див.рис.14.

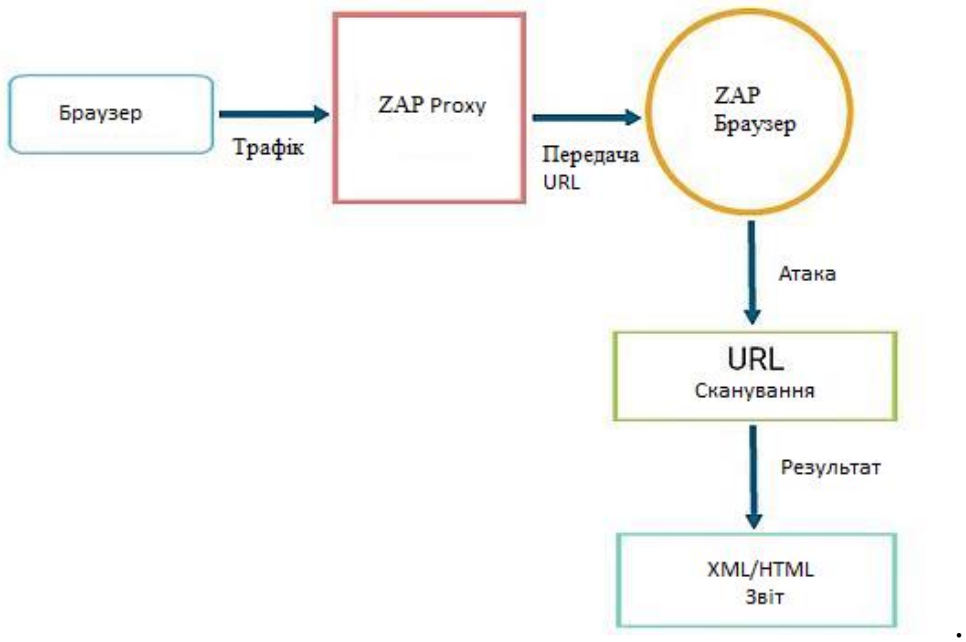



Рисунок 14 – Схема зв’язків ZAP

[1)] [18] Про Тестинг – Тестирование – Виды Тестирования ПО – Тестирование безопасности. URL: http://www.protesting.ru/testing/types/security.html . (дата звернення 17.11.2020)

## 4.1 Дослідження за допомогою OWASP ZAP.

Проведення пентестингу можливе в двох форматах. У першому випадку фахівці, які проводять тестування, не мають інформації про пристрій захисту і структурі мережі. Другий варіант передбачає попереднє ознайомлення виконавців з системою захисту. Є також проміжні схеми, коли інформація відома частково. Оптимальним вважається поєднання двох методів, яке дає максимум даних. Щоб спростити роботу пентестерам, використовують стандартизовані методології GWAPT, OWASP, IEM.

Перш ніж почати тестування, пентестеру необхідно розуміти структуру і принципи функціонування програми, як користувач і браузер взаємодіють з додатком. У міру вивчення веб-додатки, тестувальник повинен приділяти пильну увагу всім HTTP запитами (GET і POST) а також параметрами і полях форм, переданим в запитах. Крім цього, слід звернути увагу на те, в якому випадку при передачі параметрів в додаток використовуються GET запити, а в якому - POST запити. Звичайна практика - для взаємодії з додатком використовувати GET запити, але при цьому важлива інформація користувача передається в тілі POST запиту.

Для того щоб побачити параметри POST запиту, тестувальників необхідно використовувати спеціальний інструмент - intercepting proxy (перехоплює проксі), (наприклад, OWASP Zed Attack Proxy (ZAP) - посилання) або спеціальний плагін браузера. Також при аналізі параметрів POST запиту, необхідно особливу увагу звертати на приховані поля форм, що передаються в запиті, зазвичай такі параметри містять важливу інформацію: стан, кількість предметів, ціну - параметри, які за задумом розробника не повинні бачити або змінювати.

З досвіду  на цій стадії тестування дуже зручно використовувати перехоплюване проксі і табличний редактор. Проксі зберігає кожен запит і відповідь на нього в міру того, як ви досліджуєте веб-додаток. Також, маючи всі запити і відповіді від програми, тестувальник може бачити кожен

заголовок, кожен параметр і т.д., що передаються в додаток і повертаються користувачу. Такий докладний процес вивчення веб-додатки може бути досить стомлюючим і нудним, особливо на великих інтерактивних сайтах (банківський додаток). Однак, досвід підкаже на що необхідно звертати пильнішу увагу і даний етап може бути значно скорочений.

У міру вивчення веб-додатки, тестувальників слід позначати різні цікаві параметри запитів, заголовки і зберігати їх в таблицю. У таблицю також слід включати URL запитаної сторінки (непогано буде включити номер запиту з проксі для подальшого аналізу), цікаві та незвичайні параметри запитів, тип запиту (POST, GET), потрібно чи аутентифікація, чи використовується SSL, чи є запит частиною будь-якого процесу, що складається з декількох кроків (процес відновлення пароля). Після того як тестувальник задокументував усі розділи веб-додатки, можна приступати до тестування кожного розділу. Інша частина цього гайда присвячена опису тестування різних розділів веб-додатки, але даний крок за визначенням всіх ключових розділів програми повинен бути завершений перед початком докладного тестування.

Після запуску програми, необхідно обрати веб-сайт за допомогою url-посилання або ip-адреси. Для цього розглянемо веб-сайт http://testphp.vulnweb.com/. Для цього слід обрати вікно “Manual Explore” та після вставлення посилання натиснути Launch, як показано на рисунку 15.

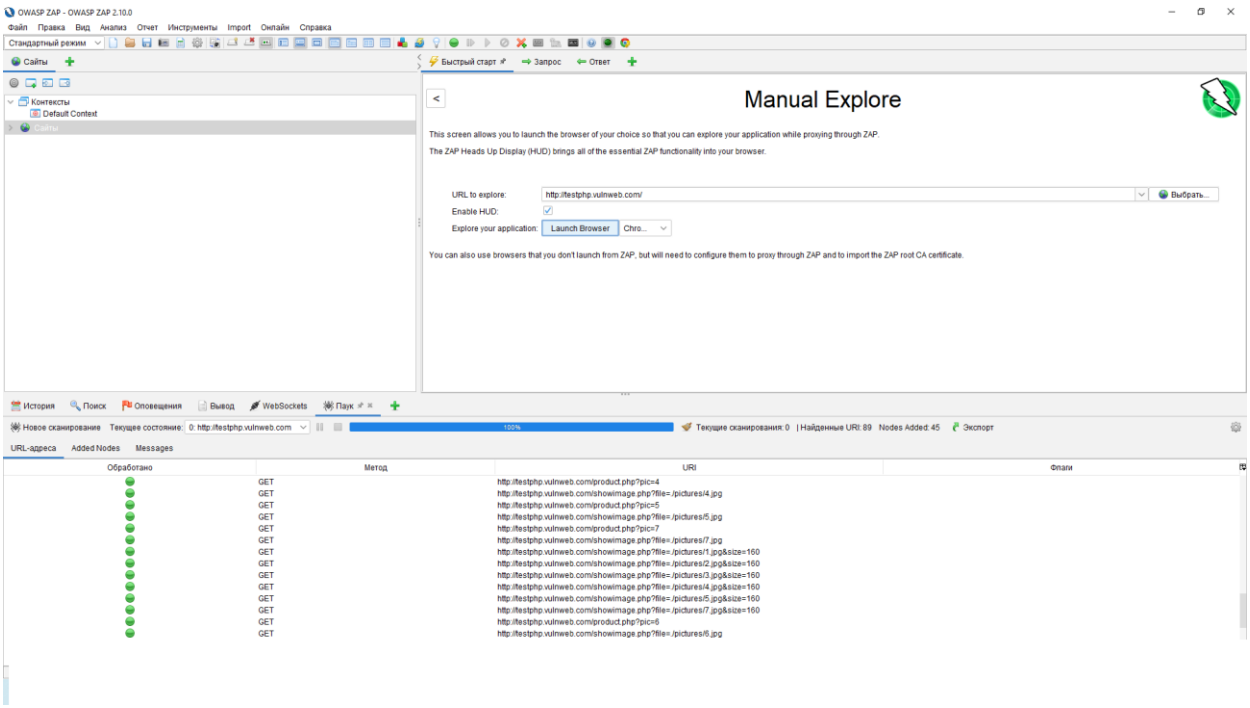



Рисунок 15 – Вибір веб-посилання

Після цього відкриється браузер де можна побачити сам сайт, його можна побачити на рисунку 16. Та відкриється новий інтерфейс по бокам.

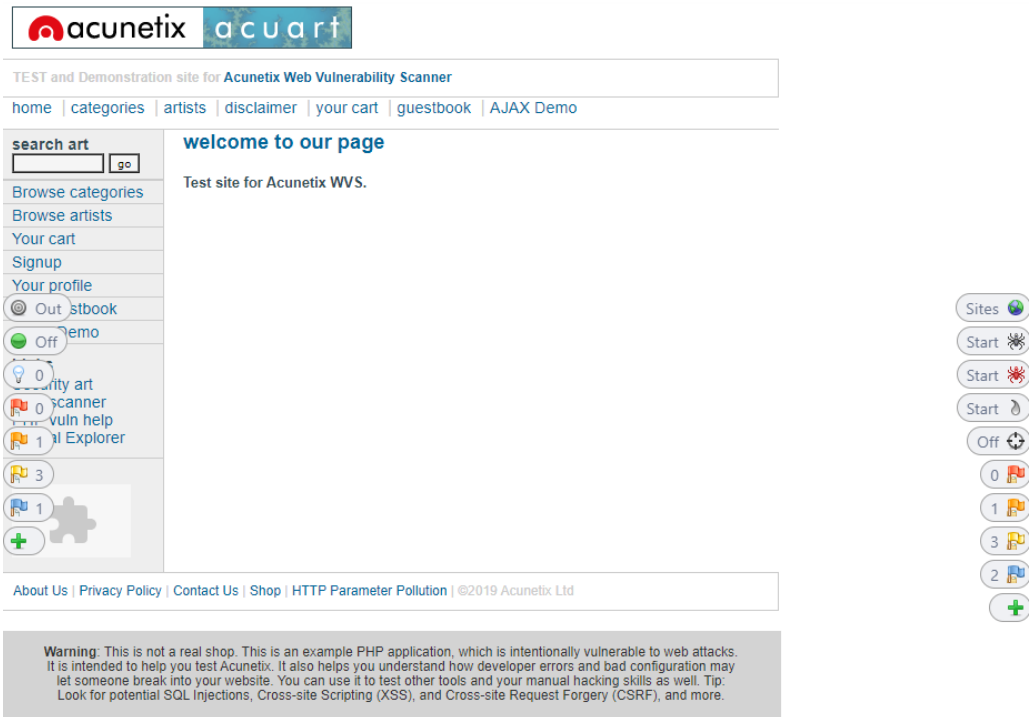



Рисунок 16 – Веб-сторінка з додатковим інтерфейсом

Отже після цього сайт з’явиться у вікні сайтів (рисунок 17) .

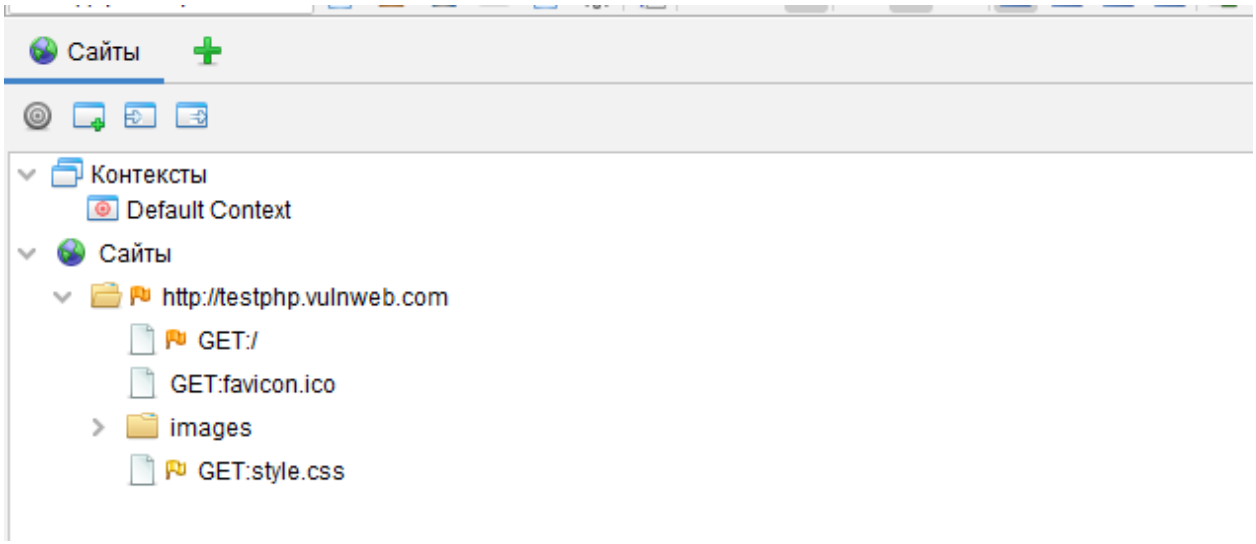



Рисунок 17 – Вікно з усіми сайтами

Далі необхідно почати сканування, наприклад за допомогою павука див.рис.18.

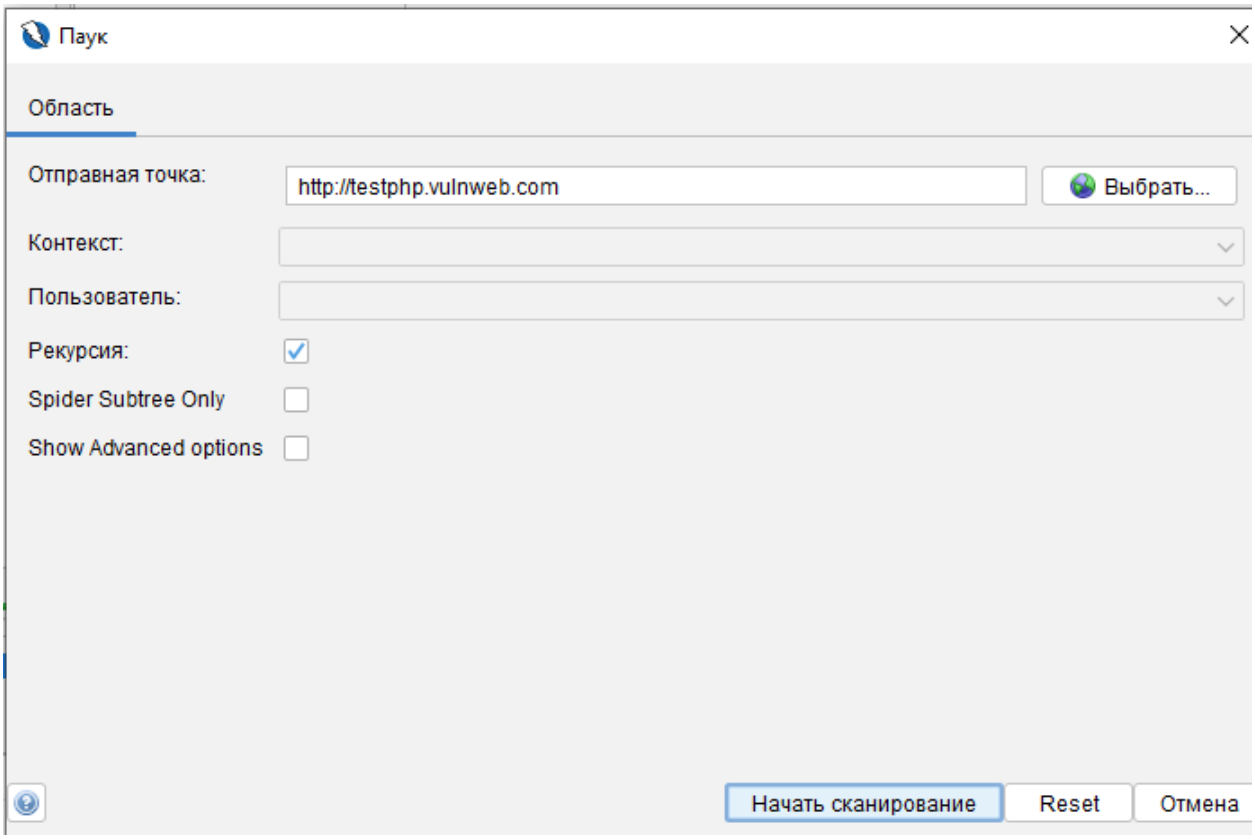



Рисунок 18 – Атака за допомогою інструмента «павук»

Павук почне перебір веб-ресурсів в суворій послідовності, індексацію нових сторінок, включення невідомих сайтів в базу. І після вся інформація зьявиться у вікні інтерфейсу, як на рисунку 19.

Без наявності активного захисту від фіксації сесії, ця атака може бути використана проти будь-якого сервера, аутентифікуючи користувачів за допомогою ідентифікатора сесії.

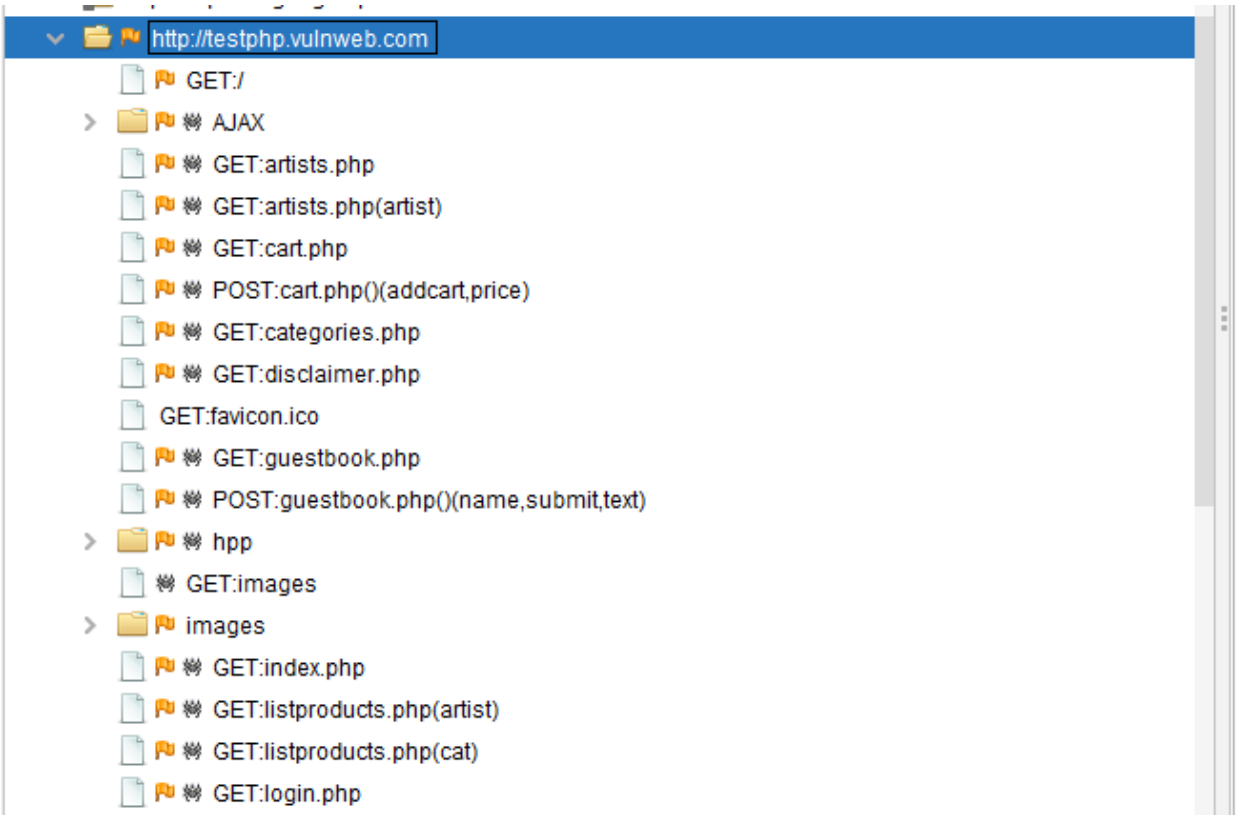



Рисунок 19 – Знайдені павуком посилання

Хоча сама программа показала що на сайті не знайдено серйозних проблем, він на спраді їх мав. Було знайдену цікаву папку “notes”, та запит до passwords ,див.рис.20.


Рисунок 20 – Знайдений запит до passwords

Після переходу за цим запитом, відкрилась вкладка з файлами зображеними на рисунку 21.

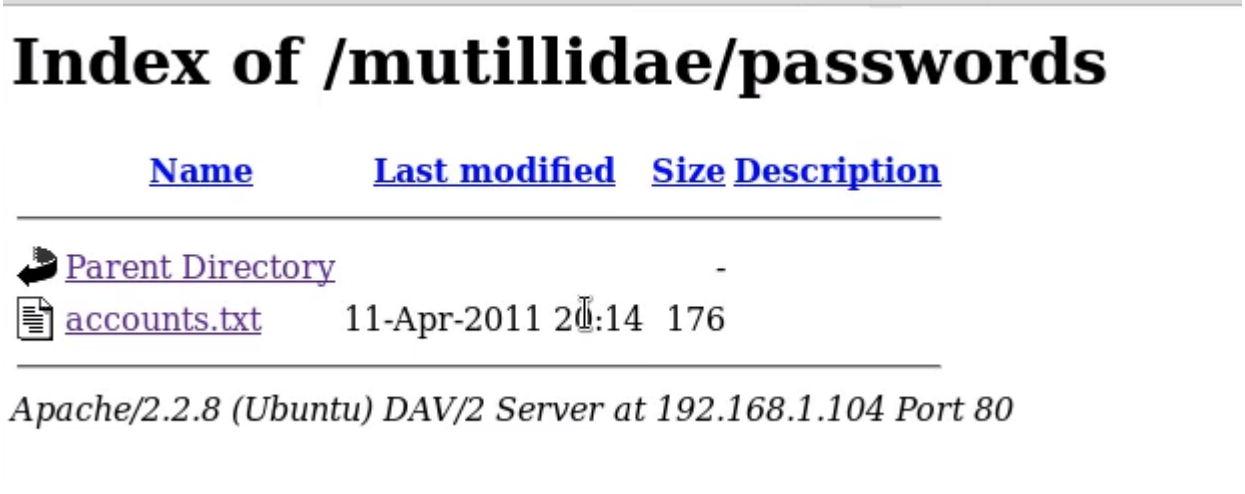



Рисунок 21 – Веб-сторінка за запитом notes:GET:passwords

Оскільки файл accounts.txt мав гіперпосилання, після його відкриття були знайдені деякі аккаунти, паролі, та інформація щодо них. В котрих я знайшов навіть свій аккаунт що був створенний для тесту на рисунку 22.

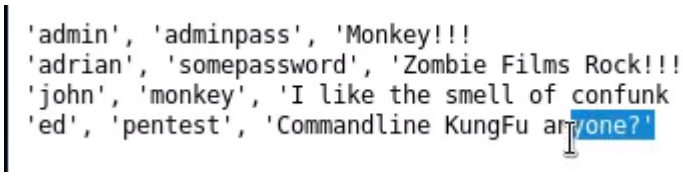



Рисунок 22 – Вигляд данних файлу accounts.txt

Також цікавим був наступний запит пов’язаний з базою даних, що є дуже кретичним, див.рис.23.


Рисунок 23 – Запит до БД

Після його відкриття в браузері виникло наступне вікно, див. рис. 24.

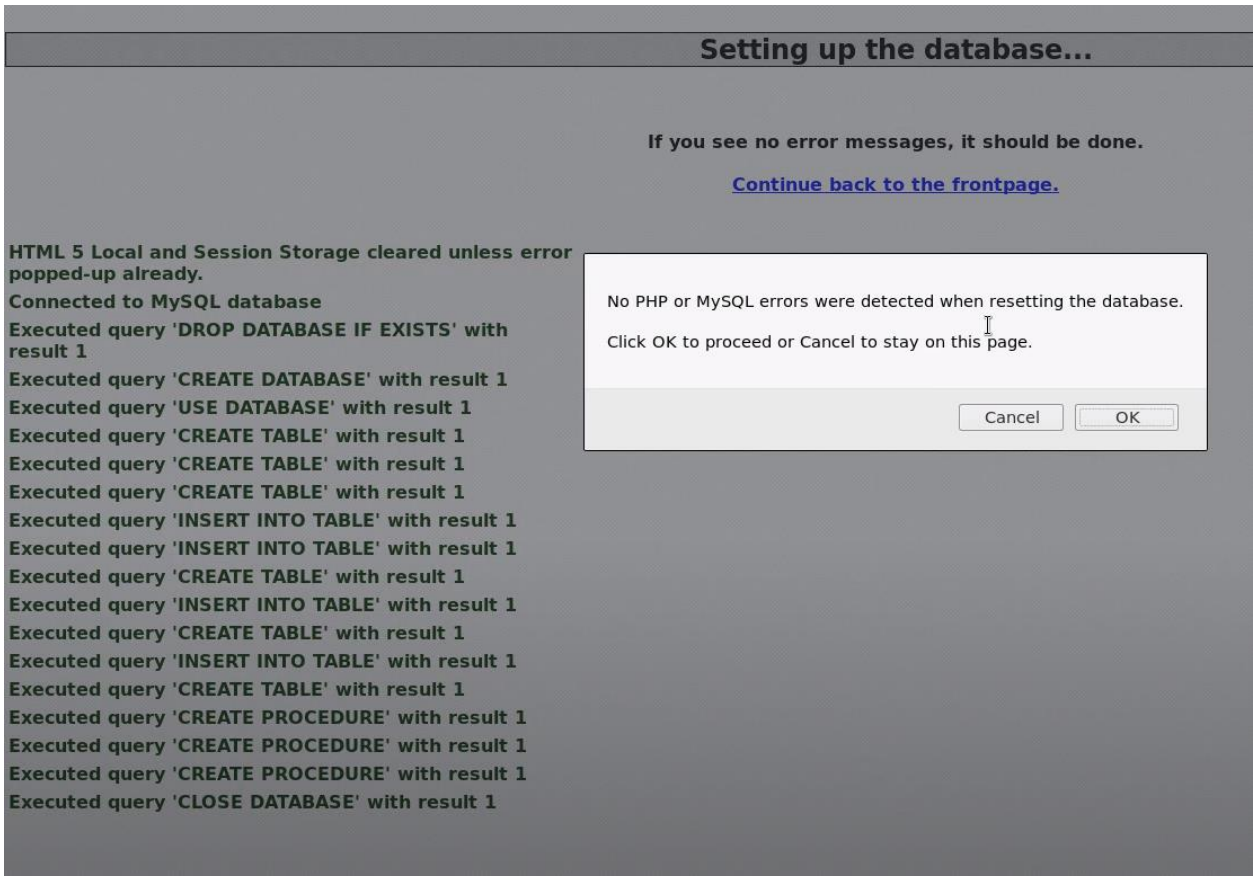



Рисунок 24 – Вікно видалення БД

За допомогою нього можна скинути базу до її початкової форми. Це демонструє те що після цієї дії логін та пароль від мого аккаунту вже не працював.

Також після того як я натиснув пошук на сайті, в программі зьвився новий запит на той-же пошук, див. рис. 25.

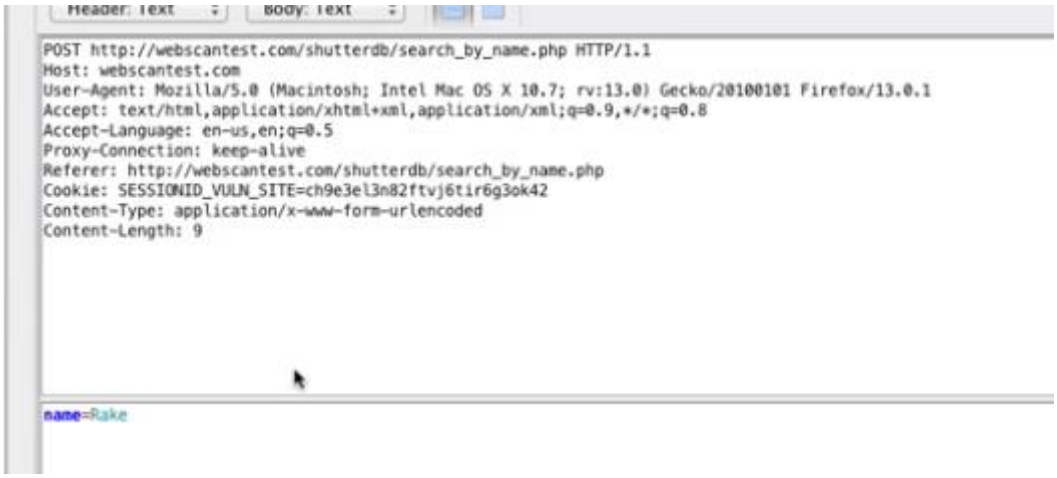



Рисунок 25 – Вікно з результату запиту на пошук

Після чого я вирішив використати Fuzz, для пошуку уразливості на sql-інєкцію. див.рис. 26.. У цьому випадку стандартні повідомлення про помилки модифіковані, і сервер повертає зрозумілу для користувача інформацію про неправильне введення. Здійснення SQL Injection може бути здійснене і в цій ситуації, проте виявлення уразливості утруднене. Найбільш поширений метод перевірки наявності проблеми – додавання виразів, повертають істинне і помилкове значення.

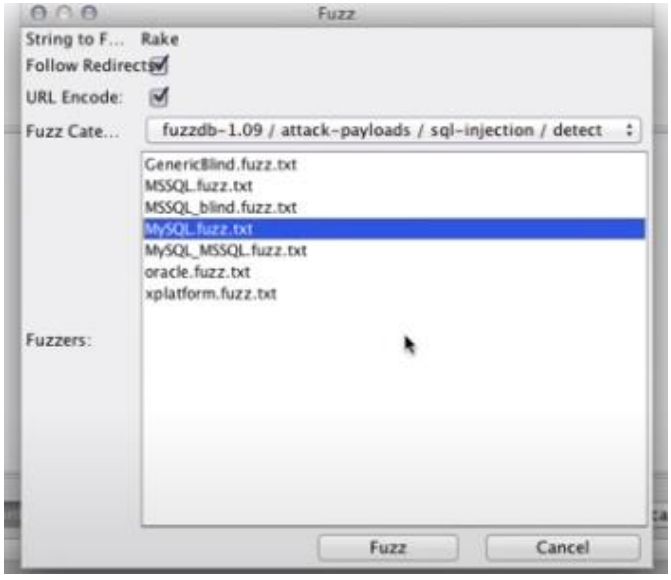



Рисунок 26 – Використання інструменту Fuzz

Після чого OWASP показав вразливість. Вони примічені жовтим кругом, див.рис.27

Ідея полягає в тому, щоб побудувати повідомлення про помилку, яке розкриває весь запит, і ввести корисне навантаження в оператор WHERE, який завжди повертає true. Такий запит може виглядати наступним чином:
SELECT * From user_data WHERE Login_Count = 0 and userid= 1 or 1=1;--.
Оскільки поля в запиті SQL є числовими, нам не потрібно використовувати жодні лапки. На наступному скріншоті показано результати експлуатації.

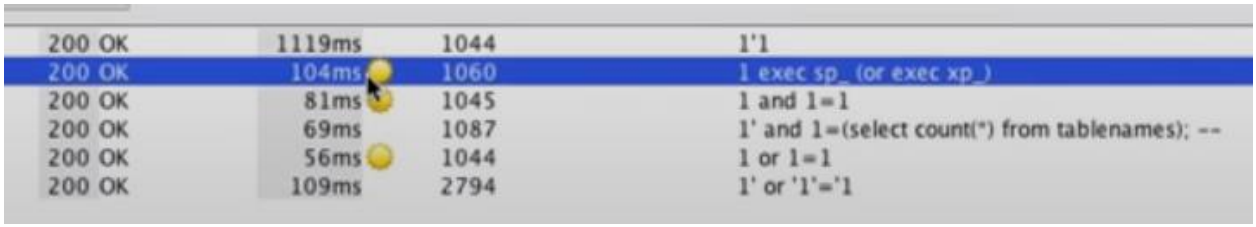
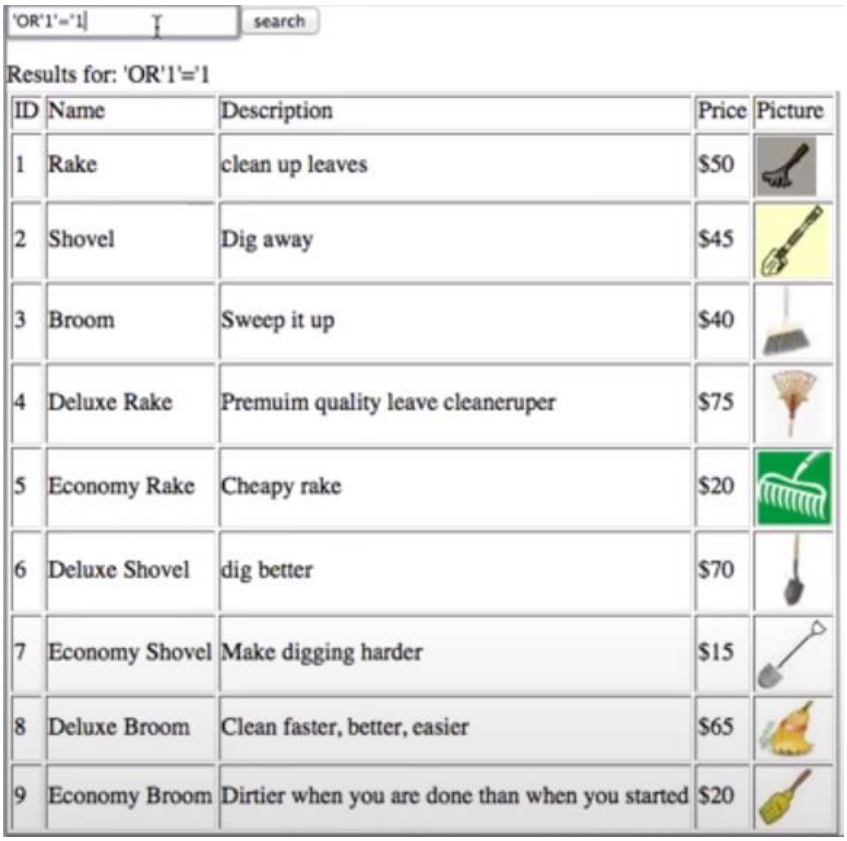



Рисунок 27 – Звіт о можливих уразливостях ін’єкції.

Тобто наш запит має бути одним ін. вище показаних, якщо не всі одразу. Лише з третьої спроби вдалося знайти повний список товарів на сайті, за допомогою третього варіанту “1’ or ‘1’=’1”, див.рис.28 .


'OR'1'='1 search

Results for: 'OR'1'='1

| ID | Name | Description | Price | Picture |
|---|---|---|---|---|
| 1 | Rake | clean up leaves | $50 | |
| 2 | Shovel | Dig away | $45 | |
| 3 | Broom | Sweep it up | $40 | |
| 4 | Deluxe Rake | Premuim quality leave cleaneruper | $75 | |
| 5 | Economy Rake | Cheapy rake | $20 | |
| 6 | Deluxe Shovel | dig better | $70 | |
| 7 | Economy Shovel | Make digging harder | $15 | |
| 8 | Deluxe Broom | Clean faster, better, easier | $65 | |
| 9 | Economy Broom | Dirtier when you are done than when you started | $20 | |

Рисунок 28 – Список товарі що був отриманий за допомогою уразливості.

Також сайт не мав ніякої капчи, чи ліміту на кіл-кість логінів за годину. Тому теоретично його можна зламати за допомогою брутфорсу. Це робиться дуже просто за допомогою OWASP Zap. Після невдачної спроби вводу випадкових данних у логін, зьявилось посилання в программі на рисунку 29.

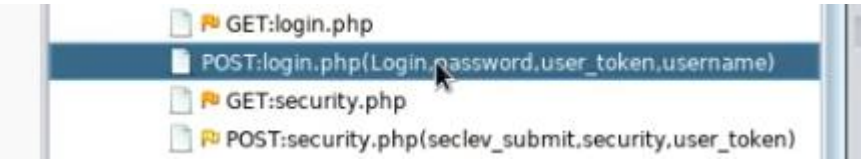



Рисунок 29 – Запит що з'явився після спроби логіну

Далі необхідно завантажити чи створити файл з перебором усіх можливих паролей та логінів. Я використовував уже створений знайдений

дуже просто в інтернеті, та додав до нього свої данні аккаунта. Його необхідно обрати, див.рис.30.

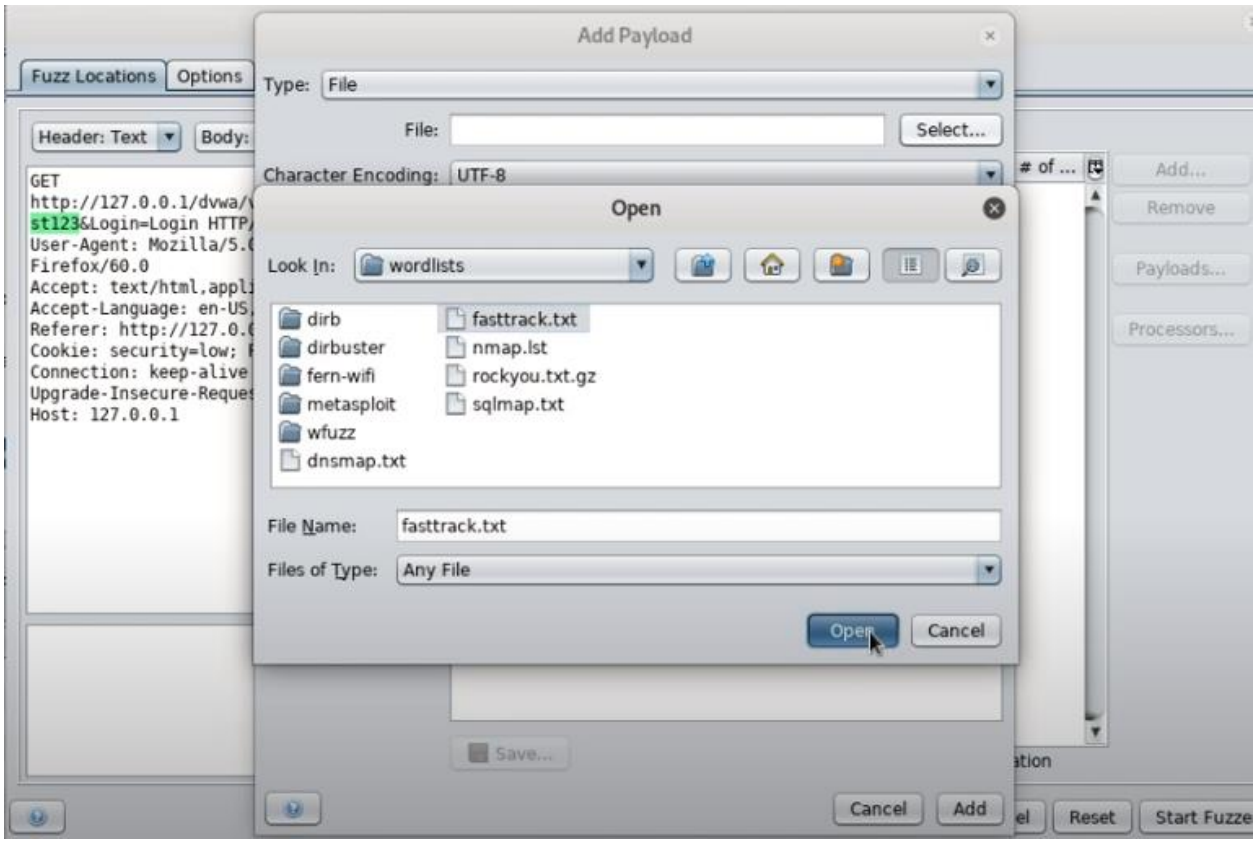
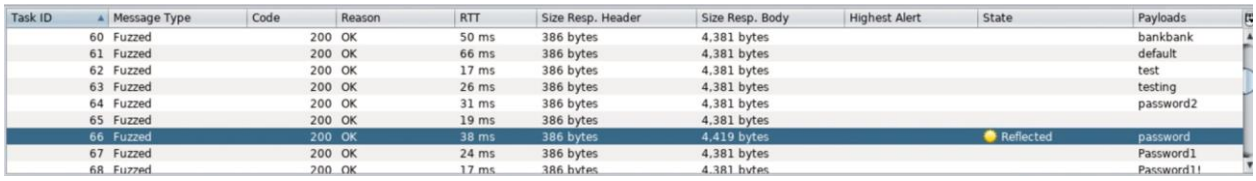



Рисунок 30 – Вибір файла для брутфорсу паролів.

Після вибору файла почав атаку, що призвела до того що один із запитів був успішним. Це було зображено в звіті програми, див.рис.31.



| Task ID | Message Type | Code | Reason | RTT | Size Resp. Header | Size Resp. Body | Highest Alert | State | Payloads |
|---|---|---|---|---|---|---|---|---|---|
| 60 | Fuzzed | 200 | OK | 50 ms | 386 bytes | 4,381 bytes | | | bankbank |
| 61 | Fuzzed | 200 | OK | 66 ms | 386 bytes | 4,381 bytes | | | default |
| 62 | Fuzzed | 200 | OK | 17 ms | 386 bytes | 4,381 bytes | | | test |
| 63 | Fuzzed | 200 | OK | 26 ms | 386 bytes | 4,381 bytes | | | testing |
| 64 | Fuzzed | 200 | OK | 31 ms | 386 bytes | 4,381 bytes | | | password2 |
| 65 | Fuzzed | 200 | OK | 19 ms | 386 bytes | 4,381 bytes | | | |
| 66 | Fuzzed | 200 | OK | 38 ms | 386 bytes | 4,419 bytes | | Reflected | password |
| 67 | Fuzzed | 200 | OK | 24 ms | 386 bytes | 4,381 bytes | | | Password1 |
| 68 | Fuzzed | 200 | OK | 17 ms | 386 bytes | 4,381 bytes | | | Password1! |

Рисунок 31 – Звіт уразливостей.

Звісно на сучасних сайтах за допомогою брутфорса це майже не можливо оскільки паролі тепер складніші а сайти мають ліміти на кількість

запитів та двухфакторну авторизацію, але все ж це є одним із ефективних методів якщо мати базу вже вкрадених паролів та логінів.

Далі можна протеступати AJAX-павука, що може показати вразливість cookies, або js. Для початку необхідно зайти на будь-який аккаунт, щоб створилась сесія. Це потрібно для того щоб знайти потрібний параметр що має данні щодо теперішніх сесій, адже сесія триває пока користувач знаходиться на сайті. Це можна використати в поганих цілях. Виконуємо логін до сайту та дивимось створені кукі файли на рисунку 31.

| Type | Name | Used | # Values | % Change | Flags | Values |
|---|---|---|---|---|---|---|
| Cookie | .ASPXAUTH | | 4 | 1 | 0 HttpOnly, path=/ | 3417DD8E3E44B02... |
| Cookie | __RequestVerificationToken_L1NxbGlNb2... | | 12 | 1 | 0 HttpOnly, path=/ | ODEgFajQujF89PKX... |
| FORM | Password | | 1 | 1 | 0 | adminadmin |
| FORM | RememberMe | | 1 | 1 | 0 | false |
| FORM | UserName | | 1 | 1 | 0 | admin |
| FORM | __RequestVerificationToken | | 1 | 1 | 0 | UkPpzuO3F2Z27tjG... |
| URL | ReturnUrl | | 2 | 1 | 0 | /SqliModernApp |

Рисунок 31 – Пошук точену сессії

Тут можна побачити що тут є два cookie, один виходячи з імя запрошує токен авторизації, а інший ASPXAUTH показує чи автерізований користувач, і з його параметру видно що він зашифрований. Отже далі позначаємо його в програмі що він відповідає за токен сесії, див.рис.32.

Далі я спробував запустити ajax-павука, сканувати уразливості та відслудковати вже позначений кукі, див.рис.33.

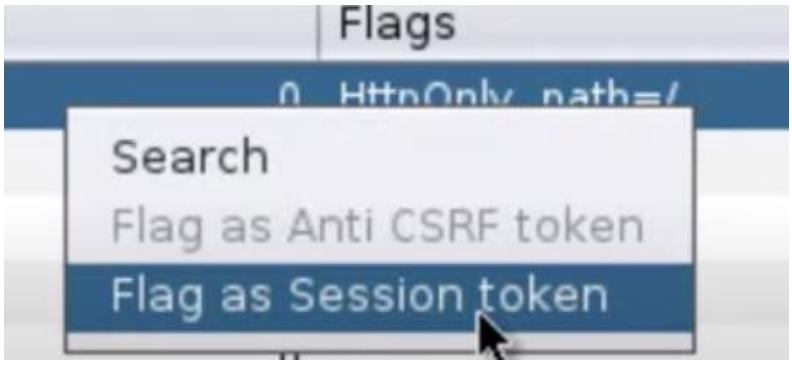



Рисунок 32 – Позначення токену

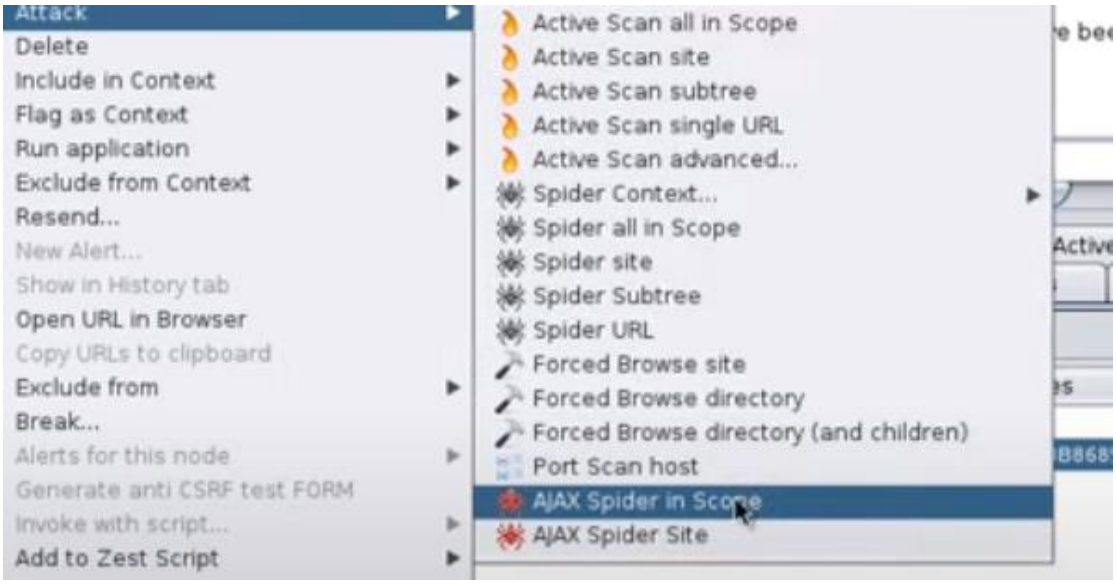



Рисунок 33 – Пошук за допомогою Ajax-павука

При попитці вийти з сайту, поле з номером сесії приймало значення null, це підтвердило мою догадку тому я додав ще одного павука, спеціально для цього параметру що він записав усі параметри при логіні та сесії. Після чого зьявився доступ до запитів з товарами, котрі працюють з БД. Дані атаки спрямовані на використання функцій Web-додатки з метою обходу механізмів розмежування доступу. Деякі механізми Web-додатки, включаючи функції забезпечення безпеки, можуть бути використані для цих цілей. Наявність уразливості в одному з, можливо, другорядних компонентів системи може призвести до компрометації всього додатку. Рівень ризику та потенційні можливості зловмисника в разі проведення атаки дуже сильно залежать від конкретного додатка, див.рис. 34.


Рисунок 34 – Знайдені запити

А це означає що за допомогою них можна використати ін’єкцію SQL. При повторному скануванні на уразливості программа видала попередження що це підтвердило, див.рис.35.

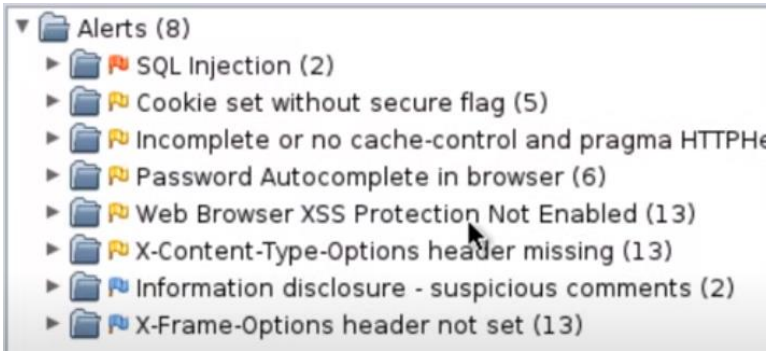



Рисунок 35 – Звіт зі знайденими небезпеками

Підводячи підсумки, було знайдено п’ять уразливостей на сайті. Вони приведені нижче у таблиці див.табл. 2.

Таблиця 2 – Уразливості веб-сайтів.

| Веб-сайт | Вразливість | Рівень загрози | Комментар |
|---|---|---|---|
| http://testphp.vulnweb.com/ | sql-injection | високий | уразливість в полі логіну |
| http://testphp.vulnweb.com/ | bruteforce | середній | перебір паролів за допомогою программи |
| http://testphp.vulnweb.com/ | cookie-leak | високий | виконанє закріплення за токеном |
| http://www.itsecgames.com/ | sql-injection | високий | уразливість в полі пошуку |
| http://www.itsecgames.com/ | crawler | низький | посилання без захисту с важливими даними |

# 5 РЕКОМЕНДАЦІЇ ЩОДО ПІДВИЩЕННЯ ЗАХИЩЕНОСТІ ВІД УРАЗЛИВОСТЕЙ

Під час проведення даного дослідження, результати якого представлені у розділі 4, можна зробити висновок, що розглянуты веб ресурси не надійно захищені. Проведений аналіз показав, що існують уразливості як високого так і малого рівня. Можливість sql-інєкції є кретичною вразливість, котра необхідна бути негайно вирішена.

Якщо, з системою скоїться якась неприємність, така як видалення або похилення БД, брутфорс атака, або вровадження злоякісного програмного забезпечення(віруси, які дають можливість зловмисникам віддалено підключитися до сервера та управляти ним), цє може поширити проблему від самого веб-ресурсу до його користувачів котрі стануть носіями такого шкідливого програмного забезпечення..

Отже починаючи з уразливості брутфорсу, необхідно:

- поширити мінімальну довжину паролю
- покращити складність паролю (додати обов’язків символ чи велику літеру)
- зробити ліміт спроб для входу на аккаунт
- використовувати капчу.
- додати двухфакторну авторизацію з нової ip адреси.

Наступна проблема з розповсюдженням важливих url-посилань, до котрих не має бути доступу у звичайних користувачів. Її можна вирішити за допомогою:

- впровадження системи доступу до посилань, тобто створа рівнів доступу та їх назначення для користувачів.
- перенос таких сторінок чи їх видалення.

- використовуючи можливості захисту ботів із глибоким поведінковим аналізом, щоб визначити поганих ботів та запобігти вискакуванню з інтернету

Далі для запобігання sql-ін’єкції рекомендовано:

- найкращим підходом є контроль та перевірка введення даних користувача для відстеження шаблонів атак
- профілактика пошуку таких уразливостей.
- не використовувати динамічний sql
- не використовувати запити при вводі користувача

Якщо казати про запобігання вкрадення cookie, то веб сайту необхідно:

- встановіть сертифікат ssl
- шифрувати дані перед їх передачею, тож навіть якщо хакеру вдається його вкрасти, вони не можуть прочитати дані.
- якщо використовується cms для управління веб-сайтом, то рекомендується встановити пагін безпек щоб виявляти та видаляти такі спроби злому безпосередньо перед тим, як вони завдають шкоди.

На основі отриманих результатів було розглянуто найнебезпечніші уразливості що кожного року є пречиною багатормільйардних збитків. Для усунення таких проблем необхідно їх вчасне знаходження. Для їх знаходження найефективнішим методом .є пінтестинг уразливостей, котрий і був використаний в дипломній роботі. Для дослідження Веб-сайтів на уразливості було обрано програму OWASP ZAP. Ця система дозволяє повною мірою відсканувати декілька сайтів одночасно, не витрачаючи на це велику кількість ресурсів комп’ютера. Це дає можливість швидко і масово сканувати всі необхідні веб-ресурси.

# ВИСНОВКИ

В даній роботі були розглянуті основні уразливості Web-сайтів та серверів і наведено їх приклади. Також було розглянуто ієрархію та рівні захисту Webсерверів. Далі були розглянуті механізми роботи засобів пошуку уразливостей. Після чого були вибрані поширені сканери уразливостей для подальшого аналізу їх можливостей.

Наступним кроком було проведення аналізу основних можливостей обраного сканера, а також вивчено їх функціонал.

Сканер представив детальний опис знайдених уразливостей, а також запропонував можливі рішення та перелік корисних посилань. Також, по результатам роботи обраних сканера були згенеровані звіт знайдених уразливостей.

Отримані результати сканувань можна легко використати для покращення роботи просканованих Web-сайтів, так як вони уже містять опис причин вразлвостей та шляхи боротьби з ними. Усунення знайдениних уразливостей неодмінно підвищить рівень безпеки перевірених сайтів.

В результаті аналізу уразливостей, були створені рекомендації для вирішення проблем та покращення загальної безпеки ресурсів.

## ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ


1. Актуальные киберугрози: 1 квартал 220 года. URL: https://www.ptsecurity.com/ru-ru/research/analytics/cybersecurity-threatscape-2020-q1/ (дата звернення 11.08.2020).
2. Уязвимости сайтов. URL: https://www.anti-malware.ru/threats/site-vulnerability (дата звернення 12.08.2020)
3. Внедрение команд ОС. URL: https://hackware.ru/?p=1133(дата звернення 25.08.2020)
4. Website Vulnerability Scanner. URL: https://pentest-tools.com/website-vulnerability-scanning/website-scanner (дата звернення 26.08.2020)
5. 10 WEB vulnerabilities you can prevent. URL: https://www.toptal.com/security/10-most-common-web-security-vulnerabilities (дата звернення 26.08.2020)
6. OWASP top 10 security vulnerabilities. URL: https://sucuri.net/guides/owasp-top-10-security-vulnerabilities-2020/ (дата звернення 27.08.2020)
7. Website vulnerabilities and tests. URL: https://cobweb-security.com/security_lessons/website-vulnerabilities-threats/ (дата звернення 27.08.2020)
8. Common website security vulnerabilities. URL: https://www.weblite.com.my/blog/website-security-vulnerabilities. (дата звернення 4.09.2020)
9. Website vulnerability scanner tool. URL: https://www.cybersecuritywebtest.com/website-vulnerability-scanning/website-scanner-tool. . (дата звернення 7.09.2020)
10. Website vulnerability scanner online. URL: https://www.acunetix.com/vulnerability-scanner/website-vulnerability-scanner-online/ . (дата звернення 10.9.2020)

11. Best 14 open-source web application vulnerability scanners URL: https://resources.infosecinstitute.com/topic/14-popular-web-application-vulnerability-scanners/ . (дата звернення 12.09.2020)
12. Тестирование на XSS и другие уязвимости с помощью OWASP ZAP. URL:https://medium.com/@svyatoslavlogyn/%D1%82%D0%B5%D1%81%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BD%D0%B0-xss-%D0%B8-%D0%B4%D1%80%D1%83%D0%B3%D0%B8%D0%B5-%D1%83%D1%8F%D0%B7%D0%B2%D0%B8%D0%BC%D0%BE%D1%81%D1%82%D0%B8-c-%D0%BF%D0%BE%D0%BC%D0%BE%D1%89%D1%8C%D1%8E-owasp-zap-a99183c32013 (дата звернення (27.09.2020)
13. PentestBox. URL: https://pentestbox.org/ . (дата звернення 13.9.2020)
14. Ethical hacking and cross-site scripting. URL: https://www.tutorialspoint.com/ethical_hacking/ethical_hacking_cross_site_scripting.htm (дата звернення 13.09.2020)
15. Лучшие дистрибутивы для проведения тестирования на проникнове-ние. URL: https://habr.com/ru/post/276477/ . (дата звернення 06.10.2020)
16. PentestBox. URL: https://pentestbox.org/ . (дата звернення 13.10.2020)
17. Тестирование. Фундаментальная теория. URL: https://habr.com/ru/post/279535/ . (дата звернення 15.11.2020)
18. Про Тестинг – Тестирование – Виды Тестирования ПО – Тестирование безопасности. URL: http://www.protesting.ru/testing/types/security.html . (дата звернення 17.11.2020)