МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Сумський державний університет

Факультет електроніки та інформаційних технологій

Кафедра кібербезпеки

«До захисту допущено»

Завідувач кафедри

_____ Володимир ЛЮБЧАК
(підпис)

____ _____ 20__ р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня бакалавр

зі спеціальності 125 Кібербезпека, освітньо-професійної програми Кібербезпека
на тему: Порівняльний аналіз захищеності систем керування вмістом
Здобувача групи КБ-01 Опихайленко Богдана Юрійовича

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

_____ Богдан ОПИХАЙЛЕНКО
(підпис)

Керівник: ст.викладач, к. е. наук, доцент Сергій ВАХНЮК _____
(підпис)

Суми – 2024

Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра кібербезпеки

«Затверджую»
Завідувач кафедри
_____ Володимир ЛЮБЧАК
(підпис)
«___»_____ 20___ р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня бакалавр

зі спеціальності 125 – Кібербезпека, освітньо-професійної програми «Кібербезпека»
здобувача групи КБ-01 Опихайленко Богдана Юрійовича

1. Тема роботи: «Порівняльний аналіз захищеності систем керування вмістом».

затверджено наказом по СумДУ №0212-VI від 04.03.2024 р. зі змінами згідно Наказу №0566-VI від 21.05.2024 р
2. Термін подання студентом роботи: «____» _____ 20 ___ р.

3. Вихідні дані до роботи: _____

4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити):
1) Аналіз проблеми предметної області, постановка та формування завдань дослідження.
2) Огляд досліджень, що були вже проведені. 3) Організація власного дослідження та порівняння його з іншими результатами. 4) Аналіз результатів.
5. Перелік графічного матеріалу (із зазначенням плакатів, презентацій тощо) _____
6. Консультанти до проекту (роботи), із зазначенням розділів, що їх стосуються

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання «____» _____ 20 ___ р.

Завдання прийняв до виконання _____ Керівник _____
(підпис) (підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Термін виконання	Примітка
1	Аналіз проблеми предметної області, постановка та формування завдань дослідження		
2	Визначення найслабкіших сторін		
3	Проведення власного дослідження		
4	Аналіз отриманих результатів		
5	Оформлення пояснювальної записки до кваліфікаційної роботи		

Здобувач вищої освіти _____ Керівник _____
(підпис) (підпис)

АНОТАЦІЯ

Кваліфікаційна робота виконана на 50 аркушах та містить 23 рисунки, 6 таблиць та 26 джерел.

Об’єкт дослідження: веб-ресурси, побудовані на різних CMS-платформах.

Мета роботи: провести дослідження, наскільки сучасні системи керування вмістом, та розгорнуті на них веб-сайти є захищеними.

Метод дослідження: метод аналітичного, практичного дослідження

Результати: результатом є проведена аналітична робота стосовно поставленого питання, проведено дослідження з наявними практичними кроками, які демонструють процес дослідження безпеки різних CMS.

Ключові слова: cms, дослідження безпеки, wordpress, joomla, wpscan, joomscan, owasp zap.

ЗМІСТ

ВСТУП

Дослідження захищеності веб-ресурсів, створених на базі CMS-систем, є ключовою темою в сучасній інформаційній безпеці. Оскільки системи управління контентом (CMS), такі як WordPress, Joomla, використовуються для створення та управління веб-сайтами різного типу, вони стають мішенями для кібератак та інших загроз безпеці.

Дослідження цієї теми включає в себе аналіз різних аспектів безпеки веб-ресурсів. Це охоплює виявлення потенційних вразливостей в коді CMS, плагінах або темах, оцінку ефективності захисних механізмів, таких як механізми аутентифікації та авторизації, а також розгляд можливостей здійснення атак, таких як крос-сайтовий скриптинг, SQL-ін'єкції та інші.

Актуальність. Тема кваліфікаційної роботи є актуальною, оскільки майже половина веб-ресурсів побудована за допомоги CMS систем. З кожним роком загрози в Інтернеті стають все більш складними та розповсюдженими. Веб-сайти, засновані на CMS, стають мішенями для кіберзлочинців, які шукають вразливості для злому, внесення змін або викрадення конфіденційної інформації.

Враховуючи чутливість інформації, яка зберігається на багатьох веб-ресурсах, таких як особисті дані користувачів, фінансові дані та інтелектуальна власність компаній, безпека стає пріоритетом для власників сайтів і їх користувачів.

Об’єкт дослідження: системи керування вмістом, та розгорнуті на них веб-сайти.

Предмет дослідження: Процес, методи аналізу та оцінки захищеності веб-ресурсів, що базуються на CMS-системах.

Структура. Дана робота складається зі вступу, аналітичного огляду, постановки задачі, опису проблем безпеки сучасних CMS, дослідження рівня безпеки в CMS, висновків та списку використаних джерел.

1 АНАЛІТИЧНИЙ ОГЛЯД

1.1 Аналіз поняття CMS

CMS (Content Management System) – це платформа для управління вмістом веб-сайту. У професійному середовищі CMS часто називають "двигуном сайту". За даними агентства W3Techs [1], більшість веб-сайтів у мережі працюють на CMS. Деякі з найпопулярніших CMS включають WordPress, Joomla! та OpenCart.

На практиці CMS – це веб-додаток, що дозволяє користувачам створювати та управляти веб-сайтами. Однією з основних переваг CMS є можливість створення веб-сайтів без необхідності знання програмування.

Всі системи управління контентом можна умовно розділити на чотири типи: системи з відкритим кодом, коробкові, саморобні та фреймворки. Коротко розглянемо кожну з них.

Системи з відкритим кодом

Відкритий код означає, що будь-хто може модифікувати двигун. Це призводить до регулярного появлення нових доповнень та тем у таких CMS, а також швидшого виявлення та виправлення вразливостей. Це одна з основних причин, чому WordPress став настільки популярним. Популярні системи з відкритим кодом включають WordPress, OpenCart, Joomla!, Drupal, Magento, PrestaShop.

Коробкові CMS

У сутності ці двигуни відрізняються лише тим, що у них зашифрований код, тому внесення змін можливе лише для офіційних розробників. Це не означає, що такі CMS менш безпечні або мають гірший функціонал, проте кількість доступних тем та доповнень зазвичай дійсно обмежена. Популярні коробкові CMS включають Tilda, Wix, SitePro, Shopify, Squarespace.

Саморобні CMS

Ці двигуни розробляються на замовлення для конкретних проектів. Їх функціонал не такий розгалужений, як у коробкових CMS або CMS з відкритим

кодом, проте він максимально відповідає поставленим завданням і не містить зайвих компонентів. Однак, якщо потрібно розширити функціонал або усунути уразливості, доведеться звертатися до розробника двигуна або шукати фахівця, який зможе аналізувати код з нуля, що коштує грошей і часу.

Фреймворки

Фреймворк – це надбудова над мовою програмування, набір бібліотек, за допомогою яких можна створити сайт для будь-яких завдань. Розробка на фреймворку вимагає більше зусиль та часу, і крім сайту потрібно окремо розробляти панель управління, що також є сайтом. Проте це дозволяє реалізувати будь-який функціонал, який вам потрібен. Такий спосіб ідеально підходить для нетипових проектів. Крім того, продуктивність сайту на фреймворку може бути вищою, якщо проект добре розроблений. Популярні фреймворки включають Laravel, Ruby on Rails, Django [2].

Переваги та недолі CMS

При огляді особливостей CMS важливо відзначити їх позитивні аспекти. Серед ключових переваг систем можна виокремити таке:

- Швидко та легко створити власний сайт.
- Не потрібно мати навичок у дизайні чи програмуванні.
- Розробка сайту вимагає меншого бюджету.
- Зручне керування вмістом порталу.

Звичайно, продукт має і деякі негативні риси. Серйозною перешкодою для початківців може стати відстеження актуальних версій та їх сумісність з розширеннями. Не завжди вдається реалізувати весь доступний функціонал, і не кожна CMS впорається з нетиповими завданнями [3].

1.2 Порівняння популярних CMS

Згідно даних сайту w3techs найпопулярнішим CMS є Wordrpess, далі йде Shopify та Wix (рис.1.1).

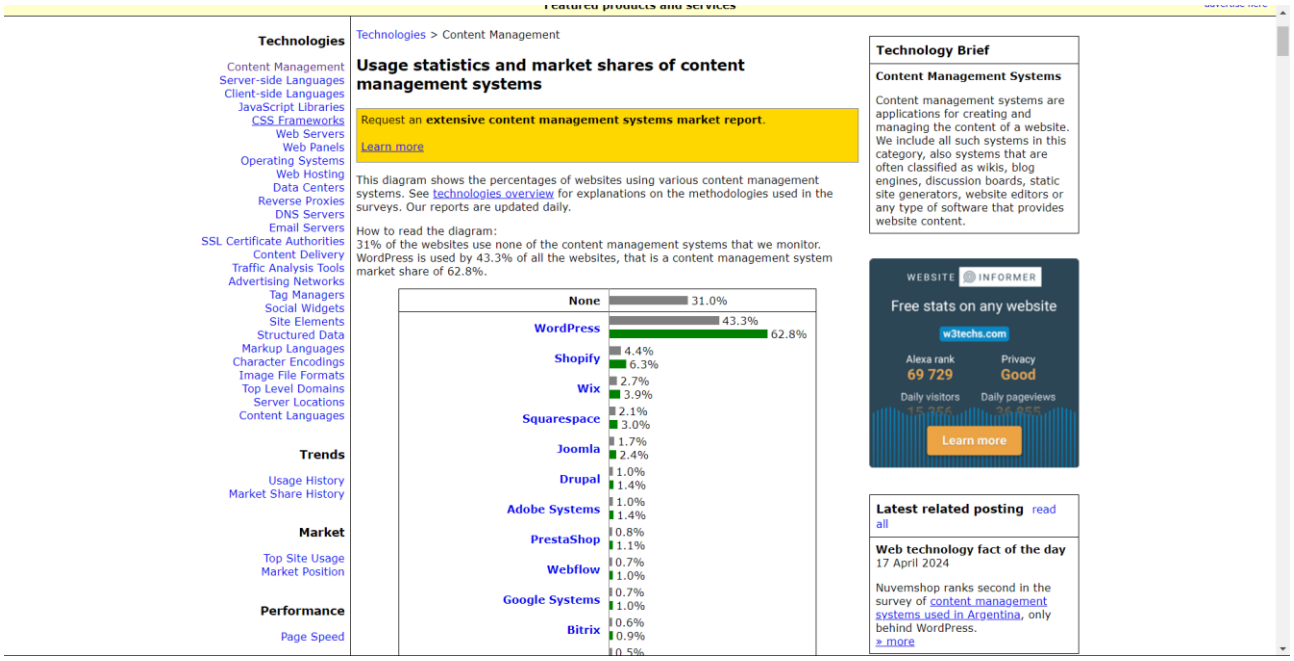


Рисунок 1.1 – Дані сайту w3techs

Далі найчастіше буде йти робота з WordPress та Joomla, оскільки це дві найпопулярніші CMS.

WordPress – як вже було сказано раніше, це одна з найпопулярніших CMS у світі. Ця система використовується для майже будь-яких видів сайтів, починаючи від звичайних інформаційних ресурсів і закінчуючи комерційними проектами [4].

Переваги:

– Популярність.

– Wordpress має одну з найбільших спільнот користувачів і розробників, що забезпечує підтримку, оновлення та велику кількість безкоштовних тем і плагінів.

– Зручна та інтуїтивна адміністраторська панель.

Недоліки:

– Оскільки Wordpress є однією з найпопулярніших CMS, вона може бути більш піддатливою до кібератак і потребує постійних оновлень та захисту.

– При порівнянні з іншими системами керування контентом, Wordpress

може бути менш гнучким у вирішенні деяких складних веб-задач або створенні корпоративних рішень високого рівня.

Joomla – є потужною CMS, яка підходить для середніх і великих веб-проектів. Вона має широкі можливості для організації та структурування вмісту, розширюється за допомогою розширень (компонентів, модулів, плагінів) і забезпечує високий рівень безпеки [5].

Переваги:

- Joomla має більшу гнучкість у порівнянні з Wordpress, що дозволяє створювати складніші веб-проекти та впроваджувати різноманітні функціональності.
- Joomla також має велику спільноту користувачів і розробників, що забезпечує підтримку, оновлення та багатий вибір розширень для розширення можливостей платформи.

Недоліки:

- Продуктивність: Joomla є менш продуктивною CMS якщо порівнювати її з іншими системами. Особливо це помітно на великих і складних веб-сайтах які використовують цю систему.
- Складність використання: Для новачків ця система може здаватися надто складною, з великою кількістю функцій які не потрібні для роботи.

Для більш чіткого порівняння, використаємо таблицю(таблиця 1.1).

Таблиця 1.1 – Порівняння CMS

Параметр	**Wordpress**	**Joomla**
1	**2**	**3**
Популярність	Дуже популярна CMS з великою спільнотою користувачів і розробників	Має значну кількість користувачів і розробників

Продовження табл.1.1

1	2	3
Адміністраторська панель	Зручна та інтуїтивна адміністраторська панель	Адміністраторська панель з великою кількістю функцій
Гнучкість	Менш гнучка у вирішенні складних веб-задач	Більша гнучкість для створення складніших веб-проектів
Безпека	Більш піддатлива до кібератак і потребує постійних оновлень та захисту	Високий рівень безпеки
Продуктивність	Добра продуктивність для більшості веб-сайтів	Менш продуктивна на великих і складних веб-сайтах
Складність використання	Легка для використання, підходить для початківців	Може викликати складнощі для новачків з великою кількістю функцій
Розширення	Велика кількість безкоштовних тем і плагінів	Багатий вибір розширень, але більшість платні

1.3 Безпека CMS

Всупереч поширеній думці, сайти CMS так само безпечні, як і сайти, розроблені за вимогами. Якщо використовувати CMS, яка добре відома і широко використовується в усьому світі, це є надійне рішення, яке регулярно розвивається. Це не обов'язково стосується користувацького веб-сайту, розробленого агентством або фрілансером, коли обслуговування не передбачено.

Проте CMS становлять ризик для більш серйозних атак. Дійсно, вони є метою для хакерів, які прагнуть використовувати вразливості. На п'ять основних CMS припадає 46,8% усіх веб-сайтів: тому зловмисники частіше стикаються з вразливими сайтами.

Типологія користувачів платформи CMS також ширша, ніж для сайтів, розроблених за запитом. Це означає, що є користувачі, менш знайомі з

передовими методами забезпечення безпеки, що збільшує ймовірність успішних атак на CMS.

З погляду безпеки основні CMS з відкритим вихідним кодом мають перевагу: дослідники безпеки мають до них доступ і тестують їх, що дає змогу виявляти проломи в безпеці. Зворотний бік медалі полягає в тому, що зловмисники також мають до них доступ, що також дає їм змогу знаходити вразливості та використовувати їх.

Велика спільнота навколо основної CMS розробила безліч плагінів, які додають функціональні можливості, але також додають ризики, пов'язані з інтеграцією, налаштуванням та обслуговуванням цих плагінів.

Безпека CMS і плагінів залежить від спільноти, яка для основної CMS, як правило, дуже активна. Проте, необхідно проявляти пильність щодо обслуговування CMS. Завжди існує ризик того, що рішення може бути менш популярним, менш підтримуваним або навіть занедбаним у якийсь момент.

Як і у випадку з будь-яким власним продуктом, безпека залежить від важливості, яку надає йому компанія, і від знань у сфері безпеки команди розробників.

Перевага власної CMS у тому, що за її розробку і безпеку відповідає безпосередньо команда. Тому цілком імовірно, що буде план запланованих оновлень, функцій, які необхідно протестувати, елементів, які необхідно виправити, тощо.

Проте слід пам'ятати про один момент пильності: можливо, що продукт може бути закинутий редактором у певний момент, і що більше не буде проводитися будь-яке технічне обслуговування.

Одним з основних ризиків, пов'язаних із CMS, є оновлення. Оновлення повинні проводитися регулярно, оскільки CMS швидко розвивається.

Крім того, регулярно виявляються і потім виправляються нові вразливості. Тому необхідно якомога швидше встановлювати оновлення і часто перевіряти доступні патчі.

Оновлення стосуються не тільки версій самої CMS, а й версій різних використовуваних плагінів. Для основної CMS з відкритим вихідним кодом доступна велика кількість плагінів.

Є два основні принципи, які слід враховувати під час вибору плагінів:

- Перевага належить плагінам, які постійно оновлюються (щоб не наражатися на ризик у разі виявлення нових вразливостей у непідтримуваному плагіні).
- Перевагу віддають визнаним і широко використовуваним плагінам, а не «внутрішнім» плагінам, розробленим для конкретних потреб, які не є особистими.

Інший серйозний ризик CMS, пов'язаний із конкретними розробками. Багато веб-сайтів на основі CMS залежать не тільки від конфігурації, а й від індивідуальних розробок, що виконуються або власною командою розробників, або постачальником послуг [6].

1.4 Аналіз атак на різні CMS

Фахівці Sucuri представили звіт [7], присвячений трендам у сфері злому сайтів у 2022 році. Аналітики дійшли висновку, що WordPress став найбільш зламуваною CMS року, на яку припадає 96% подібних атак. Слідом, з величезним відривом, йдуть Joomla (1,9%) і Magento (0,7%), дані наведено на рис.1.2.

Експерти пишуть, що більшість зломів, як і раніше, пов'язані з вразливостями не в самих CMS, а з неправильною конфігурацією, а також вразливостями в плагінах і темах, які адміністратори часто забувають оновити.

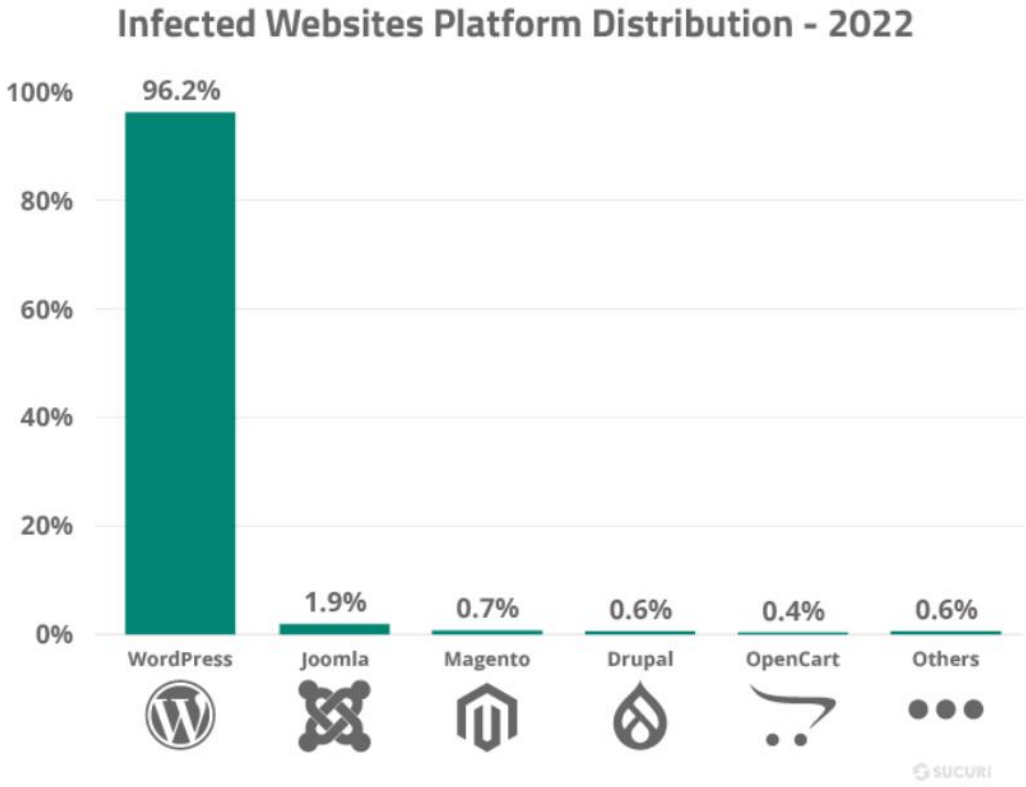


Рисунок 1.2 – Розповсюдження платформи заражених веб-сайтів

Так, лише 49% вивчених аналітиками сайтів працювали з актуальними версіями ПЗ (рис.1.3.).

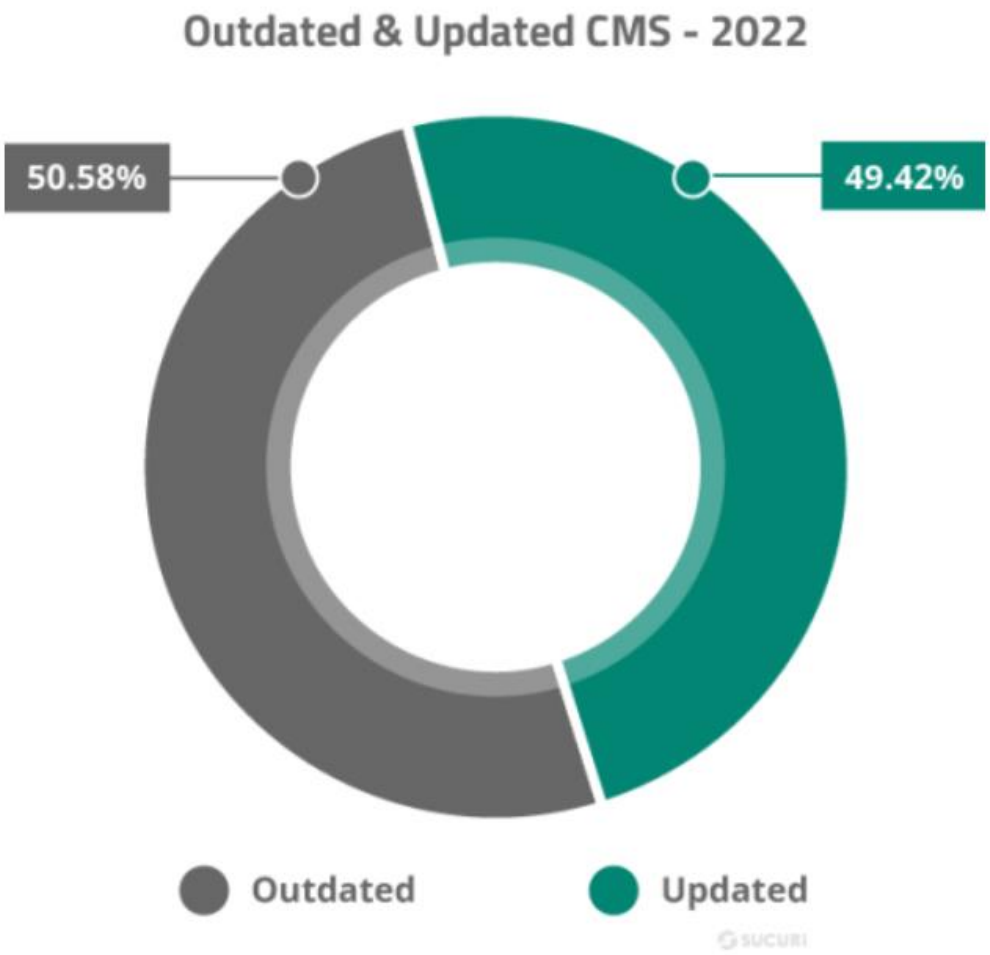


Рисунок 1.3 – Відсоток оновлених та неоновлених CMS

Як не дивно, будучи "найбільш зламуваним" WordPress не є CMS, яку найгірше оновлюють. Навпаки, фахівці Sucuri пишуть, що лише 49% зламаних сайтів працювали із застарілими версіями WordPress, тоді як злом PrestaShop, vBulletion, osCommerce або OpenCart практично гарантує, що проблема крилася в застарілому софті.

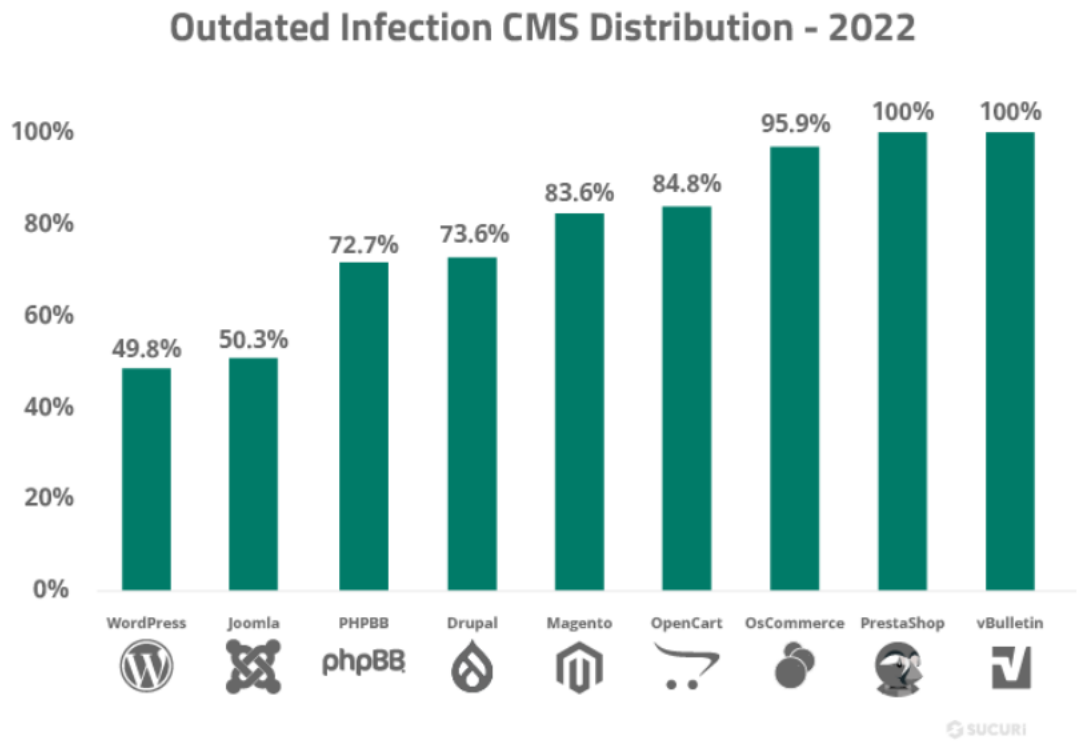


Рисунок 1.4 – Процентне співвідношення неоновлених та застарілих CMS

Звіт компанії Sucuri ще гарний тим, що він виде не просто статистику, а показує файли/частину коду де найчастіше було використано вразливість та як саме вона працює.

1.5 Постановка задачі

Метою роботи є дослідити, наскільки сучасні системи керування вмістом, та розгорнуті на них веб-сайти є захищеними.

Для досягнення поставленої мети необхідно вирішити наступні задачі:

1) Провести аналіз джерел стосовно поставленого питання;
2) Визначити найслабкіші сторони CMS базуючись на знайдених дослідженнях;
3) Провести власне дослідження визначення рівня захищеності різних CMS систем;
4) Провести аналіз отриманих результатів.

2 ПРОБЛЕМА БЕЗПЕКИ СУЧАСНИХ CMS

Сучасна веб-розробка потребує нових підходів, щоб відповідати постійно змінюваним вимогам. Веб-сайти мають швидко й ефективно оновлювати свою інформацію. Для задоволення цієї потреби були створені системи управління контентом (CMS), які дозволяють користувачам легко змінювати вміст сайту за допомогою зручних інтерфейсів. CMS стали одним із найпопулярніших інструментів для управління веб-порталами, і багато студій веб-дизайну тепер пропонують послуги з розробки на основі різних CMS-платформ.

Проте існуючі CMS-рішення не гарантують повної безпеки веб-сервера навіть для компаній середнього розміру. Компанії часто стикаються з проблемами безпеки своїх корпоративних веб-сайтів через вразливості в CMS. Для обслуговування CMS також потрібен системний адміністратор, що збільшує операційні витрати й все одно не гарантує повної безпеки. Значну роль у цих проблемах відіграє людський фактор. Для розробки безпечної системи управління вкрай важливо оцінити сильні та слабкі сторони найпоширеніших платформ CMS [8].

Основною перевагою CMS є простота редагування та додавання нового контенту на сайт. Системи управління контентом також дозволяють швидко й зручно оновлювати сторінки новин, гостьові книги, форуми, блоги тощо. Багато CMS мають простий у використанні візуальний редактор, що робить їх доступними для людей з обмеженими технічними навичками.

Проте загальним недоліком більшості CMS є динамічна адресація. Наприклад, адресація за замовчуванням у CMS Joomla дозволяє зловмиснику легко змінювати значення переданих змінних, компрометуючи систему безпеки. Динамічна адресація також створює проблеми для реєстрації сайту в пошукових системах. Для вирішення цієї проблеми майже всі CMS використовують функцію mod_rewrite, яка замінює адресу запиту на таку, що не містить імен змінних. На жаль, mod_rewrite не завжди підтримується хостинг-провайдерами й не повністю захищає CMS від атак модифікованих змінних. Відповідальність за обробку та

виявлення таких атак лягає на систему управління, що робить CMS вразливою до низки атак, включно із SQL-ін'єкціями.

Будь-яка система управління контентом має вразливості, які можуть призвести до злому сайту та сервера. Оновлення системи управління контентом є складним процесом, оскільки більшість систем не підтримують повністю автоматизоване оновлення, покладаючи цю відповідальність на адміністраторів. Цю проблему можна вирішити лише за допомогою активної системи оновлення, яка виконує оновлення автоматично. У більшості систем управління автоматичні оновлення частково виконуються на основі запиту адміністратора. Крім того, більшість CMS виконують неповний аналіз переданих параметрів. На цьому рівні дуже важливо захистити систему управління від SQL-ін'єкцій та атак на включення PHP [9].

2.1 Ризики CMS сайтів

WordPress, Joomla, Drupal, Shopify та інші платформи надають реальні можливості для створення ергономічних та ефективних веб-сайтів. Однак існує негативний імідж, пов'язаний із сайтами на CMS: їх вважають незахищеними і вразливими для хакерів.

Всупереч поширеній думці, сайти на CMS можуть бути настільки ж безпечними, як і індивідуально розроблені сайти. Використання добре відомої CMS, яка активно розвивається, забезпечує надійне рішення. Це не завжди стосується сайтів, розроблених на замовлення, де технічне обслуговування може бути обмеженим.

Сайти на CMS частіше стають мішенню для атак. Користувачі цих платформ часто менш обізнані про належні практики безпеки, що підвищує ризик успішних атак.

З точки зору безпеки, основні CMS з відкритим кодом мають перевагу завдяки відкритому вихідному коду, який дозволяє дослідникам безпеки

перевіряти його на наявність вразливостей. Але цей код також доступний зловмисникам, що може дозволити їм знайти та використати вразливості.

Велика спільнота навколо популярних CMS розробила багато плагінів, що додають функціональність, але також збільшують ризики, пов'язані з їх інтеграцією, конфігурацією та обслуговуванням. Безпека CMS та плагінів залежить від активності спільноти, яка зазвичай є дуже високою. Проте потрібно бути пильними щодо обслуговування CMS, оскільки завжди існує ризик, що певне рішення може стати менш популярним або навіть покинутим.

Як і у випадку з будь-яким пропрієтарним продуктом, безпека залежить від того, наскільки серйозно компанія ставиться до цього питання і наскільки добре команда розробників володіє знаннями з безпеки.

Перевага власної CMS полягає в тому, що команда безпосередньо відповідає за її розробку та безпеку, з чіткою дорожньою картою оновлень і виправлень. Однак існує ризик, що розробник може покинути продукт, і він більше не буде підтримуватися.

Незалежно від того, чи використовується CMS з відкритим вихідним кодом чи пропрієтарна, вибір залежить від потреб ситуації. Безпека сайту визначається тим, як він управляється, налаштовується і підтримується, а не тим, на якій платформі він побудований.

Одним з основних ризиків, пов'язаних з CMS, є оновлення. Вони повинні здійснюватися регулярно, оскільки CMS швидко розвиваються і нові вразливості з'являються часто. Оновлення стосуються не лише самої CMS, але й плагінів.

Інший важливий ризик пов'язаний із спеціальними розробками, які можуть створювати додаткові вразливості. Ризики для веб-сайтів варіюються залежно від їх розміру та функціональності, включаючи крадіжку даних, перебої в обслуговуванні або нелегальний хостинг контенту.

Найкращим рішенням є виявлення та виправлення недоліків безпеки через проведення веб-пентесту, який спочатку може зосередитися на основних ризиках.

2.2 Найпоширеніші атаки на CMS

Веб-сайти, побудовані на системах управління контентом (CMS), часто стають мішенню для кібератак. Зловмисники застосовують різні методи для порушення безпеки таких сайтів.

Brute Force Attack (на прикладі WordPress)

Brute Force Attack використовує метод спроб і помилок, при якому багаторазово вводяться різні імена користувачів, паролі або їх комбінації, поки не буде знайдено правильну. За замовчуванням сайти на WordPress не обмежують кількість спроб входу, що дозволяє зловмисникам використовувати ботів для автоматизованих атак. Якщо пароль слабкий, brute force інструмент зрештою знайде правильні облікові дані та отримає доступ до сайту. Сторінки адміністратора WordPress легко знайти, оскільки всі сайти на WordPress мають стандартну сторінку входу, наприклад: test.com/wp-admin або test.com/wp-login.php. Навіть якщо атака не вдасться, велика кількість спроб входу може перевантажити сервер і сповільнити роботу сайту. Під час атаки деякі хостинги можуть призупинити роботу акаунта через перевантаження системи, особливо якщо використовувати віртуальний хостинг [10].

SQL Injection

SQL-ін'єкція виникає, коли значення, що додаються до полів введення, не захищені належним чином, що дозволяє виконувати небажані SQL-запити. Після успішної SQL-ін'єкції зловмисник може отримати доступ або створити новий привілейований обліковий запис, який використовується для входу та отримання повного доступу до сайту. SQL-ін'єкції також можуть бути використані для вставки нових даних у базу даних, зміни або видалення існуючих [11].

Cross-Site Scripting (XSS)

Міжсайтовий скриптинг (XSS) – це ін'єкційна атака на стороні клієнта, при якій зловмисник впроваджує шкідливі скрипти у веб-сторінку. Ці скрипти завантажуються на стороні клієнта і можуть збирати дані або перенаправляти на інші шкідливі сайти. Ця вразливість використовується для крадіжки сесійних

файлів cookie користувачів, що дозволяє зловмиснику видавати себе за жертву. XSS ділиться на три категорії: збережені XSS, відображені XSS і XSS на основі DOM. Це одна з найпоширеніших вразливостей на веб-сайтах, яку легко знайти та виправити. Для запобігання потрібно використовувати належну перевірку даних на сайті [12].

DDoS-атака

Під час атак на відмову в обслуговуванні (DoS) до веб-сервера надходить велика кількість запитів, що уповільнює його роботу і врешті-решт призводить до його падіння. Сервер перевантажується і стає недоступним для користувачів. Розподілена відмова в обслуговуванні (DDoS) – це розширена версія DoS. DDoS здійснюється з декількох машин (ботнетів), що дозволяє приховати походження трафіку і збільшити обсяг запитів [13].

File Inclusion Exploits

Уразливості включення файлів зустрічаються на погано запрограмованих сайтах і виникають, коли сайт дозволяє користувачеві вводити дані у файли або завантажувати файли на сервер без належної перевірки. Є два типи: Локальне підключення файлів (LFI) і Віддалене підключення файлів (RFI). LFI дозволяє зловмиснику читати або виконувати файли на сервері, що може дати доступ до конфіденційної інформації. RFI дозволяє виконувати код, розміщений на машині зловмисника [14].

Directory Traversal

Directory Traversal – це HTTP-атака, яка дозволяє зловмиснику отримати доступ до обмежених файлів, каталогів і команд за межами кореневого каталогу веб-сервера. Вона відома як атака ../ (dot dot slash). Зловмисник може отримати доступ до інших частин файлової системи, на які веб-сервер має дозвіл на читання, що може дозволити переглянути файли з обмеженим доступом і компрометувати систему [15].

2.3 Методи аналізу захищеності CMS систем

При аналізі захищеності систем управління контентом (CMS), таких як WordPress, Joomla або Drupal, існує потреба в ретельному скануванні, тестуванні та моніторингу їх безпеки. Це важливо для запобігання можливим атакам та забезпечення захищеності веб-сайту. Отже, важливо мати на увазі різні інструменти, які допоможуть в цьому процесі.

Такі інструменти включають в себе програми сканування вразливостей, які автоматично перевіряють веб-сайти на наявність вразливостей у коді або конфігурації. Крім того, існують інструменти для аналізу коду та налаштувань CMS, а також для моніторингу журналів активності, що дозволяє вчасно виявляти підозрілу або неправомірну діяльність.

Коротко розглянемо ці інструменти.

Nessus

Це потужний сканер вразливостей, який може виявляти слабкі місця в безпеці не лише в самій CMS, але й в її компонентах та розширеннях [16].

Acunetix

Інструмент для автоматизованого сканування веб-додатків, включаючи CMS, для виявлення вразливостей, таких як кросс-сайтовий скриптинг (XSS), SQL-ін'єкції та інші [17].

OpenVAS

Це відкрите програмне забезпечення для сканування вразливостей, яке забезпечує розширені можливості для аналізу безпеки, включаючи сканування веб-додатків та виявлення вразливостей у CMS [18].

WPScan

Це спеціалізований інструмент для сканування вразливостей в WordPress, однак він також може виявляти вразливості в інших CMS, якщо вони мають подібні слабкі місця [19].

Burp Suite

Це інтегроване середовище тестування безпеки веб-додатків, яке може використовуватися для виявлення вразливостей в CMS шляхом ручного аналізу трафіку та автоматизованих атак [20].

ZAP (Zed Attack Proxy)

Це інший безкоштовний інструмент для тестування безпеки веб-додатків, який надає можливості сканування вразливостей і аналізу трафіку [21].

JoomScan

JoomScan – це ще один корисний інструмент для аналізу безпеки систем управління контентом, зокрема, для виявлення вразливостей у веб-сайтах, побудованих на платформі Joomla.

За допомогою наведених інструментів можна ефективно аналізувати та підтримувати високий рівень захищеності систем управління контентом (CMS), таких як WordPress, Joomla та інші [22].

3 ДОСЛІДЖЕННЯ РІВНЯ БЕЗПЕКИ В CMS

Перед початком дослідження, необхідно мати орієнтовану на злом ОС (операційну систему), наприклад, Kali Linux. Після її встановлення, починаємо роботу з терміналом. Запускаємо *sudo apt update*, ця команда гарантує, що всі пакунки, які завантажуються, мають найновіші версії (рис.3.1).

Рисунок 3.1 – Оновлення пакетів в Kali Linux

Після оновлення даних, необхідно встановити необхідні інструменти, всього їх буде два:

- wpscan
- joomscan

Для їх встановлення необхідно ввести команди *sudo apt install wpscan* та *sudo apt install joomscan.*

Також слід зазначити, що адреса всіх використаних сайтів, для тестування безпеки, не буде розголошено, на скріншотах всі відповідні строки буде закрито.

Дані інструменти працюють схожим чином, в їх основі лежить готова база вразливостей, які перевіряються під час сканування сайту.

3.1 Дослідження рівня безпеки в CMS WordPress

WordPress використовується для створення різноманітних веб-сайтів: від блогів до інтернет-магазинів, корпоративних сторінок та персональних портфоліо.

Першим кроком для визначення, чи працює сайт на WordPress, є огляд сайту та пошук кількох характерних ознак. Ручне дослідження сайту може бути ефективним та точним методом, оскільки багато сайтів відкрито показують, що використовують WordPress. Наприклад, на багатьох сайтах можна побачити текстовий блок "powered by WordPress".

Власники сайтів можуть видалити цей текст, але багато хто цього не робить. Відсутність цього блоку не обов’язково означає, що сайт не використовує WordPress. Інший спосіб перевірити це вручну – переглянути вихідний код сторінки. У більшості веб-браузерів це можна зробити, клацнувши правою кнопкою миші на будь-якому елементі сторінки і вибравши "Inspect" (рис.3.2).

Рисунок 3.2 – Кнопка Inspect

Витягнувши HTML-код сайту, можна швидко дізнатися про нього багато цікавого. Простий пошук "WordPress" часто може показати точну версію, на якій працює сайт (рис.3.3).

Якщо пошук не дає результатів, все одно можна шукати додаткові підказки, наприклад, такі елементи, як wp-block. Їх наявність також є вагомим свідченням того, що використовується саме Wordpress.

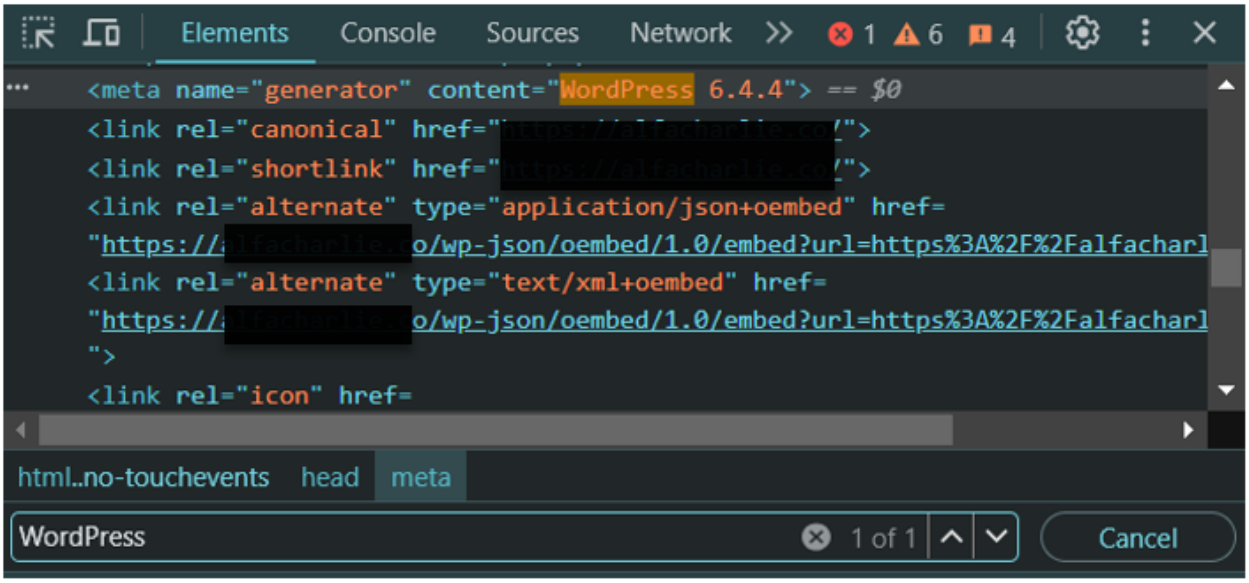


Рисунок 3.3 – Підтвердження, що сайт використовує WordPress

WordPress має специфічну структуру контенту, яка часто призводить до наявності сторінок з однаковими назвами. Деякі з найпоширеніших назв, які залишаються незмінними, включають:

– wp-admin.
– wp-content.
– wp-includes.
– readme.html.
– wp-login.php.
– wp-signup.php.

Однією з таких сторінок, яка зазвичай не підлягає змінам, є сторінка wp-admin. Це пов'язано з тим, що вона вбудована в код сайту і важко доступна для змін користувачам-початківцям. Через це її часто залишають без змін. Крім того, ця сторінка викликає певний інтерес, оскільки деякі версії WordPress можуть мати на ній деякі вразливості.

Також існують онлайн-інструменти, які дозволяють визначити, чи використовується на сайті WordPress. Наприклад, BuiltWith і Wappalyzer можуть автоматизувати процес виявлення. Обидва надають розширення для браузерів Chrome і Firefox (рис.3.4).

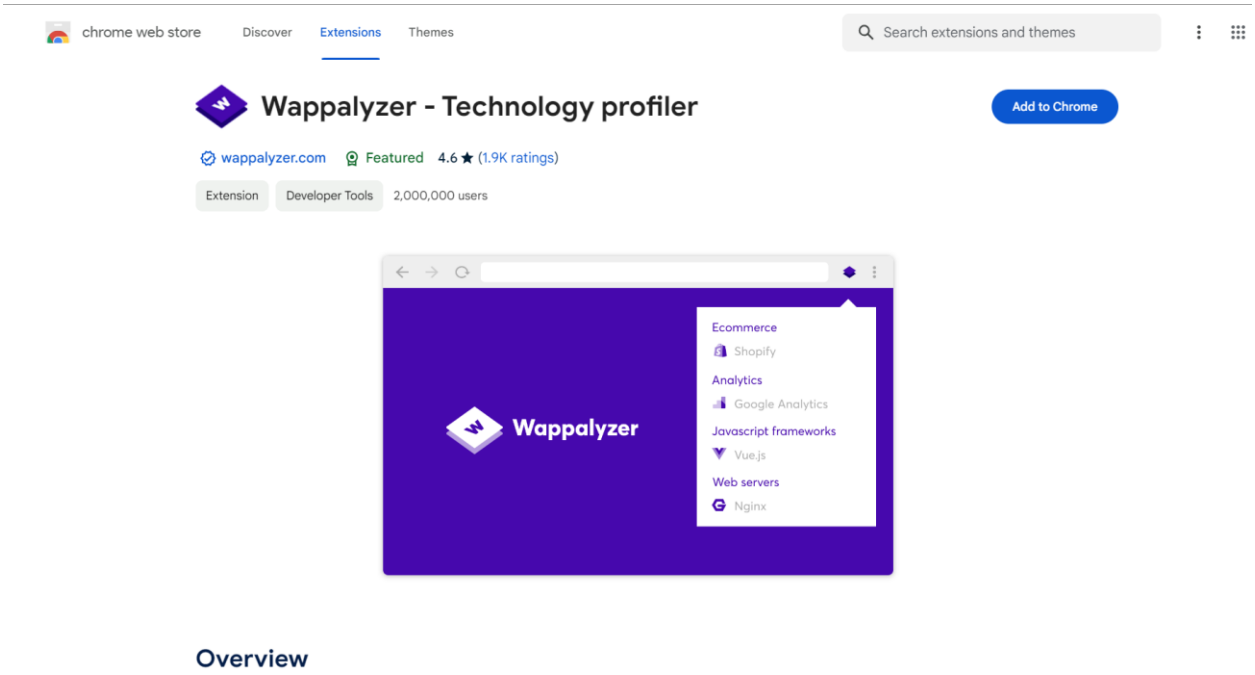


Рисунок 3.4 – Розширення Wappalyzer

Для проведення дослідження, буде використано інструмент WPScan. WPScan дуже популярний і може знаходити уразливості як в самому сайті WordPress, так і в його плагінах.

Перед початком, для налаштування отримання інформації про вразливість, необхідно отримати API Token на сайті WPScan (рис.3.5).

Процес реєстрації простий: просто переходимо на сайт і заповнюємо форму.

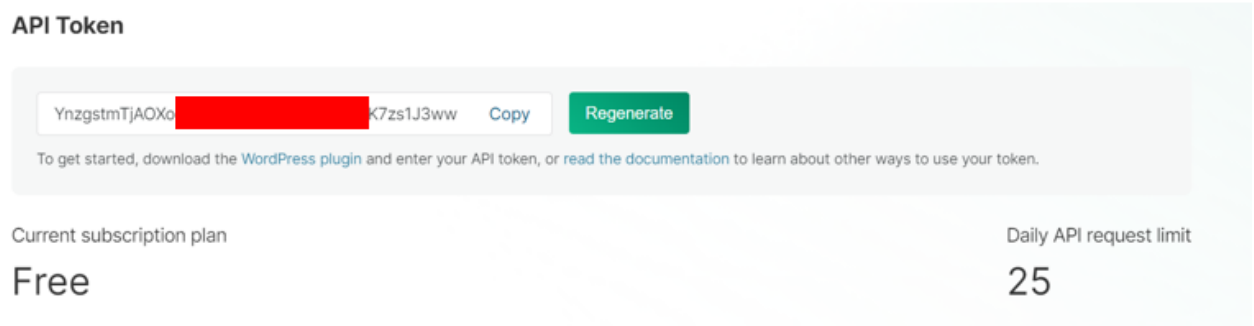


Рисунок 3.5 – Отриманий API ключ

Після отримання API-токену можна використати WPScan в повному обсязі. Це дозволить розпочати перерахування сайтів через командний рядок всередині Kali, використовуючи команду: wpscan --url <http://WEBSITE_URL/> --api-token <API_TOKEN>.

Застосуємо дану команду до обраної адреси (рис.3.6).

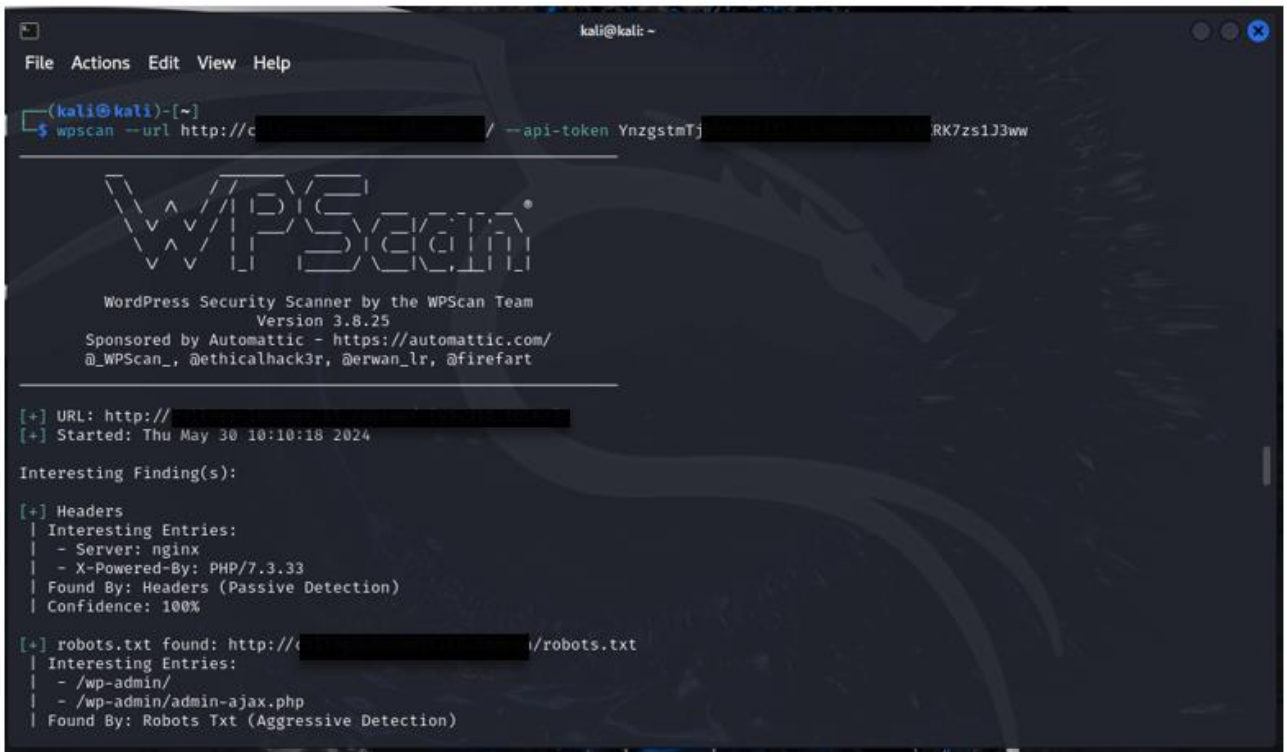


Рисунок 3.6 – Запуск скрипту

Запуск цієї команди поверне кілька результатів. Потрібно буде переглянути виявлені вразливості та всі плагіни, які він зміг виявити (рис.3.7).

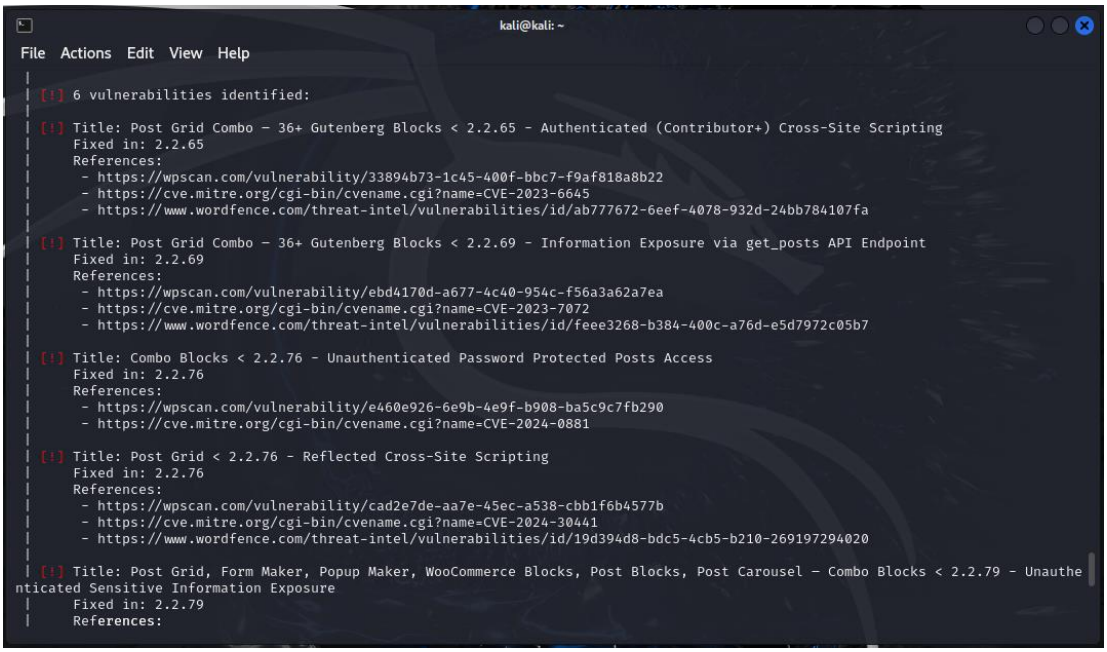


Рисунок 3.7 – Знайдені вразливості

У цьому випадку WPScan знайшов декілька вразливостей. Відомо, що наша цільова версія WordPress - 5.6.2, тому знайдені вразливості можуть спрацювати для отримання доступу до системи. Наразі візьмемо до уваги ці можливі уразливості і прокрутимо вниз до результатів по плагінам, щоб побачити, чи можна їх використати (рис.3.8).

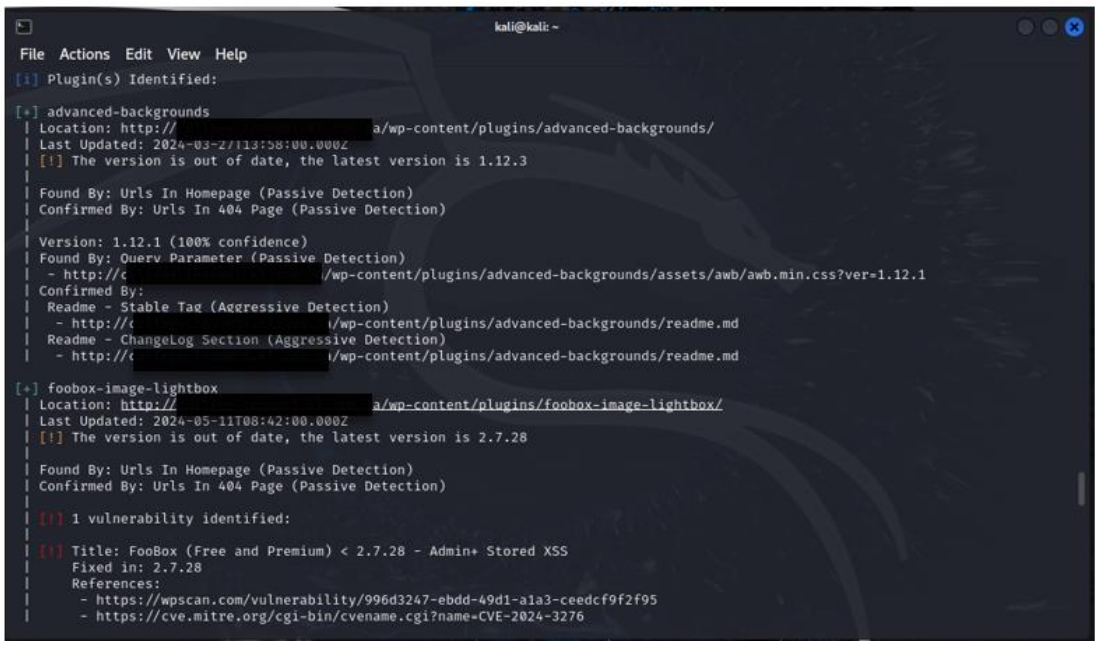


Рисунок 3.8 – Результати по плагінам

Схоже, що сайт використовує вразливу версію плагіна під назвою «advanced-backgrounds» та «foobox-image-lightbox». Оскільки в результатах пошуку є CVE, ми можемо загуглити його і подивитися, чи існують публічні експлойти (рис.3.9).

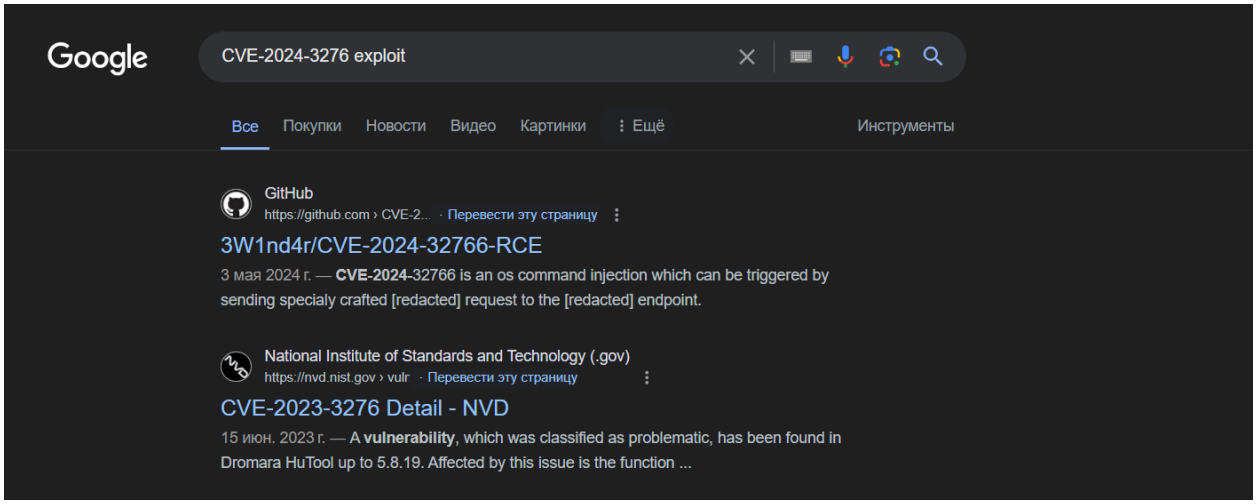


Рисунок 3.9 – Знайдені експлойти

Перед тим, як спробувати виконати експлойти, застосуємо метод перебору паролю. WPScan містить модуль вразливостей, за допомогою якого можна знайти будь-яку відому вразливість на сайті або в будь-якому плагіні WordPress. За замовчуванням, коли використовується WPScan з дійсним ключем API, він повертає результати публічно відомих вразливостей для сайту. Вони можуть варіюватися від експлойтів обходу логіну до вразливостей завантаження файлів та багато іншого. Іншим способом атаки на сайти WordPress за допомогою WPScan є використання атак грубої сили. Хоча це один з найпомітніших способів атаки на сайт WordPress, він може бути ефективним. WPScan містить модуль для підбору паролів спеціально для сайтів WordPress. Оскільки це специфічний інструмент для WordPress, не потрібно вказувати імена користувачів; він буде використовувати деякі імена користувачів WordPress за замовчуванням для підбору паролів (рис.3.10).

Рисунок 3.10 – Метод перебору паролю

Нажаль даний метод не надав ніякого результату, тому повертаємось до експлойту.

Щоб переконатися, що на сайті працює один з вражених плагінів, ми можемо перевірити його вихідний код і подивитися, чи отримаємо ми якесь підтвердження (рис.3.11).

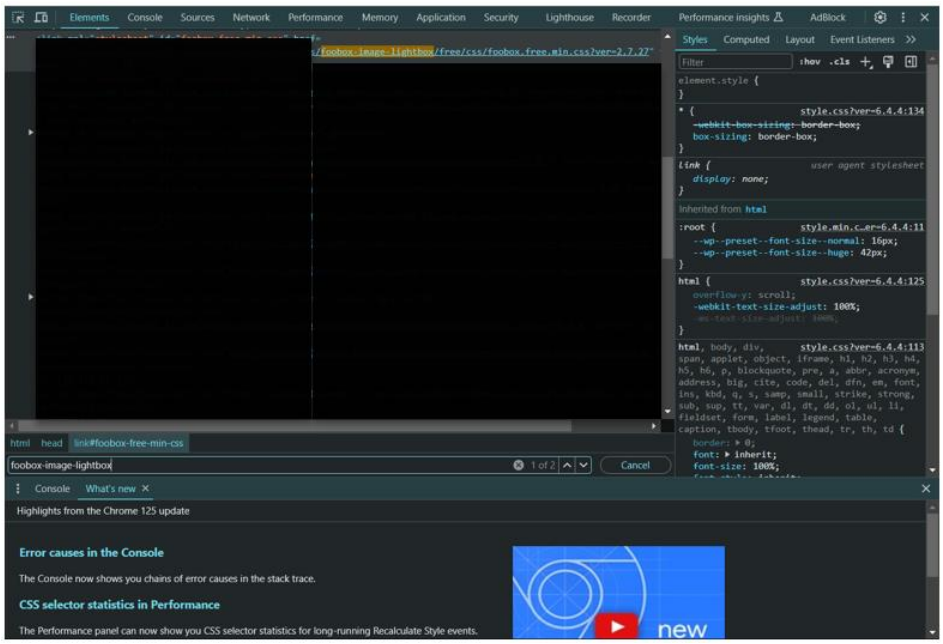


Рисунок 3.11 – Перевірка наявності плагінів

Протестувавши всі експлойти, нажаль, отримати корисної інформації не вдалось.

Таким же чином було досліджено ще 5 сайтів, результати тестування наведено в таблиці 3.1.

Таблиця 3.1 – Результати тестування безпеки сайтів WordPress

Номер сайту	**Можливість тестування через WPScan**	**Наявність WordPress вразливостей**	**Наявність вразливих плагінів**	**Результат Брутфорсу**	**Загальний опис**
1	**2**	**3**	**4**	**5**	**6**
Сайт 1	Так	Ні	Так	-	Вдалось знайти експлойт до вразливості, але він не робочий
Сайт 2	Ні (Https)	Ні	Ні	-	Не вдалось отримати ніяких даних
Сайт 3	Так	Ні	Ні	-	Не вдалось отримати ніяких даних
Сайт 4	Так	Ні	Так	-	Вдалось отримати хешований пароль, як результат виконання експлойту
Сайт 5	Ні (Https)	Ні	Так	-	Не вдалось отримати ніяких даних, окрім вразливих плагінів

3.2 Дослідження рівня безпеки в CMS Joomla

Joomscan – це інструмент, який можна використовувати для пошуку вразливостей, його також називають сканером вразливостей OWASP Joomla.

Для пошуку сайтів на тестування, було задіяно систему пошуків Гугл. Для цього було введено наступний запит: inurl:"http://" "Joomla".

Після вибору п’яти цікавих сайтів, наведемо кроки тестування одного з них (рис.3.12).

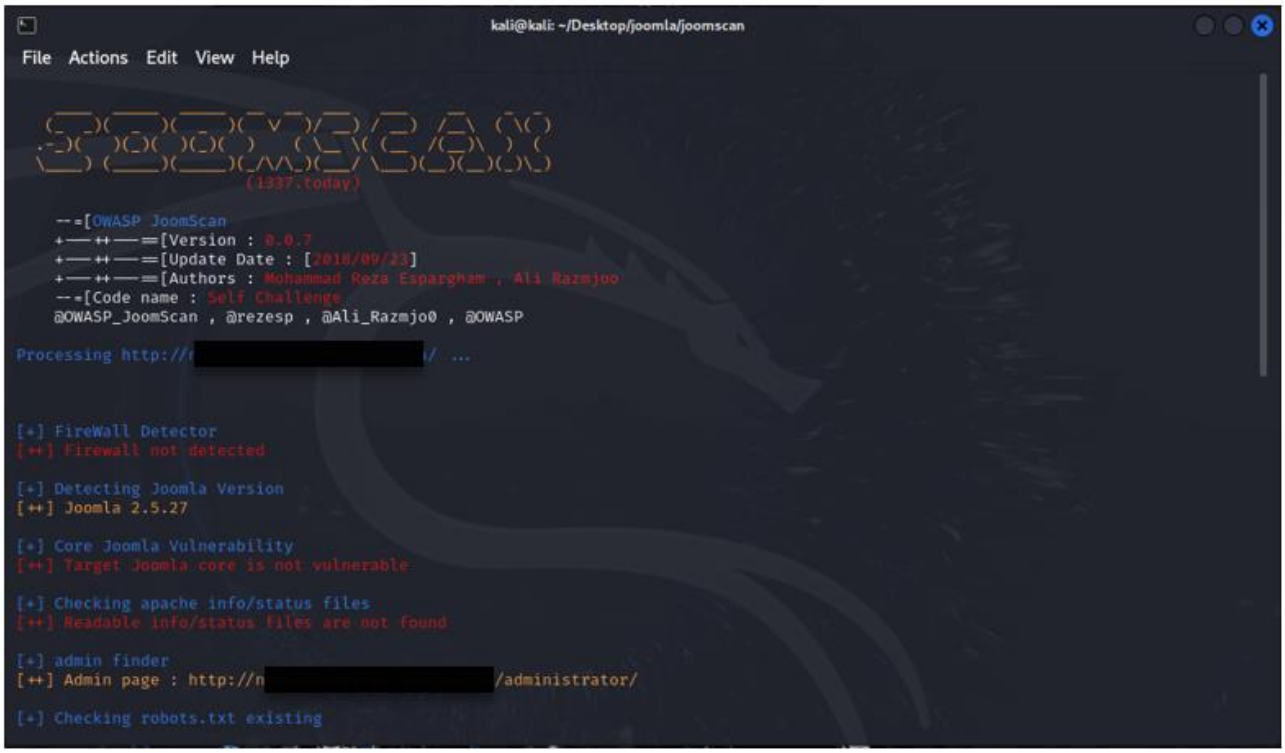


Рисунок 3.12 – Використання JoomScan для пошуку вразливостей (1)

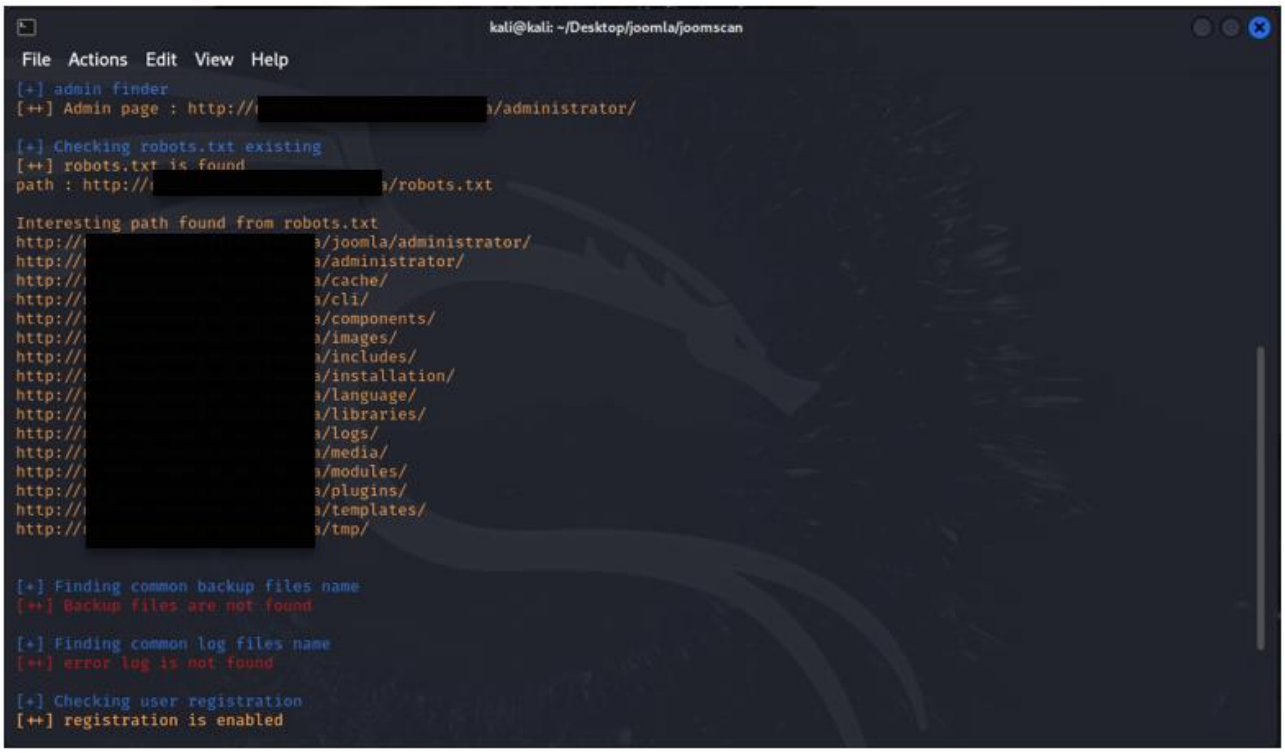


Рисунок 3.12 – Використання JoomScan для пошуку вразливостей (2)

Деяка інформація може бути використана зловмисниками. Під час сканування виявлено, що списки каталогів активні, що може дозволити зловмисникам знайти файли, які користувач хотів залишити прихованими. Знання URL-адреси адміністратора означає, що зловмисник може використовувати ПЗ Hydra або аналогічний інструмент для проведення словникової атаки на облікові дані для входу в систему.

Хоча під час тестування на скріншотах не виявлено жодних вразливостей, проте той факт, що сторінку адміністратора легко знайти, а також увімкнення списків каталогів може викликати занепокоєння.

JoomScan також може сканувати компоненти, щоб визначити, яке додаткове програмне забезпечення Joomla встановлено власником сайту. Якщо будь-який з цих компонентів має відомі вразливості в безпеці, вони можуть стати додатковими шляхами для атаки (рис.3.13).

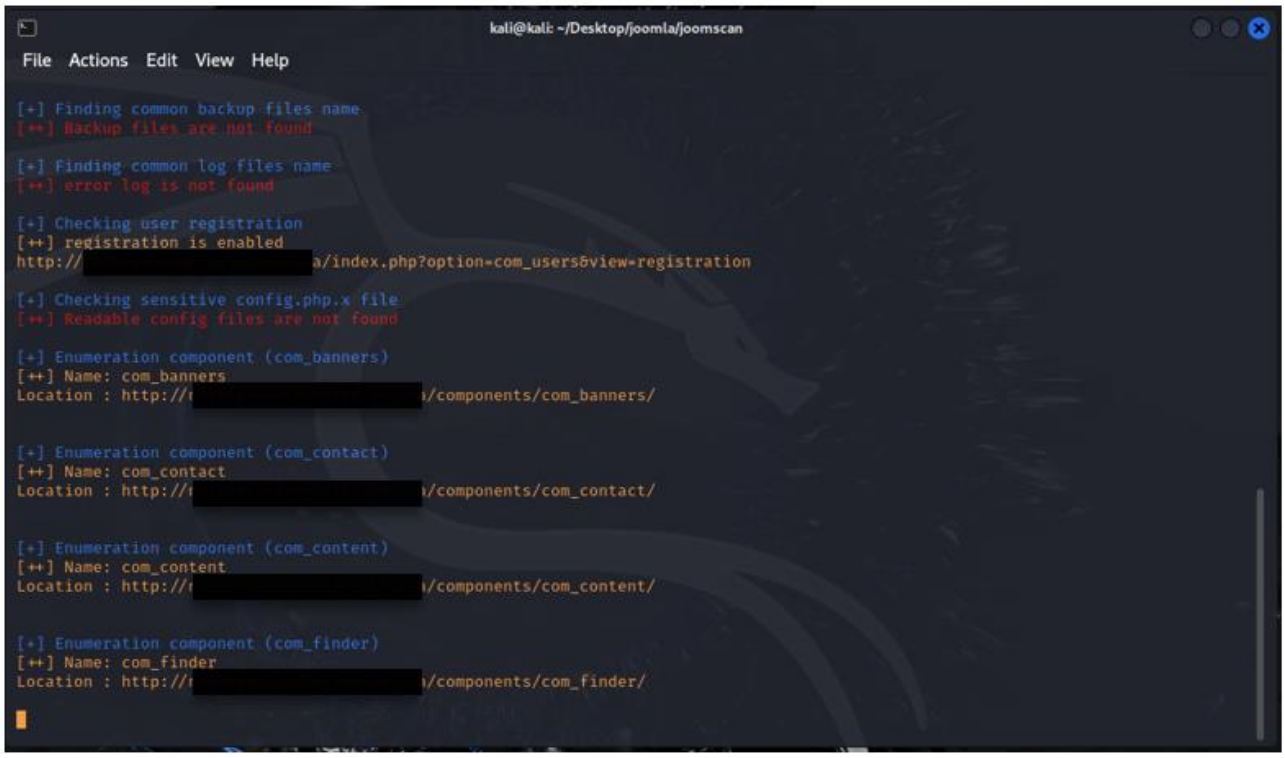


Рисунок 3.13 – Використання JoomScan для сканування компонентів

Результати тестування інших сайтів наведено в таблиці 3.2.

Таблиця 3.2 – Результати тестування безпеки сайтів Joomla

Номер сайту	**Можливість тестування через JoomScan**	**Наявність Joomla вразливостей**	**Наявність вразливих плагінів**	**Загальний опис**
1	**2**	**3**	**4**	**6**
Сайт 1	Так	Ні	Ні	Було знайдено тільки сторінку адміністратора
Сайт 2	Так	Ні	Ні	Не вдалось отримати ніяких даних
Сайт 3	Так	Ні	Ні	Не вдалось отримати ніяких даних
Сайт 4	Так	Ні	Ні	Не вдалось отримати ніяких даних
Сайт 5	Так	Ні	Ні	Було знайдено тільки сторінку адміністратора

Використання інших інструментів для тестування сайтів на базі Joomla немає ніякого сенсу, оскільки інструмент JoomScan є найефективнішим і якщо він не видав ніяких результатів, шанс того, що інші інструменти допоможуть, досить малий.

3.3 Використання спеціалізованих утиліт для пошуку вразливостей

Проаналізувавши сайти за допомоги WPScan та Joomla, переходимо до інструменту OWASP ZAP, який дозволить провести тестування на впровадження сторонніх скриптів.

Першим на черзі будуть сайти з WordPress, для прикладу буде використано той же сайт, що і демонструвався до цього. Додаємо необхідний сайт до утиліти, та запускаємо інструмент Active Scan (рис.3.14).

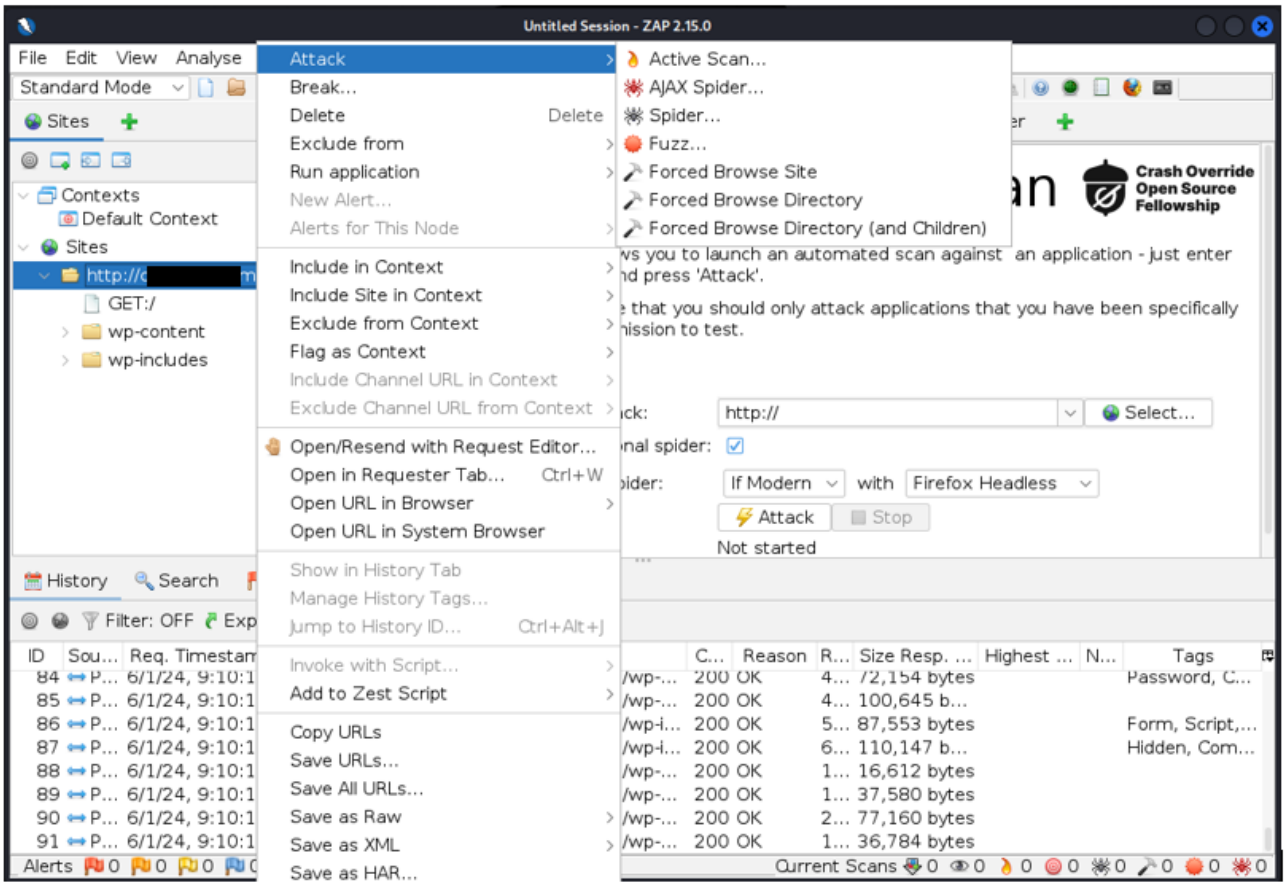


Рисунок 3.14 – Сканування сайту

По завершенню сканування, отримали наступний результат зображений на рисунку 3.15

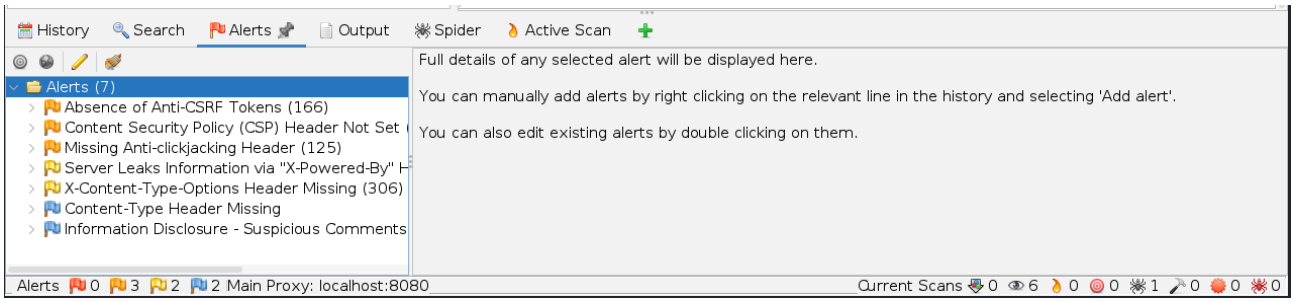


Рисунок 3.15 – Результат сканування Wordpress

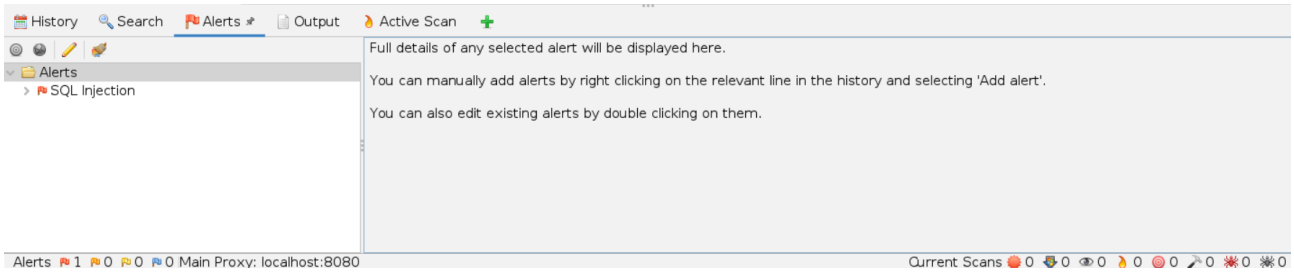


Рисунок 3.15 – Результат сканування Wordpress

Великим здивуванням виявилась відсутність на даному сайті вразливостей, які виявив інструмент WPScan (див.рис.3.7).

Загальний же результат не надав ніяких критичних зауважень. Аналіз всіх п’яти сайтів наведено в табл.3.3.

Таблиця 3.3 – Результати тестування сайтів WordPress через OWASP ZAP

Номер сайту	**Вразливості з високим пріоритетом**	**Вразливості з середнім пріоритетом**	**Вразливості з низьким пріоритетом**
1	**2**	**3**	**4**
Сайт 1	1	3	2
Сайт 2	1	3	4
Сайт 3	0	5	3
Сайт 4	1	6	4

Продовження табл.3.3

1	2	3	4
Сайт 5	0	2	3

Результати сканування інших чотирьох сайтів наведено в додатку А. Тепер застосуємо той же метод, але для сайту на Joomla (рис.3.16).

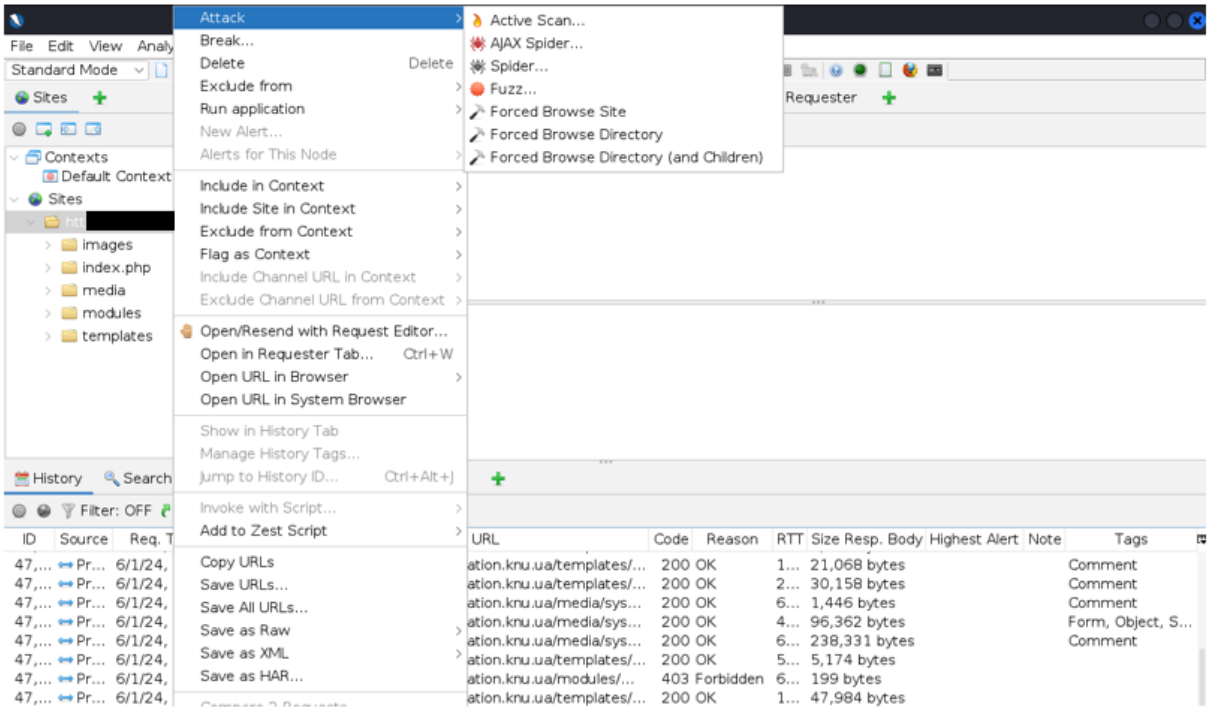


Рисунок 3.16 – Запуск сканування сайту на Joomla

Після завершення сканування, було виявлено вісім загроз, але жодної критичного рівня (рис.3.17).

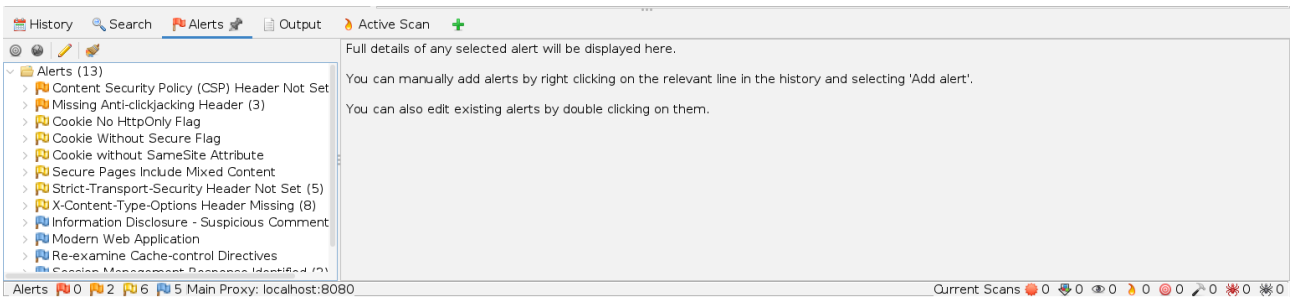


Рисунок 3.17 – Результат сканування

Таким же чином було проаналізовано ще чотири сайти на Joomla (табл.3.4). Скріншоти сканування наведено в Додатку Б.

Таблиця 3.4 – Результати тестування сайтів Joomla через OWASP ZAP

Номер сайту	**Вразливості з високим пріоритетом**	**Вразливості з середнім пріоритетом**	**Вразливості з низьким пріоритетом**
1	**2**	**3**	**4**
Сайт 1	0	2	6
Сайт 2	0	4	5
Сайт 3	0	5	6
Сайт 4	0	5	6
Сайт 5	1	0	0

3.4 Аналіз результатів

В загальній кількості було досліджено десять сайтів. П’ять розташовані на базі WordPress, п’ять на базі Joomla.

Найбільшу ефективність пошуку вразливостей продемонстрував інструмент WPScan, який використовується для тестування сайтів WordPress.

Тестування же сайтів Joomla продемонструвало гарний результат з іншого боку, сайти які підпали під перевірку, не видали жодної вразливості.

Найбільше всього було знайдено вразливостей в плагінах WordPress, саму ж платформу можна вважати захищеною, оскільки не було виявлено жодної вразливості.

Такі результати гарно описують матеріал поданий в першому розділі, де було проаналізовано дані фахівців Sucuri. За їхньою статистикою, найбільш вразливі місця в WordPress, це саме плагіни, проведене дослідження це доказує.

Також проведене дослідження не ставить під сумнів дані щодо злому сайтів Joomla, відсоток яких всього 1.9%.

Застосування інструменту OWASP ZAP, також надало багато результатів. Якщо взяти всі десять сайтів з їх вразливостями, то можна зробити висновок, що вразливостей в сайтах WordPress значно більше (табл.3.5).

Таблиця 3.5 – Порівняння результатів сканування

Тип CMS	**Вразливості з високим пріоритетом**	**Вразливості з середнім пріоритетом**	**Вразливості з низьким пріоритетом**
1	**2**	**3**	**4**
WordPress	3	19	16
Joomla	1	16	23

З таблиці видно, що вразливостей які несуть серйозну загрозу для сайту, в WordPress більше. Інші вразливості мають майже однакову кількість.

Дане дослідження ще раз підтверджує статистику надану фахівцями з Sucuri.

3.5 Рекомендації підвищення безпеки

Одним з найкращих способів підвищення та перевірки рівня захисту, є аудит, тому розглянемо його етапи більш детально на прикладі WordPress.

Аудит безпеки WordPress – це аналіз заходів безпеки та вразливостей, що використовуються на веб-сайті WordPress. Під час аудиту безпеки WordPress розробники оцінюють конфігурацію, налаштування, теми, плагіни, ролі користувачів та інші елементи сайту, щоб виявити потенційні ризики безпеки або слабкі місця, якими можуть скористатися зловмисники.

По суті, це комплексна оцінка поточного стану безпеки веб-сайту. Мета аудиту – виявити слабкі місця і підвищити загальний рівень безпеки сайту, захистити його від несанкціонованого доступу, витоку даних, зараження шкідливим програмним забезпеченням та інших потенційних загроз.

Такі види аудиту дозволяють усунути вразливості в системі безпеки до того, як вони призведуть до втрати даних або будь-якої іншої атаки. У цьому сенсі вони є інвестицією, яка захищає бізнес від подальших, більших витрат.

Що стосується того, як часто необхідно проводити аудит безпеки сайту на WordPress, це залежить від того, наскільки він активний. Адміністратори менш активних сайтів можуть проводити аудит безпеки кожні один-три місяці. Однак більш активні сайти з високим трафіком і великою кількістю користувачів можуть виграти від більш частих аудитів.

Виконання аудиту безпеки WordPress, як правило складається з 9 елементів

1. Сканування вразливостей

Кілька онлайн-сервісів надають послуги сканування безпеки для веб-сайтів на WordPress. Ці сервіси сканують файли сайту, щоб виявити потенційні вразливості безпеки, зараження шкідливим програмним забезпеченням, проблеми цілісності файлів, ризиковані конфігурації веб-сайту, а також перевірити, чи є домен у чорному списку пошукової системи та підозрілі дії. Після завершення сканування ці інструменти надають звіт, в якому висвітлюються всі потенційні проблеми з безпекою.

Деякі з найпопулярніших сканерів включають Sucuri [23], Wordfence [24] і iThemes Security [25]. Однак майте на увазі, що ці сканери можуть перевіряти лише публічні сторінки та публікації, тому будь-які приватні файли будуть недоступні.

2. Перевірка плану хостинг-провайдера

Хоча вивчення власного сайту на WordPress має важливе значення для підтримки високих стандартів безпеки, також важливо враховувати, чи забезпечує веб-хостинг високий рівень безпеки. Деякі з функцій, на які слід звернути увагу при виборі хостингу, включають наступне:

- Резервне копіювання WordPress (бажано регулярне і автоматичне) і відновлення.
- Виявлення шкідливого програмного забезпечення.

- Запобігання розподіленій відмові в обслуговуванні (DDoS).
- Підтримка мережі доставки контенту (CDN).
- Моніторинг мережі.
- Незалежно від того, чи передбачає ваш план керований, віртуальний або виділений хостинг.
- Заходи безпеки електронної комерції.

3. Перевірка плагіну безпеки

Плагіни безпеки WordPress – це програмні пакети безпеки, які захищають сайт від відомих вразливостей і загроз. Вони дозволяють захистити конфіденційну інформацію, уникнути блокування сайту та зупинити кібератаки. Їхні найпоширеніші функції включають наступне:

Моніторинг і сканування веб-сайту на наявність шкідливого програмного забезпечення.

- Фільтрація спаму.
- Перевірка SSL-сертифікатів.
- Захист вашого сайту від атак нульового дня.
- Ремонт і відновлення зламаних сайтів.
- Зміцнення вашого сайту.
- Безпека процесу аутентифікації.
- Застосування брандмауерів.

Незалежно від того, безкоштовні вони чи платні, плагіни безпеки є обов'язковими для захисту сайту в постійно мінливому ландшафті безпеки, де WordPress, зважаючи на його популярність, залишається CMS, яка найчастіше піддається атакам.

Серед популярних плагінів безпеки – Sucuri, Jetpack, Wordfence, iThemes Security, All In One WP Security та BulletProof Security.

4. Перевірка наявності оновлень теми, плагінів та ядра

Загрози безпеці, вразливості та рішення постійно змінюються. Оновлення тем, плагінів, ядра та PHP містять виправлення вразливостей безпеки, а також покращення продуктивності, сумісності та додаткову функціональність.

Необхідно регулярно перевіряти всі активні теми і плагіни, щоб визначити, які з них потребують оновлення, і те ж саме стосується основних файлів WordPress.

5. Перевірка надійності методу резервного копіювання вашого сайту

Резервне копіювання сайту на WordPress важливе для запобігання втрати даних, швидкого відновлення сайту в разі катастрофічного збою, перенесення сайту на новий сервер або навіть CMS, а також для захисту цілісності сайту при оновленні тем, плагінів, основних файлів або версій PHP.

Три основні способи резервного копіювання сайту – це створення резервної копії вручну, автоматично за допомогою плагіна та автоматично за допомогою веб-хостингу. Автоматичне резервне копіювання є найбільш рекомендованим, а ручне – найменш рекомендованим, оскільки воно більш схильне до людських помилок і займає більше часу. Проте, в деяких випадках ручне резервне копіювання може бути гарною альтернативою.

6. Посилення безпеки входу

Сторінка входу – це сторінка, на якій необхідно ввести свої облікові дані для доступу до бекенду сайту. Хоча наявність стандартної URL-адреси для входу за замовчуванням добре підходить для початківців, які тільки вчаться адмініструвати сайт на WordPress, вона також зручна для хакерів, оскільки їм не потрібно витрачати час на визначення вашої URL-адреси для входу, перш ніж намагатися зламати ваші облікові дані. Тому одним з найкращих заходів безпеки для входу в систему є зміна URL-адреси на унікальну, нестандартну адресу, до якої хакери не матимуть доступу.

Інші заходи щодо посилення безпеки входу включають наступне:

- Впровадження двофакторної автентифікації для спроб входу.
- Використання довгих, надійних та унікальних паролів для всіх

облікових записів з правами адміністратора на вашому сайті.

- Обмеження невдалих спроб входу за допомогою плагінів, таких як Limit Login Attempts Reloaded або Wordfence Security.
- Використання CAPTCHA на сторінці входу.
- Блокування входів від адміністраторів з використанням відомих скомпрометованих паролів.

7. Видалення невикористовуваних плагіни, тем та файлів

Плагіни є найпоширенішим джерелом вразливостей, які хакери використовують для отримання несанкціонованого доступу до сайтів. Теми є набагато менш поширеним джерелом, але хакери все одно можуть використовувати їх. Нарешті, основні файли містять найменше джерел вразливостей. Це означає, що WordPress має високий рівень безпеки, але стороннє програмне забезпечення може бути не таким.

8. Моніторинг активності та вихід неактивних користувачів

Моніторинг активності користувачів означає створення автоматизованого журналу їхньої діяльності (які сторінки вони відкривали, які файли завантажували тощо). Це може здатися нав'язливим, але це необхідно для того, щоб визначити, які користувачі є легітимними, а які – ботами або скомпрометованими акаунтами.

Деякі з найпопулярніших плагінів, які реєструють активність користувачів, включають Sucuri, WP Security Audit Log, Simple History та Activity Log.

На додаток до реєстрації активності користувачів, необхідно виходити з системи користувачів, які простоювали або були неактивними протягом певного періоду. Це хороша практика безпеки, оскільки вона мінімізує ймовірність того, що хтось інший захопить їхній обліковий запис і спричинить неприємності. Наприклад, CSRF-атаки можуть відбуватися тільки в контексті сеансу користувача.

9. Перевірка на використання сайтом HTTPS

HTTP (HyperText Transfer Protocol – протокол передачі гіпертексту) – це традиційний метод, який використовують браузери для обміну даними з веб-сайтами. Однак він застарів і більше не є безпечним, тому замість нього слід використовувати HTTPS (HyperText Transfer Protocol Secure – захищений протокол передачі гіпертексту). HTTPS необхідний для захисту даних користувачів, покращення ранжування в пошукових системах та дотримання стандартів безпеки.

Щоб розгорнути HTTPS на сайті, потрібен сертифікат SSL/TLS. Отримати ці сертифікати можна від довірених центрів сертифікації (ЦС) або через безкоштовних постачальників сертифікатів, таких як Let's Encrypt, які деякі хостинги використовують, щоб надати своїм клієнтам доступні сертифікати з мінімальними зусиллями. Після отримання сертифікату також потрібно оновити URL-адреси сайту, налаштувати перенаправлення та забезпечити безпечне завантаження всіх ресурсів веб-додатку.

Багато браузерів вимагають від веб-сайтів SSL-сертифікати, перш ніж дозволити користувачам отримати доступ до нього. Якщо сайт буде позначений як "небезпечний" через відсутність сертифіката, це негативно вплине на пошукову оптимізацію [26].

Регулярно проводячи аудит безпеки, можна підвищити загальний рівень безпеки сайту і захистити його від несанкціонованого доступу, витоку даних, зараження шкідливим програмним забезпеченням та інших потенційних загроз. Також вирішуються проблеми до того, як вони стануть проблемами, що впливають на доступність і зручність використання сайту, заощадивши час і ресурси, які необхідно витратити на вирішення такої кризи.

ВИСНОВКИ

Дана робота демонструє актуальне питання, щодо захищеності сучасних систем керування вмістом, на яких побудовані веб сайти.

На основі отриманих даних, було проведено власне дослідження за рахунок тестування сайтів двох CMS систем, WordPress та Joomla.

Найбільшу ефективність пошуку вразливостей продемонстрував інструмент WPScan, який використовується для тестування сайтів WordPress.

Тестування сайтів Joomla через JoomScan продемонструвало гарний результат з іншого боку, сайти які підпали під перевірку, не видали жодної вразливості.

Тестування вразливостей через OWASP ZAP продемонструвало гарний результат сканування, з якого можна зробити наступний висновок: сайти побудовані на CMS все ще є достатньо вразливими та несуть загрозу даних для користувачів, найбільш вразливими є сайти побудовані на WordPress, які використовують стороні плагіни.

Також слід враховувати, що при тестуванні, важливо використовувати декілька ресурсів, це було доведено при використанні WPScan та OWASP ZAP, але ці два інструменти видали різні результати.

У ході виконання кваліфікаційної роботи були виконані такі завдання:

1. Проведено аналіз джерел стосовно поставленого питання;
2. Визначено найслабкіші сторони CMS;
3. Проведено власне дослідження визначення рівня захищеності різних CMS систем;
4. Проведено аналіз отриманих результатів.

Надалі планується вдосконалити процес тестування та зробити його ще більш автоматизованим, для цього необхідно розробити власний інструмент сканер, який буде універсальним для всіх відомих CMS.

СПИСОК ЛІТЕРАТУРИ

1. Usage Statistics and Market Share of Content Management Systems, April 2024. *W3Techs - extensive and reliable web technology surveys*. URL: https://w3techs.com/technologies/overview/content_management.

2. CMS – що це таке і як працює, види та приклади | HOSTiQ Wiki. *HOSTiQ Wiki*. URL: https://hostiq.ua/wiki/ukr/cms/.

3. CMS сайту: як працює, переваги, види – Новини Тернополя і області - За Збручем. *Новини Тернополя і області - За Збручем*. URL: https://zz.te.ua/cms-saytu-iak-pratsiuie-perevahy-vydy/.

4. Про нас. *Україна*. URL: https://uk.wordpress.org/about/.

5. Учасники проектів Вікімедіа. Joomla! – Вікіпедія. Вікіпедія. URL: https://uk.wikipedia.org/wiki/Joomla!

6. Security for Content Management Systems - CMS. -. URL: https://news.registro.gt/en/2022/02/21/security-for-content-management-systems/.

7. 2022 Website Threat Research Report. *Sucuri*. URL: https://sucuri.net/reports/2022-hacked-website-report/.

8. What are the current trends and challenges in CMS security and privacy?. *LinkedIn: Log In or Sign Up*. URL: https://www.linkedin.com/advice/0/what-current-trends-challenges-cms#:~:text=CMS%20security%20threats%20can%20come,force%20attacks,%20and%20data%20leakage.

9. Cybersecurity: What Risks if your Website is Based on a CMS?. *VAADATA - Ethical Hacking Services*. URL: https://www.vaadata.com/blog/cybersecurity-what-risks-for-websites-built-with-a-cms/.

10. What is a Brute Force Attack? | Definition, Types & How It Works. *Fortinet*. URL: https://www.fortinet.com/resources/cyberglossary/brute-force-

attack#:~:text=A%20brute%20force%20attack%20is,and%20organizations'%20systems%20and%20networks.

11. SQL (Structured query language) Injection. Imperva. URL: https://www.imperva.com/learn/application-security/sql-injection-sqli/#:~:text=SQL%20injection%2C%20also%20known%20as,lists%20or%20private%20customer%20details

12. What Is Cross Site Scripting (XSS) and How Does It Work? | Synopsys. *Synopsys | EDA Tools, Semiconductor IP and Application Security Solutions*. URL: https://www.synopsys.com/glossary/what-is-cross-site-scripting.html#:~:text=Definition,the%20user%20to%20click%20it.

13. Що таке DDoS-атаки та яку мету вони переслідують - Infocom UA. *Infocom UA*. URL: https://infocom.ua/що-таке-ddos-атаки/.

14. Apocalypse. File inclusion. *Medium*. URL: https://medium.com/@timsinabishal232/file-inclusion-faa044106f83#:~:text=A%20File%20Inclusion%20Attack%20is,to%20input%20or%20upload%20files.

15. What is a Directory Traversal Attack?. *Acunetix*. URL: https://www.acunetix.com/websitesecurity/directory-traversal/.

16. Nessus. *CMU School of Computer Science*. URL: https://www.cs.cmu.edu/~dwendlan/personal/nessus.html.

17. Acunetix Web Vulnerability Scanner Overview | Acunetix. *Acunetix*. URL: https://www.acunetix.com/support/docs/wvs/overview/#:~:text=Acunetix%20Overview,from%20a%20standard%20web%20browser.

18. OpenVAS | Bugcrowd. *Bugcrowd*. URL: https://www.bugcrowd.com/glossary/openvas-vulnerability-scanner/#:~:text=OpenVAS%20is%20a%20widely%20used,OpenVAS%20is%20open%20source.

19. Using WPScan to find WordPress vulnerabilities on your

website. *Melapress*. URL: https://melapress.com/wpscan-wordpress-security-scanner/.

20. Burp Suite - Application Security Testing Software. *Web Application Security, Testing, & Scanning - PortSwigger*. URL: https://portswigger.net/burp.

21. What is Zed Attack Proxy? - GeeksforGeeks. *GeeksforGeeks*. URL: https://www.geeksforgeeks.org/what-is-zed-attack-proxy/.

22. JoomScan - OWASP Joomla Vulnerability Scanner Project | CYBERPUNK. *CYBERPUNK*. URL: https://www.cyberpunk.rs/joomscan-owasp-joomla-vulnerability-scanner-project.

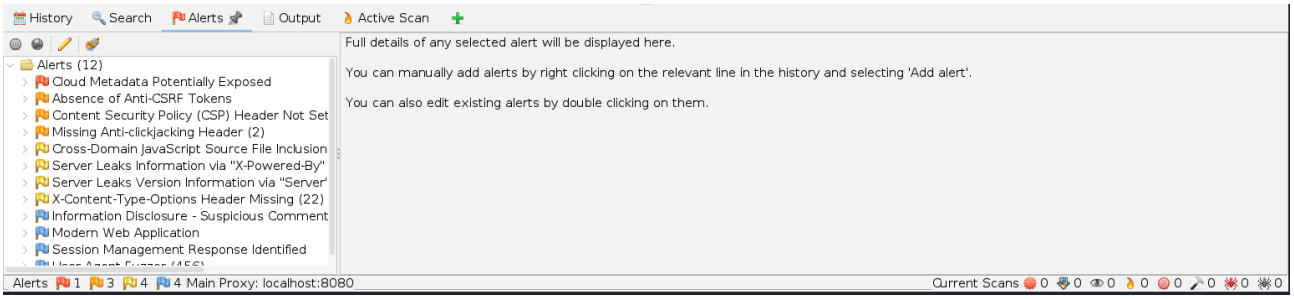
23. Sucuri - Complete Website Security, Protection & Monitoring. *Sucuri*. URL: https://sucuri.net/.

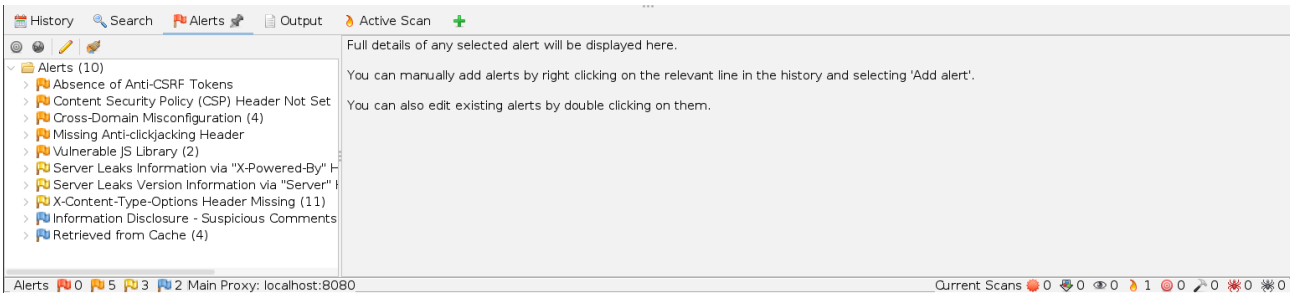
24. WordPress Security Plugin | Wordfence. *Wordfence*. URL: https://www.wordfence.com/.

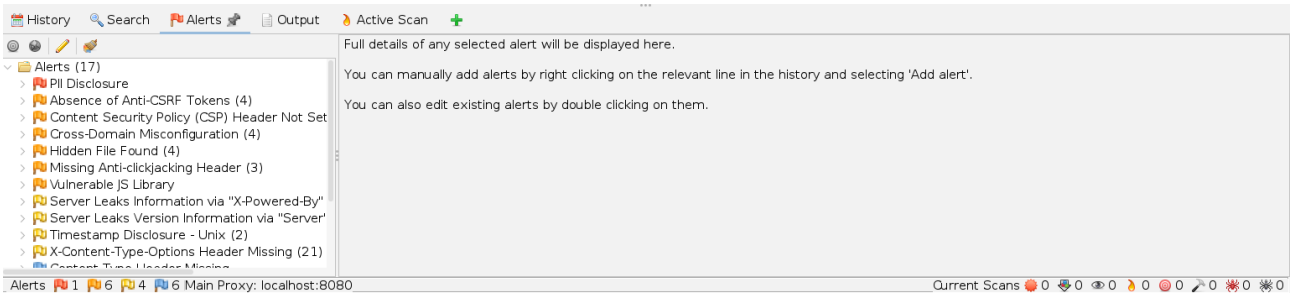
25. iThemes Security Review: Is It Worth Installing on Your WordPress Site? - MalCare. *MalCare*. URL: https://www.malcare.com/blog/ithemes-security-review/.

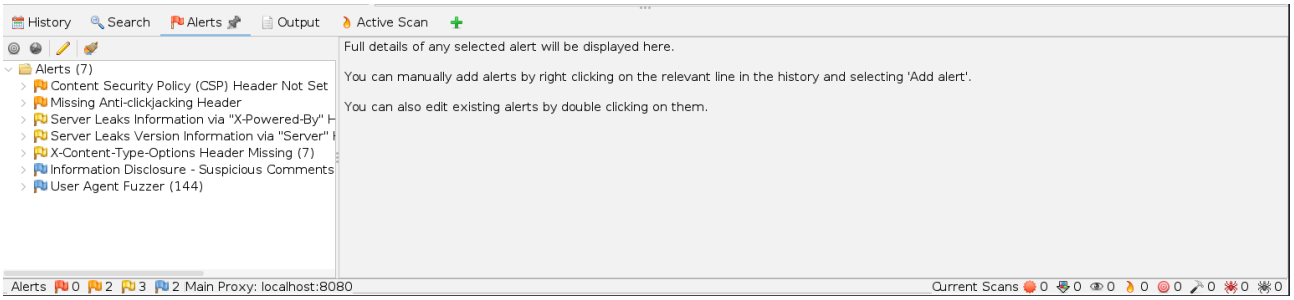
26. How to audit a WordPress website's security? - White Canvas. *White Canvas*. URL: https://wcanvas.com/blog/how-to-audit-a-wordpress-websites-security/.

ДОДАТОК А
РЕЗУЛЬТАТИ СКАНУВАННЯ WORDPRESS









ДОДАТОК Б
РЕЗУЛЬТАТИ СКАНУВАННЯ JOOMLA