**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**Національний університет кораблебудування імені адмірала Макарова**

Навчально-науковий інститут автоматики і електротехніки

Кафедра комп'ютерних технологій та інформаційної безпеки

ЗАТВЕРДЖУЮ
Завідувач кафедри комп'ютерних технологій та інформаційної безпеки, к.т.н., доцент
_____ С.М. Нужний
«____» _____ 2025 р.

# КВАЛІФІКАЦІЙНА РОБОТА

# АНАЛІЗ СИСТЕМИ ЗАХИСТУ ВЕБ-ЗАСТОСУНКІВ НА БАЗІ WORDPRESS

Ступінь вищої освіти: магістр
Галузь знань: 14 Електрична інженерія
Спеціальність: 141 «Електроенергетика, електротехніка та електромеханіка»
Освітньо-професійна програма: Системи технічного захисту інформації, автоматизація її обробки

Виконав: студент 6 курсу групи 6366мз
Лобан Ілля Олегович _____

Керівник:
к.т.н., доцент кафедри комп'ютерних технологій та інформаційної безпеки

Сірівчук Андрій Сергійович _____

Миколаїв – 2025

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**Національний університет кораблебудування імені адмірала Макарова**

Навчально-науковий інститут автоматики і електротехніки

ЗАТВЕРДЖУЮ
Гарант освітньої програми, д.т.н., професор,
професор кафедри КТІБ
_____ Передерій В.І.
«14» 10 2025 р.

ЗАВДАННЯ

на кваліфікаційну роботу

Студент: Лобан Ілля Олегович

Курс:6

Група: 6366мз

Ступінь вищої освіти: магістр

Галузь знань: 14 Електрична інженерія

Спеціальність: 141 «Електроенергетика, електротехніка та електромеханіка»

Освітньо-професійна програма: Системи технічного захисту інформації, автоматизація її обробки

1. Тема роботи: Аналіз системи захисту веб-застосунків на базі WordPress

затверджена наказом НУК від «14» 10 2025 р. № 1217 ук

2. Вихідні дані до роботи: вебресурс, створений на базі CMS WordPress, Процеси забезпечення інформаційної безпеки вебресурсу WordPress, зокрема: вразливості системи; моделі порушника; потенційні загрози; механізми атак та експлуатації слабких місць; засоби захисту й контрзаходи;

3. Перелік питань, які необхідно розробити: огляд WordPress та структура веб-сайту на WordPress; аналіз можливості забезпечення захисту відповідно до нормативних актів; провести аналіз основних вразливостей WordPress;створити модель порушника.

4. Терміни виконання завдання:
термін видачі завдання: «01- 05» вересня 2025 р.
термін подання роботи: «18» грудня 2025 р.

6. План-графік виконання кваліфікаційної роботи

| п/п | Заходи | Дата | | Готовність |
|---|---|---|---|---|
| | | Навч. тиждень | Контрольна дата | |
| 1. | Наукове стажування | 1 - 8 | 27.10.2025 | Звіт з наукового стажування |
| 2. | Організаційні збори | 9 | 29.10.2025 | Попередній варіант змісту КР |
| 3. | Рубіжний контроль | 10 | 05.11.2025 | Попередній варіант оглядових розділів, звіт з практика |
| 4. | Рубіжний контроль | 13 | 27-28.11.2025 | Доповідь на 13-ій Всеукраїнській науково-технічній конференції з міжнародною участю «СПІБТ–2025» |
| 5. | Рубіжний контроль | 14 | 3.12.2025 | 1. Попередній варіант повної КР<br>2. Попередній варіант презентації |
| 6. | КЕ на рівень «магістра» | 15 | 8,9.12.2025 | Складання державного екзамену зі спеціальності на ступінь магістра |
| 7. | Перевірка на плагіат | 15 | 10.12.2025 | Електронний варіант кваліфікаційної роботи, попередній захист (робота передається студентом на кафедру для внутрішньої рецензії). |
| 8. | Оформлення КР та супутньої документації | 16 | 15-17.12.2025 | 1. Оформлена КР (в форматі pdf) У раз отримання позитивної внутрішньої рецензії КР подається на зовнішню рецензію.<br>2. Оформлена презентація (в форматі pdf). |
| 9. | Подання документів на захист | 16 | 18.12.2025 | 1. Подання щодо захисту КР: тема, успішність, висновок керівника та висновок кафедри про КР.<br>2. Зовнішня рецензія (окремо від КР). Резюме на КР.<br>3. Електронний варіант презентації.<br>4. Електронний варіант КР на диску |
| 10. | Захист ДР | 17 | 22,23,24 12.2025 | 1. Приєднання студента до засідання кваліфікаційної комісії (конференції) у встановлений час для проведення процедури дистанційного захисту<br>2. Процедура захисту КР. |

Керівник
к.т.н., доцент кафедри комп'ютерних технологій
та інформаційної безпеки, _____ А.С. Сірівчук

Студент _____ І.О. Лобан

# АНОТАЦІЯ

Лобан І.О. Аналіз системи захисту веб-застосунків на базі *WordPress*. – Кваліфікаційна робота на правах рукопису.

Кваліфікаційна робота на здобуття ступеня вищої освіти «магістр» за спеціальністю 141 «Електроенергетика, електротехніка та електромеханіка» галузі знань 14 «Електрична інженерія» освітньо-професійної програми «Системи технічного захисту інформації, автоматизація її обробки». – Національний університет кораблебудування імені адмірала Макарова, Миколаїв, 2025.

Пояснювальна записка: 61 с., 35 джерел.

Кваліфікаційна робота присвячена актуальній темі аналізу розповсюдженої системи керування змістом WordPress. У кваліфікаційній роботі проведено комплексне дослідження загроз, вразливостей та механізмів захисту вебресурсів на базі WordPress з урахуванням вимог НД ТЗІ

У межах дослідження розроблено детальну модель порушника, класифіковано можливі вектори атак, сформовано розширені таблиці вразливостей та ризиків, виконано оцінювання початкових і залишкових ризиків із застосуванням кількісної шкали ймовірності та збитку.

Створено рекомендації щодо підвищення захисту *WordPress*, включаючи впровадження організаційних і технічних заходів, WAF, контролю доступів, моніторингу та аудитів безпеки.

Ключові слова: WordPress, вебзастосунки, вразливість, модель порушника, засоби захисту.

## *ANNOTATION*

*Loban I.O. Analysis of the system for protecting web applications based on WordPress. – Qualification work in the form of a manuscript.*

*Qualification work for the degree of higher education "Master" in the specialty* 141 *"Electrical power engineering, electrical engineering and electromechanics" field of knowledge* 14 *"Electrical engineering" of the educational and professional program "Systems of technical protection of information, automation of its processing". – Admiral Makarov National University of Shipbuilding, Mykolaiv,* 2025*.*

*Explanatory note:* 61 *p.,* 35 *sources.*

*The qualification work is devoted to the topical topic of analysis of the widespread content management system WordPress. The qualification work conducted a comprehensive study of threats, vulnerabilities and mechanisms for protecting web resources based on WordPress, taking into account the requirements of the ND TZI.*

*As part of the study, a detailed model of the attacker was developed, possible attack vectors were classified, extended tables of vulnerabilities and risks were formed, and initial and residual risks were assessed using a quantitative scale of probability and damage.*

*Recommendations were created to improve WordPress protection, including the implementation of organizational and technical measures, WAF, access control, monitoring and security audits.*

*Keywords: WordPress, web applications, vulnerability, attacker model, protection tools.*

# ЗМІСТ

# ПЕРЕЛІК СКОРОЧЕНЬ

*CMS – Content Management System*

*SEO –Search Engine Optimization*

*CMA – Content Management Application*

*CDA – content delivery application*

БД – база даних

КСЗІ – Комплексна система захисту інформації

*SEO – Search Engine Optimization*

*ACF – Advanced Custom Fields*

ПЗ – програмне забезпечення

*REST API – Representational State Transfer API*

*CVE – Common Vulnerabilities and Exposures*

# ВСТУП

Як фахівець з інформаційної безпеки, важливо розуміти мережі, операційні системи, бази даних, веб-додатки, мови сценаріїв та програмування тощо. Протягом своєї кар'єри ви, найімовірніше, стикатиметеся з різноманітними типами веб-додатків. Зі збільшенням досвіду ви почнете неодноразово бачити деякі з веб-додатків, як з точки зору зловмисника, так і захисника. *WordPress* – один з найпоширеніших веб-додатків, що використовуються організаціями всіх розмірів і типів.

Об’єктом кваліфікаційної роботи є надати глибоке розуміння того, як функціонують веб-сайти *WordPress*, щоб краще позиціонувати їх для атаки та захисту від них .

Предметом кваліфікаційної роботи є сукупність загроз, вразливостей та механізмів захисту системи керування змістом *WordPress*, а також методи забезпечення інформаційної безпеки.

Метою роботи є дослідження вразливостей системи *WordPress* та визначення засобів протидії даним вразливостям.

Для вирішення поставленої мети вирішено наступні завдання:

– огляд *WordPress* та структура веб-сайту на *WordPress*;

– аналіз можливості забезпечення захисту відповідно до нормативних актів;

– провести аналіз основних вразливостей *WordPress*;

– визначення можливостей тестування на проникнення;

– визначити засоби захисту та створити модель порушника.

Результати роботи частково опубліковані в наступному джерелі:

Лобан І.О., Перспективи викристання *WPScan* для аналізу вразливостей сайту кафедри. Сучасні проблеми інформаційної безпеки на транспорті:

матеріали XIII Всеукраїнської науково-технічної конференції з міжнародною участю. Миколаїв : НУК, 2025. Ч. 1. 32-34 сс.

Кваліфікаційна робота відповідає компетентностям:

– КФ3 – Здатність досліджувати, розробляти і супроводжувати методи та засоби інформаційної безпеки та/або кібербезпеки на об’єктах інформаційної діяльності та критичної інфраструктури;

– КФ5 – Здатність до дослідження, системного аналізу та забезпечення безперервності бізнес/операційних процесів з метою визначення вразливостей інформаційних систем та ресурсів, аналізу ризиків та визначення оцінки їх впливу у відповідності до встановленої стратегії і політики інформаційної безпеки та/або кібербезпеки організації;

– КФ6 – Здатність аналізувати, контролювати та забезпечувати систему управління доступом до інформаційних ресурсів згідно встановленої стратегії і політики інформаційної безпеки та/або кібербезпеки організації;

– КФ7 – Здатність досліджувати, розробляти та впроваджувати методи і заходи протидії кіберінцидентам, здійснювати процедури управління, контролю та розслідування, а також надавати рекомендації щодо попередження та аналізу кіберінцидентів в цілому.

Дана кваліфікаційна робота може бути використана для:

– підвищення рівня навчання безпеки інформації веб-застосунків;

– підвищення рівня захисту майбутнього сайту кафедри.

# 1 ПРИЗНАЧЕННЯ ТА ОБЛАТЬ ЗАСТОСУВАННЯ

## 1.1 Огляд системи *WordPress*

### 1.1.1 Загальний огляд *WordPress*.

*WordPress* – найпопулярніша система керування вмістом (*Content Management System* (*CMS*)) з відкритим кодом, яка використовується майже на третині всіх веб-сайтів у світі [1]. Її можна використовувати для різних цілей, таких як розміщення блогів, форумів, електронна комерція, управління проектами, управління документами та багато іншого. *WordPress* має високі можливості налаштування, а також оптимізований для пошукової оптимізації (*Search Engine Optimization* (*SEO*)), що робить його популярним серед компаній. Він має велику бібліотеку розширень, які називаються темами та плагінами, як безкоштовними, так і платними, які можна додати для покращення веб-сайту. Деякими прикладами плагінів є *WPForms*, надійна контактна форма, *MonsterInsights*, яка взаємодіє з *Google Analytics*, та *Constant*, популярний сервіс *email*-маркетингу. Однак його налаштування та розширюваність роблять його схильним до вразливостей через сторонні теми та плагіни. *WordPress* написаний на *PHP* і зазвичай працює на *Apache* з *MySQL* як серверною частиною. Багато хостингових компаній пропонують *WordPress* як опцію під час створення нового веб-сайту та навіть допомагають із завданнями серверної частини, такими як оновлення безпеки.

*CMS* – це потужний інструмент, який допомагає створювати веб-сайти без необхідності писати все з нуля (або навіть знати, як взагалі писати код). *CMS* виконує більшу частину «важкої роботи» з боку інфраструктури, зосереджуючись більше на дизайні та презентації веб-сайту, а не на структурі серверної частини. Більшість *CMS* пропонують багатофункціональний редактор, де користувачі можуть редагувати контент так, ніби вони працюють у текстовому редакторі, такому як *Microsoft Word*. Користувачі можуть

завантажувати медіафайли безпосередньо з інтерфейсу медіатеки, замість того, щоб взаємодіяти з веб-сервером або з порталу керування, або через *FTP* чи *SFTP*.

*CMS* складається з двох ключових компонентів:

– додаток для керування контентом (*CMA*) – інтерфейс, який використовується для додавання та керування контентом;

– додаток для доставки контенту (*CDA*) – це серверна частина, яка приймає вхідні дані, введені в *CMA*, та збирає код у робочий, візуально привабливий веб-сайт.

Ось короткий опис того, що робить *CMS*:

– створення контенту: *CMS* надає зручний інтерфейс, щоб ви могли легко додавати нові дописи в блозі, статті, сторінки та інший контент на свій вебсайт;

– редагування контенту: *CMS* дозволяє вам легко редагувати існуючий контент на вашому вебсайті;

– організація: підтримуйте порядок на своєму вебсайті, класифікуючи контент та використовуючи теги для зручного пошуку. Уявіть собі це як акуратне розміщення інформації на вашому вебсайті в шафах та шухлядах;

– публікація: за допомогою кількох кліків ви можете опублікувати свій контент і зробити його доступним для всього світу на своєму вебсайті;

– співпраця: деякі *CMS*-платформи дозволяють кільком користувачам одночасно працювати над контентом веб-сайту. Це особливо корисно для команд, які працюють разом над веб-проектом.

Особливості *WordPress* [2]:

– простота використання: *WordPress* може похвалитися інтуїтивно зрозумілим інтерфейсом. Додавання контенту, завантаження медіафайлів та налаштування вашого веб-сайту більше схоже на використання програми для редагування документів, ніж на кодування;

– керування контентом: по суті, *WordPress* – це потужна *CMS*. Ви можете без зусиль створювати, редагувати, упорядковувати та публікувати різні типи контенту, такі як дописи в блогах, сторінки, статті та навіть товари для інтернет-магазинів;

– гнучкість: *WordPress* не є універсальним рішенням. Тисячі попередньо розроблених тем підходять для широкого спектру стилів веб-сайтів, від простих блогів до складних бізнес-сайтів. Ці теми легко взаємозамінні, що дозволяє вам миттєво змінити зовнішній вигляд вашого веб-сайту;

– потужність плагінів: уявіть собі плагіни як функції, що розширюють можливості вашого веб-сайту. Величезна бібліотека пропонує такі функції, як контактні форми, галереї зображень, інструменти *SEO*-оптимізації, рішення для електронної комерції та багато іншого.

– оптимізація для пошукових систем (*SEO*): *WordPress* створено з урахуванням *SEO*, що полегшує пошуковим системам пошук і ранжування вашого веб-сайту. Це призводить до збільшення трафіку веб-сайту та видимості вашого контенту.

1.1.2 Структура *WordPress*

*WordPress* вимагає повністю встановленого та налаштованого стеку *LAMP* (операційна система *Linux, HTTP-сервер Apache*, база даних *MySQL* та мова програмування *PHP*) перед встановленням на хост *Linux*. Після встановлення всі допоміжні файли та каталоги *WordPress* будуть доступні в кореневому каталозі *webroot*, розташованому за адресою */var/www/html*.

Кореневий каталог *WordPress* містить файли, необхідні для коректної роботи *WordPress*. Ключовими файлами є:

– *index.php* – є головною сторінкою *WordPress*;

– *license.txt* – містить корисну інформацію, наприклад таку як встановлена версія *WordPress*;

– *wp-activate.php* – використовується для процесу активації електронної пошти під час налаштування нового сайту *WordPress*;

– *wp-admin* – тека містить сторінку входу для доступу адміністратора та панель керування серверною частиною. Після входу користувач може вносити зміни до сайту на основі призначених йому дозволів. Сторінка входу може знаходитися за одним із таких шляхів:

*/wp-admin/login.php*

*/wp-admin/wp-login.php*

*/login.php*

*/wp-login.php*

Цей файл також можна перейменувати, щоб ускладнити пошук сторінки входу:

– *xmlrpc.php* – це файл, що представляє функцію *WordPress*, яка дозволяє передавати дані за допомогою *HTTP* як механізму транспортування та *XML* як механізму кодування. Цей тип зв’язку було замінено на *WordPress REST API* ;

– *wp-config.php* – містить інформацію, необхідну *WordPress* для підключення до бази даних, таку як ім'я бази даних, хост бази даних, ім'я користувача та пароль, ключі автентифікації та солі, а також префікс таблиці бази даних. Цей файл конфігурації також можна використовувати для активації режиму DEBUG, що може бути корисним для усунення несправностей.

Ключові каталоги *WordPress*:

– *wp-content* – це головний каталог, де зберігаються плагіни та теми. Підкаталог *uploads*/ зазвичай є місцем, де зберігаються будь-які файли, завантажені на платформу. Ці каталоги та файли слід ретельно пронумерувати, оскільки вони можуть містити конфіденційні дані, які можуть призвести до віддаленого виконання коду або використання інших вразливостей чи неправильних конфігурацій;

– *wp-includes* – містить усе, крім адміністративних компонентів та тем, що належать веб-сайту. Це каталог, де зберігаються основні файли, такі як сертифікати, шрифти, файли *JavaScript* та віджети.

1.1.3 Ролі користувачів *WordPress*

Однією з ключових функцій, що робить *WordPress* зручним для користувача та безпечним, є його система керування користувачами [3]. Ця система використовує різні ролі користувачів для контролю того, що користувачі можуть і не можуть робити на веб-сайті, що є важливим для підтримки безпеки сайту та ефективного управління доступом.

Ролі користувачів у *WordPress* є критично важливими для ефективного та безпечного керування веб-сайтом. Вони дозволяють призначати дозволи та можливості різним користувачам на основі їхніх ролей та обов'язків. Призначаючи відповідні ролі, ви можете:

Підвищення безпеки: Обмежте доступ до конфіденційних областей вашого сайту, зменшуючи ризик несанкціонованих змін та порушень безпеки.

Типи ролей користувачів у *WordPress*.

*WordPress* має кілька попередньо визначених ролей користувачів, кожна з яких має унікальний набір дозволів. Ці ролі допомагають вам контролювати, що користувачі можуть робити на вашому сайті, гарантуючи, що кожна людина має відповідний рівень доступу. Ось основні ролі користувачів у *WordPress*.

1) Адміністратор. Роль адміністратора має найвищий рівень доступу та повний контроль над сайтом *WordPress*. Адміністратори можуть:

– керувати всім контентом (публікаціями, сторінками, коментарями).

– встановлення та оновлення тем і плагінів.

– додавати, редагувати або видаляти будь-якого користувача, включаючи інших адміністраторів.

– змінювати налаштування сайту та виконувати завдання з обслуговування сайту.

Ця роль зазвичай зарезервована для власників сайтів або головних менеджерів і повинна призначатися з обережністю через її широкий спектр дозволів.

2) Редактор. Редактори мають широкий контроль над контентом вашого сайту, але не можуть вносити зміни до всього сайту. Редактори можуть:

– додавати, редагувати, публікувати та видаляти всі публікації та сторінки, незалежно від того, хто їх створив;

– керувати категоріями, тегами та коментарями;

– завантажувати та керувати медіафайлами.

Ця роль ідеально підходить для контент-менеджерів, яким потрібно контролювати весь контент, але які не потребують адміністративного доступу.

3) Автор. Автори мають обмеженіші дозволи, зосереджені переважно на управлінні власним контентом. Автори можуть:

Писати, редагувати та публікувати власні дописи.

Завантажувати медіафайли до власних публікацій.

Видаляють власні публікації.

Автори не можуть редагувати або видаляти публікації, створені іншими користувачами, а також не можуть вносити зміни на весь сайт, що робить цю роль придатною для постійних учасників, яким не потрібен ширший доступ.

4) Учасник. Учасники можуть створювати контент, але мають обмежені права на публікацію. Учасники можуть:

– писати та редагувати власні публікації;

– надсилати публікації на перевірку редактору або адміністратору.

Однак вони не можуть публікувати власні дописи або завантажувати медіафайли. Після публікації допису учасники не можуть його далі редагувати. Ця роль найкраще підходить для запрошених авторів або менш досвідчених творців контенту.

5) Передплатники. Абоненти мають найбільш обмежений доступ. Вони можуть:

– керувати власним профілем;

– переглядати та коментувати публікації.

Ця роль зазвичай використовується для користувачів, яким потрібно увійти в систему, щоб отримати доступ до певних функцій сайту або коментувати контент, але не потрібно вносити свій внесок або керувати ним.

## 1.2 Основні можливості *WordPress*

У своїй основі *WordPress* пропонує повноцінний функціонал для публікації та управління контентом, де ключовими елементами є редакційні інструменти, система ролей, підтримка мультимедіа й можливість створення складних структур сторінок. Редактор *Gutenberg*, який базується на блочному

підході, дозволяє будувати контент із візуальних компонентів – текстових блоків, зображень, відео, галерей, колонок, таблиць та інших елементів. Цей підхід дає змогу не просто редагувати текст, а створювати повністю дизайнерські сторінки без технічних навичок. При цьому *WordPress* має масштабовану структуру контенту: записи, сторінки, кастомні типи записів (наприклад, портфоліо, події, продукти) та власні таксономії. Завдяки цьому сайт може бути простим або складним, і структура контенту завжди адаптується до потреб конкретного проєкту [4].

Базова частина системи включає в себе управління користувачами і доступом, де кожна роль – адміністратор, редактор, автор, учасник і передплатник – має свій набір прав. Це дозволяє делегувати частину адміністративних функцій, забезпечити контроль якості контенту, а також захистити критичні елементи сайту. Додатковим елементом є медіабібліотека, яка спрощує роботу з файлами, дозволяє завантажувати й сортувати зображення, документи, аудіо та відеоматеріали. *WordPress* забезпечує базові *SEO*-можливості: дружні посилання, структурування сторінок, коректну генерацію заголовків та метаданих. Завдяки наявності величезної кількості тем сайт може легко змінювати зовнішній вигляд, а сучасні теми на основі *Full Site Editing* дозволяють змінювати кожен елемент інтерфейсу – від футера до шаблонів окремих типів записів – без написання коду.

Однак справжня сила *WordPress* розкривається через його плагіни. Це модулі, які додають нові функції – від дрібних утиліт до повноцінних систем. У сфері *SEO WordPress* перетворюється на потужну платформу завдяки плагінам *Yoast SEO, Rank Math* або *All in One SEO Pack*. Вони дозволяють налаштовувати метадані, аналізувати текст, перевіряти читабельність, генерувати XML-карти сайту та контролювати індексацію. Багато з них інтегруються з пошуковими системами й надають рекомендації для покращення контенту. Таким чином *WordPress* стає серйозним інструментом для *SEO*-фахівців і маркетологів, адже дозволяє виконувати більшість оптимізацій без сторонніх сервісів.

Питання безпеки у *WordPress* вирішується як на рівні ядра, так і за допомогою IT-фахівців та плагінів. Хоча система надає вбудовані механізми захисту, більшість проєктів використовують додаткові інструменти. *Wordfence, Sucuri* та *iThemes Security* забезпечують захист від атак на рівні файлів, баз даних і мережевих запитів. Вони включають брандмауери, моніторинг змін у файловій системі, блокування підозрілих IP-адрес, аналіз шкідливої активності та захист від *brute force*-атак на логін. Ці інструменти доповнюються обмеженням доступу до панелі адміністратора, двофакторною автентифікацією та налаштуванням обмеження прав користувачів. Безпека *WordPress* великою мірою залежить від правильного адміністрування, але екосистема плагінів дозволяє створити доволі високий рівень захисту навіть без глибоких технічних знань.

Для продуктивності *WordPress* також має багату бібліотеку рішень. Плагіни кешування, такі як *W3 Total Cache, WP Super Cache, LiteSpeed Cache* чи *WP Rocket*, дозволяють значно прискорити роботу сайту шляхом кешування сторінок, мінімізації *JavaScript* і *CSS*, оптимізації *HTML*, інтеграції зі *CDN*-сервісами та налаштуванням різних типів кешу. У поєднанні з оптимізаційними плагінами для зображень – такими як *Smush, ShortPixel* або *Imagify* – вони дозволяють зменшити обсяг медіафайлів без втрати якості та покращити показники *Core Web Vitals*. Для великих сайтів такі інструменти є критично важливими, оскільки саме продуктивність часто визначає рівень конверсій і користувацький досвід.

Однією з найбільш революційних можливостей *WordPress* стало створення інтернет-магазинів через плагін *WooCommerce*. Це повноцінний модуль електронної комерції, який підтримує товари, категорії, атрибути, оплати, доставку, купони, податки, системи управління замовленнями та аналітику. *WooCommerce* має сотні додаткових розширень, що дозволяють інтегрувати платіжні системи, служби доставки, маркетингові інструменти, *CRM*-системи та навіть складські модулі. Завдяки *WooCommerce WordPress* став платформою не лише для контентних сайтів, а й для повноцінних комерційних рішень, які

можуть працювати як маленькими магазинами, так і великими торговельними платформами з тисячами товарів.

Сфера збору даних у *WordPress* також розвинена завдяки таким плагінам, як *Contact Form 7, WPForms, Gravity Forms* або *Ninja Forms*. Ці інструменти дозволяють створювати форми будь-якої складності: від простих контактних до розгалужених анкет, форм бронювання, опитувань чи реєстрацій на події. Вони можуть інтегруватися з *CRM*, системами email-маркетингу, аналітикою та платіжними сервісами. Також їх можна використовувати для організації внутрішньої взаємодії, наприклад для подачі заявок чи внутрішніх форм підтримки.

Ще одна важлива сфера – мультимовність. *WordPress* сам по собі не має вбудованої повноцінної багатомовності, але плагіни *WPML, Polylang* і *TranslatePress* дозволяють створювати повноцінні багатомовні сайти з окремими версіями сторінок, меню, записів, таксономій та медіафайлів. Вони забезпечують переклад контенту, автоматичну синхронізацію, *SEO*-оптимізацію для різних мов і підтримку локалізації. Завдяки цим інструментам *WordPress* широко використовується для міжнародних проєктів, яким потрібні кілька мовних версій.

У сфері дизайну *WordPress* підтримує конструктори сторінок, такі як *Elementor, Divi, Beaver Builder* і *Bricks Builder*. Вони дозволяють створювати складні макети за допомогою *drag-and-drop* інтерфейсу, з сотнями віджетів і модулів для форм, галерей, таблиць цін, блоків контенту та інтеграцій. *Advanced Custom Fields* (*ACF*) став стандартом для розробників, оскільки дозволяє створювати кастомні поля, шаблони контенту та динамічні сторінки. У поєднанні з кастомними типами записів він перетворює *WordPress* на справжню *low-code* платформу.

Крім цього, *WordPress* має плагіни для резервного копіювання, міграції та відновлення. *UpdraftPlus, Duplicator* або *BackWPup* дозволяють робити регулярні резервні копії, зберігати їх у хмарі, переносити сайт між серверами або швидко

відновлювати після збою. Це важливо як для великих бізнесів, так і для невеликих сайтів, де стабільність часто є критичною.

Додатково *WordPress* підтримує форуми (*bbPress*), соціальні мережі (*BuddyPress*), платформи для навчання (*LearnDash, TutorLMS, LifterLMS*), системи для продажу цифрових продуктів (*Easy Digital Downloads*), аналітичні інструменти (*Google Site Kit*), антиспам-рішення (*Akismet*) і велику кількість інтеграцій з зовнішніми сервісами – від чатів підтримки до API-хмарних сервісів.

Таким чином *WordPress* є потужною, універсальною та надзвичайно гнучкою платформою, яка дозволяє створювати сайти будь-якої складності – від простих блогів до масштабних корпоративних систем, від персональних сторінок до інтернет-магазинів і навчальних платформ. Його сила полягає у відкритості, розширюваності та величезній екосистемі плагінів, що робить *WordPress* одним із найкращих рішень для веб-розробки незалежно від рівня складності проєкту.

## Висновки до розділу

Було здійснено огляд системи керування контентом *WordPress*, її архітектури, принципів роботи та ключових можливостей. Показано, що *WordPress* є гнучкою та розширюваною платформою, яка широко застосовується для створення різноманітних веб-ресурсів– від блогів до корпоративних систем.

Разом з тим відзначено, що модульність, наявність великої кількості сторонніх тем і плагінів, а також відкритість екосистеми призводять до підвищення кількості потенційних вразливостей. Аналіз показав, що базова функціональність *WordPress* орієнтована насамперед на зручність використання, а не на безпеку, що зумовлює потребу в додаткових засобах захисту та правильній конфігурації серверного середовища.

# 2 АНАЛІЗ НОРМАТИВНОЇ БАЗИ

## 2.1 Аналіз нормативних документів для підтримки веб-застосунків

Підтримка веб-застосунків у сучасному цифровому середовищі є багатоплановим процесом, який охоплює не лише технічну експлуатацію програмного забезпечення, але й забезпечення його безпеки, стабільності, доступності та відповідності правовим та нормативним вимогам. Веб-застосунки дедалі частіше стають ключовими інструментами надання електронних послуг, обробки фінансових та персональних даних, комунікації між організаціями та користувачами, а також інтеграції з іншими інформаційними системами. В умовах таких широких функцій будь-які збої або порушення безпеки можуть призводити до значних фінансових, правових та репутаційних втрат, що робить питання нормативного забезпечення підтримки веб-застосунків критично важливим для організацій усіх рівнів.

Нормативні документи у сфері підтримки веб-застосунків виконують функцію системи формальних правил і стандартів, які визначають вимоги до якості програмного забезпечення, надійності, безпеки даних, сумісності та доступності. Вони регламентують порядок реагування на інциденти, управління змінами, оновлення системи та контроль за дотриманням правових норм. Системне впровадження таких документів дозволяє організаціям забезпечити безперервність роботи веб-застосунків, мінімізувати технічні та юридичні ризики та підтримувати довіру користувачів.

Національне законодавство України також відіграє ключову роль у підтримці веб-застосунків. Закони «Про інформацію», «Про захист інформації в інформаційно-телекомунікаційних системах»[5], «Про основні засади забезпечення кібербезпеки України» [6] та Закон «Про захист персональних даних» встановлюють вимоги щодо обробки, зберігання та передачі інформації, контролю доступу, ведення журналів подій, резервного копіювання та реагування на інциденти. Для практичного застосування цих норм у підтримці

веб-застосунків організації розробляють внутрішні регламенти, політики безпеки, інструкції щодо оновлення та резервного копіювання, а також плани реагування на інциденти, що забезпечують відповідність законодавству та зменшують ризики юридичної відповідальності.

НД ТЗІ 2.5-010-03 [7] встановлює основні вимоги до захисту інформації, що розміщується на веб-сторінках, зокрема визначає необхідність запобігання несанкціонованому доступу, зміні, знищенню або підміні веб-контенту. Документ наголошує на важливості формування моделі загроз і визначення середовища функціонування веб-ресурсу, включаючи операційну систему, серверне ПЗ, мережеву інфраструктуру та електронні канали взаємодії. Він підкреслює, що веб-сторінка повинна мати засоби контролю цілісності, а її адміністрування – здійснюватися лише уповноваженими особами з використанням захищених каналів і механізмів автентифікації. Система повинна забезпечувати захист від типових загроз, таких як підміна сторінок, модифікація файлів, перехоплення облікових даних та втручання в канали зв’язку. Документ також наголошує на потребі обмеження прав доступу, використання актуальних програмних компонентів, регулярного оновлення ПЗ та застосування організаційних заходів, які включають ведення журналів подій, моніторинг стану безпеки та розроблення процедур реагування на інциденти. Окрему увагу приділено резервному копіюванню та відновленню працездатності веб-сайту у разі порушення його роботи або компрометації.

НД ТЗІ 2.5-004-99 [8] визначає критерії оцінки захищеності інформації від несанкціонованого доступу в автоматизованих системах і слугує базою для встановлення рівнів захисту та вимог до функціональних послуг безпеки. Документ описує класифікацію загроз, принципи оцінювання захищеності та вимоги до компонентів системи, які забезпечують ідентифікацію, автентифікацію, розмежування доступу, реєстрацію подій, контроль цілісності та стійкість до атак. Він визначає необхідність реалізації комплексу взаємопов'язаних організаційних і технічних заходів, що забезпечують узгоджений рівень захисту. Оцінювання системи здійснюється на основі аналізу

її архітектури, політик доступу, реалізованих механізмів контролю та спроможності протидіяти загрозам, що визначені у моделі порушника. Документ встановлює, що система має гарантувати цілісність, конфіденційність та доступність інформації, а також передбачати процедури управління ключами, реагування на інциденти та підтримання безпеки протягом життєвого циклу.

У поєднанні ці два документи формують узгоджену основу, на якій вибудовується комплексний захист веб-ресурсів: НД ТЗІ 2.5-010-03 задає конкретні вимоги до веб-сторінки як об’єкта захисту, тоді як НД ТЗІ 2.5-004-99 визначає загальні критерії, за якими оцінюється рівень її захищеності. Разом вони описують необхідність створення моделі загроз, застосування технічних і організаційних заходів безпеки, забезпечення цілісності та контрольованості веб-ресурсу, впровадження засобів автентифікації та розмежування доступу, ведення журналів, використання захищених каналів зв’язку, регулярного оновлення ПЗ і документування всіх процедур, що забезпечують безпеку інформації. Таким чином, вони встановлюють мінімально необхідні вимоги до захисту веб-сайтів і визначають спосіб оцінювання відповідності цих вимог у рамках побудови або атестації системи технічного захисту інформації.

Стандарти *ISO/IEC* 27001 визначають вимоги до систем управління інформаційною безпекою [9]. Для веб-застосунків вони передбачають організацію контролю доступу, ведення журналів подій, моніторинг змін у системі та формалізоване реагування на інциденти. Практична реалізація цього стандарту включає багаторівневу автентифікацію та авторизацію, шифрування даних, регулярне оновлення програмного забезпечення, аудит журналів подій, а також планування заходів превентивного реагування на кібератаки. Успішна інтеграція *ISO/IEC* 27001 у процес підтримки веб-застосунків дозволяє організаціям відповідати міжнародним аудитам та сертифікаційним вимогам.

Процеси експлуатації та супроводу веб-застосунків регламентуються також стандартами управління IT-послугами, такими як *ISO/IEC* 20000 та методологією *ITIL*. Вони визначають структуровані підходи до управління інцидентами, проблемами, змінами, релізами та рівнем наданого сервісу (*SLA*).

Практична реалізація включає документування процедур підтримки, визначення ролей та відповідальностей персоналу, використання автоматизованих систем моніторингу та збору аналітичних даних для оперативного реагування. Наприклад, у рамках *ITIL* процес управління інцидентами передбачає фіксацію будь-якого збою або несанкціонованої зміни в системі, визначення пріоритету та термінів усунення, а також звітність для керівництва. Це дозволяє організаціям скоротити час відновлення роботи системи та підвищити задоволеність користувачів.

Особлива увага приділяється захисту персональних даних, що регламентується міжнародними та національними нормативними актами, включно з *GDPR*. Для веб-застосунків це передбачає створення політик обробки персональної інформації, впровадження механізмів контролю доступу до даних, забезпечення права користувачів на видалення та перегляд своїх даних, а також повідомлення про витік інформації у встановлені терміни. Практично це реалізується через аудит доступу до системи, регулярне оновлення політик конфіденційності, навчання персоналу та впровадження технічних і організаційних заходів захисту.

Для наочності можна навести приклад інтеграції стандартів і процедур у життєвий цикл веб-застосунку. На етапі розробки застосовуються стандарти *ISO/IEC* 25000 для забезпечення якості коду та безпеки, одночасно впроваджуються рекомендації *OWASP* для попередження уразливостей. Під час експлуатації організація застосовує *ISO/IEC* 27001 для контролю доступу та ведення журналів подій, *ITIL* для управління інцидентами та проблемами, а національні закони України – для забезпечення правового захисту даних користувачів. Внутрішні регламенти регламентують конкретні дії персоналу, наприклад, проведення резервного копіювання щодня, тестування оновлень на тестовому середовищі, моніторинг доступу та реагування на інциденти безпеки протягом встановлених *SLA*. Такий комплексний підхід дозволяє організації забезпечувати стабільність роботи веб-застосунків, високий рівень безпеки та

відповідність нормативним вимогам, одночасно зменшуючи ризики фінансових та репутаційних втрат.

Отже, системний аналіз нормативних документів та інтеграція міжнародних стандартів, національного законодавства та внутрішніх регламентів у процес підтримки веб-застосунків забезпечує ефективну організацію супроводу, підвищує стабільність та безпеку системи, мінімізує технічні та юридичні ризики і сприяє довгостроковій надійності веб-застосунків у сучасному цифровому середовищі. Практична реалізація цього підходу включає документування процедур, контроль за виконанням стандартів, регулярне навчання персоналу та інтеграцію автоматизованих систем моніторингу і аудиту, що створює основу для комплексної політики підтримки веб-застосунків у будь-якій організації.

## 2.2 Основні вимоги до безпеки сайту та відповідність системи *WordPress*

Система керування контентом *WordPress*, яка є однією з найбільш розповсюджених у світі, привертає значну увагу при впровадженні в Україні у контексті нормативних вимог з технічного захисту інформації, визначених документами НД ТЗІ 2.5-010-03 та НД ТЗІ 2.5-004-99. Питання відповідності цієї системи вимогам захисту є критичним у випадках, коли *WordPress* використовується як веб-ресурс органів влади, державних підприємств або приватних структур, що працюють з інформацією з обмеженим доступом. Обидва нормативні документи встановлюють набір критеріїв та вимог до організації та технічного забезпечення захисту веб-ресурсів та автоматизованих систем, тому важливо оцінити, наскільки *WordPress* здатний забезпечити виконання цих вимог у рамках побудови комплексної системи захисту інформації.

НД ТЗІ 2.5-010-03 визначає комплекс вимог щодо захисту веб-сторінок від несанкціонованого доступу, підміни, втрати цілісності та інших ризиків,

пов’язаних з функціонуванням веб-ресурсів у відкритих мережах. *WordPress* частково виконує ці вимоги завдяки власним механізмам автентифікації, системі ролей та прав доступу, а також можливостям розширення функціоналу через плагіни безпеки. Однак слід розуміти, що базова інсталяція *WordPress* не містить повного набору інструментів, необхідних для захисту веб-сторінки відповідно до норм документа. Для відповідності вимогам щодо захисту комунікаційних каналів *WordPress* не надає вбудованого механізму шифрування трафіку, тому необхідно використовувати належним чином налаштований *HTTPS, TLS* 1.2/1.3 та захищений веб-сервер (*Nginx, Apache* або інший). На рівні адміністрування *WordPress* дозволяє обмежувати доступ до панелі керування шляхом контролю IP-адрес, використання двофакторної автентифікації та зміни стандартного *URL* входу, що відповідає напрямкам мінімізації ризиків, зазначених у НД ТЗІ.

Значна частина вимог документа НД ТЗІ 2.5-010-03 стосується забезпечення цілісності веб-ресурсу, включно з контролем змін файлів, моніторингом можливих спроб модифікації контенту, захистом від зловмисного внесення змін у скрипти або структуру сайту. *WordPress* не має вбудованого механізму аудиту цілісності, однак забезпечує можливість підключення зовнішніх рішень. Плагіни на кшталт *Wordfence* або *Sucuri* мають власні системи виявлення вторгнень, механізми перевірки контрольних сум файлів ядра та можливість відновлення їх до офіційних версій. Це дозволяє реалізувати вимоги щодо контролю цілісності на рівні веб-платформи, хоча ці рішення залишаються сторонніми й вимагають відповідальних організаційних заходів з боку адміністраторів, що передбачено нормативними документами.

Ще одним важливим блоком вимог, прописаним у НД ТЗІ 2.5-010-03, є забезпечення захисту від несанкціонованого доступу до веб-ресурсу, включаючи спроби вгадування паролів, *brute force* атаки та інші загрози, характерні для відкритих веб-сервісів. Базовими засобами *WordPress* захист від таких атак не забезпечується. Проте можливості плагінів дозволяють організувати блокування IP-адрес при багаторазових невдалих спробах входу, встановити обмеження на частоту спроб автентифікації та активувати *firewall* прикладного рівня. Це

допомагає забезпечити виконання вимог документа щодо забезпечення захисту від атак на підсистему автентифікації. У поєднанні з серверними механізмами безпеки (*Fail2ban, mod_security* тощо) можна досягти високого рівня відповідності вимогам НД ТЗІ 2.5-010-03.

Документ НД ТЗІ 2.5-004-99 визначає критерії оцінки захищеності автоматизованих систем від несанкціонованого доступу та вимоги щодо побудови системи безпеки в таких комплексах. У цьому контексті *WordPress* розглядається не як засіб захисту, а як компонент автоматизованої системи, у якій обробляється інформація. Система *WordPress* виконує частину вимог документа завдяки підтримці механізмів автентифікації, ролей, обмеження привілеїв та контролю доступу до певних функцій та частин системи. Вона також може бути інтегрована в архітектуру, яка включає контроль фізичного доступу, захист баз даних, ізольовані мережеві сегменти та криптографічні засоби захисту.

Однак слід підкреслити, що *WordPress* не є сертифікованим засобом захисту інформації й не може самостійно забезпечити криптографічні або апаратні механізми, які часто вимагаються документом НД ТЗІ 2.5-004-99. Захист бази даних *WordPress* покладається на налаштування сервера *MySQL/MariaDB*, зокрема на правильні політики доступу, шифрування збережених даних, сегментацію доступу та інші заходи, які не входять до функціоналу самої *CMS*. Таким чином, відповідність критеріям оцінювання захищеності інформації досягається лише у складі комплексної системи, де *WordPress* виконує функцію прикладного шару, а технічні засоби захисту реалізуються інфраструктурою.

Питання журналювання та моніторингу, як одні з ключових у НД ТЗІ 2.5-004, у *WordPress* вирішені частково. Базове журналювання обмежене записами про зміну паролів, оновлення системи або встановлення плагінів. Для повноцінного аудиту дій користувачів потрібні додаткові модулі, які можуть фіксувати кожну дію адміністратора — від входу до системи до редагування контенту або зміни налаштувань. Такі модулі (*Audit Log, WP Security Audit Log*

тощо) дозволяють документиувати події відповідно до вимог КСЗІ та інтегрувати процес аналізу логів у загальну модель захисту.

Вимоги документа щодо забезпечення цілісності та доступності інформації *WordPress* також забезпечує частково. Для виконання вимог щодо резервного копіювання необхідно використовувати спеціальні інструменти, які дозволяють створювати регулярні копії бази даних та файлів, зберігати їх у захищених сховищах та відновлювати у разі порушення працездатності системи. Таким чином, *WordPress* здатен бути частиною системи, яка відповідає НД ТЗІ 2.5-004, якщо додаткові організаційні та технічні заходи будуть інтегровані у загальну архітектуру інформаційної безпеки.

Таблиця 2.1 – Відповідність системи вимогам НД ТЗІ

| Група вимог | Вимоги НД ТЗІ | Можливості *WordPress* | Ступінь відповідності |
|---|---|---|---|
| 1. Автентифікація та керування доступом | Наявність механізмів автентифікації, захисту від підбору паролів, контролю прав доступу | Розвинена модель ролей і прав, підтримка складних паролів; для захисту від *brute force* потрібні плагіни (*Wordfence, Limit Login Attempts*) | Часткова відповідність (при використанні плагінів) |
| 2. Захист каналу передавання даних | Веб-сторінки, що обробляють службову інформацію, повинні передаватися захищеним каналом (*HTTPS*) | *HTTPS* забезпечується сервером, *WordPress* підтримує роботу через *TLS*, але не забезпечує шифрування самостійно | Часткова відповідність (залежить від серверної інфраструктури) |
| 3. Забезпечення цілісності веб-ресурсу | Контроль змін файлів, захист від підміни сторінок, моніторинг спроб модифікації | Немає вбудованої перевірки цілісності; забезпечується плагінами (*Wordfence, Sucuri*), які перевіряють контрольні суми | Часткова відповідність (потрібні додаткові засоби) |
| 4. Захист від НСД до панелі адміністрування | Заборона несанкціонованого доступу, обмеження спроб входу, контроль IP | Можна обмежити доступ до */wp-admin*, включити 2*FA*; для IP-фільтрації – плагіни або серверні налаштування | Часткова відповідність |

Продовження таблиці 2.1

| | | | |
|---|---|---|---|
| 5. Журналювання та аудит | Обов'язкове ведення логів подій, дій адміністраторів, інцидентів | Базове логування мінімальне; необхідні плагіни журналювання (*Audit Log, WP Activity Log*) | Часткова відповідність (при розширенні можливостей) |
| 6. Захист бази даних | Контроль доступу, сегментація, захист даних від модифікації та перегляду | *WordPress* не керує захистом БД; безпека залежить від *MySQL/MariaDB* налаштувань та ОС | Не відповідає повністю (залежить від зовнішніх засобів) |
| 7. Захист від шкідливого коду | Виявлення ін'єкцій, підміни скриптів, несанкціонованих модифікацій | Плагіни безпеки забезпечують сканування на *malware* та файли-ін'єкції | Часткова відповідність |
| 8. Резервне копіювання та відновлення | Регулярні бекапи, зберігання у захищених місцях, перевірка відновлення | Реалізується плагінами (*UpdraftPlus, BackWPup, Duplicator*); | Повна відповідність (при належній конфігурації) |
| 9. Організаційні вимоги | Наявність політик доступу, регламентів, процедур аналізу загроз, тестування КСЗІ | *WordPress* не забезпечує організаційні процедури; реалізація залежить від власника ресурсу | Не відповідає (це не функція CMS) |
| 10. Захист від *DoS/DDoS* | Вжиття технічних заходів проти перевантаження системи | Реалізується тільки зовнішніми засобами (*Cloudflare, WAF,* хостинг); не є функцією *WordPress* | Не відповідає (потрібна інфраструктура) |
| 11. Криптографічний захист | Використання сертифікованих криптозасобів, алгоритмів, сертифікатів | *WordPress* не має вбудованої криптографії; залежить від ОС, веб-сервера | Не відповідає (потрібні сертифіковані КЗІ) |
| 12. Роль у системі КСЗІ | ПЗ має працювати в середовищі із заданими політиками та захисними механізмами | *WordPress* може бути компонентом АС у складі КСЗІ | Відповідає (як складова КСЗІ) |

З урахуванням аналізу можна дійти висновку, що *WordPress* не є самодостатньою системою, здатною повністю задовольнити вимоги нормативних документів НД ТЗІ, однак не суперечить цим вимогам та може бути ефективно використаний у середовищах, де впроваджується КСЗІ. Відповідність вимогам нормативів можлива за умови застосування зовнішніх засобів захисту,

правильного налаштування інфраструктури, розробки внутрішніх політик безпеки та регулярного моніторингу. У результаті *WordPress* у складі комплексної системи захисту може забезпечити належний рівень безпеки веб-ресурсів відповідно до вимог НД ТЗІ 2.5-010 та НД ТЗІ 2.5-004.

## Висновки до розділу

*WordPress* частково реалізує критерії доступності, цілісності та автентифікації, однак не забезпечує повного набору засобів, необхідних для функціонування в системах, де обробляється інформація з обмеженим доступом. Для досягнення відповідності нормативам необхідне впровадження зовнішніх компонентів безпеки: *WAF*, журналювання, криптографічного захисту, контролю доступу на рівні ОС та серверного програмного забезпечення (ПЗ). Зроблено висновок, що *WordPress* може бути частиною КСЗІ, але лише як компонент комплексної системи, а не як самодостатній засіб захисту.

# 3 АНАЛІЗ ВРАЗЛИВОСТЕЙ WORDPRESS

## 3.1 Вразливості системи *WordPress*

Архітектура *WordPress* поєднує ядро системи, численні плагіни, теми, засоби розширення та динамічну взаємодію з базою даних і серверним середовищем. Така модульність і відкритість забезпечують великі можливості кастомізації, але разом із тим створюють широкий спектр потенційних вразливостей. Слабкі місця можуть виникати як у коді сторонніх компонентів, так і в конфігураціях, логіці роботи та параметрах серверного оточення, що робить безпеку *WordPress* багатошаровою задачею [10-16].

Одним із ключових класів ризиків є ін’єкційні атаки, що виникають внаслідок недостатньої перевірки введених даних або некоректного використання системних функцій (табл. 3.1) . Вони особливо небезпечні тому, що дають змогу напряму взаємодіяти з базою даних або операційною системою сервера. У більшості випадків джерелом таких проблем є ненадійні або застарілі плагіни, у яких розробники не використовують сучасних методів фільтрації та захисту. Розуміння принципів ін’єкцій та їхніх практичних механізмів дозволяє завчасно оцінити найкритичніші шляхи проникнення.

Таблиця 3.1 – Вразливості ін’єкцій (*Injection Vulnerabilities*)

| Група / Підтип атаки | Опис механізму | Основні джерела | Наслідки | Типові сценарії експлуатації |
|---|---|---|---|---|
| *SQL Injection (SQLi)* | Впровадження *SQL*-запитів у БД через вхідні поля | Плагіни без *prepared statements*; слабка фільтрація | Злив БД; створення адмін-акаунтів; видалення даних | Ін’єкція параметрів у *URL*, форми, *AJAX* |
| *Blind SQLi* | *SQLi* без відображення відповіді | Ті ж причини | Повільний, але повний контроль над БД | *Boolean/time-based* експлуатація |
| *NoSQL Injection* (рідко) | Ін’єкція у сторонніх API | Плагіни з *Firebase, MongoDB* | Неавторизований доступ до *API* | Модифікація конфігурацій |
| *Command Injection* | Виконання системних команд через *PHP*-функції | Плагіни, що запускають *shell*-команди | Повний контроль над системою | Виклик *system*(), *shell_exec*() через параметри |

Продовження таблиці 3.1

| | | | | |
|---|---|---|---|---|
| *Header Injection* | Ін’єкція *HTTP*-заголовків | Неперевірені параметри редіректів | Перенаправлення на *phishing* | Зміна *Location header* |
| *Email Header Injection* | Ін’єкція у форми контактів | Слабкі плагіни форм | Масова розсилка *SPAM* від імені сайту | Вставка *\nBcc:* |

Іншим великим сегментом загроз є атаки, засновані на впровадженні та виконанні стороннього сценарного коду у браузері користувача (табл. 3.2). Оскільки *WordPress* активно працює з формами, редакторами, віджетами, медіа-файлами та інтерактивними елементами, помилки в логіці обробки контенту можуть дозволити зловмиснику втручатися в роботу сайту або отримувати конфіденційні дані. У таких випадках ризики виникають не лише на рівні плагінів, але й на рівні механізмів взаємодії фронтенду та бекенду.

Таблиця 3.2 – *XSS*-атаки (*Cross-Site Scripting*)

| Підтип *XSS* | Механізм | Джерела | Наслідки | Типові вектори |
|---|---|---|---|---|
| *Stored XSS* | Шкідливий код зберігається у БД | Форми, коментарі, плагіни | Крадіжка сесій; перехоплення адмін-панелі | Додавання *JS* у пост, коментар |
| *Reflected XSS* | Сценарій відображається одразу | *URL*-параметри | Фішинг; redirection | *JS* у *URL* |
| *DOM-based XSS* | Уразливість у клієнтському *JS* | Плагіни з *JS*-логікою | Виконання *JS* без участі сервера | *location.hash, document.write* |
| *XSS* через *SVG* | Вбудований *JS* у *SVG*-файли | Імпорт медіафайлів | Повний *JS*-контроль | *SVG* з *<script>* |
| *XSS* у *REST API* | Некоректні ендпоінти | Вразливі плагіни | Впровадження контенту | *POST*-запити з полями |

До суттєвих загроз належать також атаки на автентифікацію та управління сесіями (табл. 3.3). Використання слабких паролів, логінів за замовчуванням, недостатній захист cookies та відсутність обмежень на спроби входу створюють сприятливі умови для компрометації облікових записів, особливо адміністративних. У контексті *WordPress* це часто призводить до негайної втрати

контролю над усім сайтом. Експлуатація механізмів *XML-RPC*, злитих паролів або несекурних сесій є поширеними інструментами атак.

Таблиця 3.3 – Вразливості автентифікації та сесій

| Вразливість | Механізм | Джерела | Наслідки | Приклади експлуатації |
|---|---|---|---|---|
| *Brute Force* | Масовий підбір паролів | *wp-login.php, XML-RPC* | Компрометація адміна | 10*k* запитів через *XML-RPC* |
| *Credential Stuffing* | Використання злитих паролів | Слабкі паролі | Масові злами | Паролі зі злитих баз |
| *Session Hijacking* | Перехоплення cookies | Відсутність *HTTPS* | Повний контроль акаунта | Використання викрадених *cookies* |
| *Session Fixation* | Нав'язування *session-id* | Слабкі плагіни | Викрадення сесії | Передача сесії у *URL* |
| *Weak Password Policies* | Використання простих паролів | *Admin:admin* | Легкий злам | Пробні паролі |

Важливим джерелом ризиків залишається обробка файлів (табл. 3.4). Завантаження користувацьких матеріалів, системи імпорту, файлові менеджери та сторонні медіа-плагіни інколи дозволяють проводити впровадження шкідливих файлів або отримувати доступ до внутрішніх ресурсів сервера. Саме в цьому напрямку найчастіше виникають випадки віддаленого виконання коду, оскільки навіть незначна помилка в перевірці *MIME*-типів чи розширень дає можливість обходу запобіжників.

Таблиця 3.4 – Вразливості файлової системи та завантаження файлів

| Вразливість | Механізм | Джерела | Наслідки | Вектори |
|---|---|---|---|---|
| *Insecure File Upload* | Завантаження *.php* | Плагіни галерей | *RCE*, бекдори | *Upload shell.php.jpg* |
| *Directory Traversal* | Використання . . / | Плагіни файлових менеджерів | Доступ до системних файлів | *../../wp-config.php* |
| *RCE* через *ZIP/Theme Upload* | Завантаження плагінів з кодом | Адмін або вразливий плагін | Повний контроль | *theme.zip з malware* |

Продовження таблиці 3.4

| | | | | |
|---|---|---|---|---|
| *Local File Inclusion (LFI*) | Включення локальних файлів | Вразливі теми | Перегляд конфіденційної інформації | *include*($_*GET*) |
| *File Permissions Misconfig* | Права 777 | Неправильні права | Впровадження шкідливих файлів | Написання до *wp-content* |

На логічному рівні особливо небезпечними є вразливості, пов’язані з авторизацією, перевіркою прав доступу та бізнес-логікою процесів (табл. 3.5). Некоректно реалізовані перевірки ролей, відсутність захисту запитів ключами nonce, похибки у *REST API* або неправильне трактування дій користувачів здатні призвести до несанкціонованих змін структури сайту, контенту або конфігурації. У багатьох випадках такі вразливості складно виявити без аудиту, оскільки вони пов’язані не з технічними, а з логічними помилками.

Таблиця 3.5 – Вразливості логіки застосунку та авторизації

| Вразливість | Опис | Джерела | Наслідки | Вектори |
|---|---|---|---|---|
| *Privilege Escalation* | Отримання ролі admin | Логічні помилки | Повний контроль сайту | Підміна ролі у реєстрації |
| *Access Control Bypass* | Обхід перевірок доступу | *REST API*, плагіни | Отримання закритих даних | Запит без авторизації |
| *Business Logic Flaws* | Помилки логіки | Кастомні плагіни | Некоректні операції | Подвійні запити, |
| *Unauthorized Content Modification* | Зміна контенту | Вразливі редактори | Несанкціоновані публікації | Вставка *JS* у пост |

Окремою категорією виступають ризики, що виникають унаслідок компрометації розробників або ланцюга постачання (табл. 3.6). Через відкритість екосистеми *WordPress* та велику кількість сторонніх авторів зловмисники нерідко отримують контроль над популярними плагінами й темами, додають у них прихований код або поширюють підроблені версії. Це робить атакуваний

сайт вразливим навіть у разі технічно коректної конфігурації та якісного коду ядра системи.

Таблиця 3.6 – *REST API* та *AJAX* вразливості

| Вразливість | Механізм | Джерела | Наслідки | Сценарії |
|---|---|---|---|---|
| *REST API Unauthorized POST* | Доступ до *POST* без авторизації | Тема/плагін | Зміна записів | *POST /wp-json/wp/v2/posts* |
| *AJAX Nonce Missing* | Відсутній nonce-токен | Плагіни форм | *CSRF*, зміна налаштувань | *AJAX*-запит без nonce |
| *Overly Permissive Endpoints* | Надмірні права | Плагіни | Отримання особистих даних | Виток email, phone |
| *Unrestricted Metadata Exposure* | Виток метаданих користувачів | *REST API* | Підбір логінів | *Enumeration* |

Значну роль відіграє і стан серверного середовища (табл. 3.7). Застарілі версії PHP, неправильно налаштовані веб-сервери, відкриті порти бази даних, відсутність TLS або неправильні параметри кешування можуть створювати критичні вразливості незалежно від того, наскільки добре захищений сам WordPress. Багато атак у цьому сегменті складно запобігти без контролю над інфраструктурою, особливо у випадках дешевих або некерованих хостингів.

Таблиця 3.8 – Вразливості в темах, плагінах та ланцюгу постачання

| Вразливість | Механізм | Джерело | Наслідки | Сценарій |
|---|---|---|---|---|
| *Compromised Plugin* | Плагін з вбудованим malware | Купівля/злом автора | Масове зараження | *Auto-update* включений |
| *Backdoor in Premium Themes* | Бекдори у нелегальних темах | Взломані збірки | *RCE*, ботнет | Нелегальний nulled theme |
| *Malicious Auto-Updates* | Підміна оновлень | MITM атак | Впровадження коду | Заміна *ZIP* |
| *Obfuscated Code* | Прихований *JS/PHP* | «чорні» теми | Системний бекдор | *eval(base64_decode())* |

Окрім серверних ризиків, існує низка конфігураційних недоліків, які пов’язані виключно з експлуатацією системи: неправильні права доступу до файлів, активний режим відладки, відкриті конфігураційні файли, *editor*-доступ до коду тем та плагінів через адмінпанель (табл. 3.8). Вони не потребують складних технічних навичок від зловмисника, але можуть стати першою ланкою у ланцюгу атаки.

Таблиця 3.8 – Конфігураційні вразливості *WordPress*

| Вразливість | Причина | Наслідки | Сценарій |
|---|---|---|---|
| Використання *admin/admin* | Стандартний логін | Миттєвий злам | Перебір пароля |
| *Editor Enabled (Theme Editor)* | Редактор файлів у адмінці | Ін’єкція коду | Вставка шкідливого *PHP* |
| Відкритий *wp-config.php* | Режим індексації | Витік ключів | Перегляд файлу у браузері |
| *Debug Mode ON* | Вивід помилок | Витік шляхів | Відображення *stack trace* |
| Права 777 | Надмірні дозволи | Запис *shell*-файлів | Впровадження *PHP* |

У сукупності вразливості *WordPress* охоплюють усі рівні архітектури – від введення даних користувача та авторизації до файлових систем і серверних компонентів (табл. 3.9). Їх природа варіюється від стандартних веб-ризиків до специфічних проблем, характерних саме для екосистеми (табл. 3.10) *CMS*. Тому для забезпечення безпеки недостатньо оновлювати лише ядро *WordPress* – потрібен комплексний підхід, який включає аудит плагінів, аналіз логічних помилок, захист серверної інфраструктури та впровадження політик безпеки.

Таблиця 3.9 – Серверні та оточувальні вразливості

| Вразливість | Механізм | Джерела | Наслідки |
|---|---|---|---|
| Стара версія *PHP* | Невиправлені *CVE* | *PHP* 5.x | *RCE, DoS* |
| Відкриті порти *MySQL* | Доступ ззовні | Помилки адміністрування | Компрометація БД |
| Відсутність *HTTPS* | Передача *cookies* відкрито | Недостатня конфіденційність | Викрадення сесій |

Продовження таблиці 3.9

| | | | |
|---|---|---|---|
| *Misconfigured Apache/Nginx* | *Directory Listing* | Перегляд файлів | Лістинг *wp-content* |
| *Shared Hosting Exploits* | Відсутність ізоляції | Недоліки хостерів | Крос-акаунт атаки |

Таблиця 3.10 – Атаки на кеш, оптимізацію та сторонні інтеграції

| Вразливість | Механізм | Джерело | Наслідки |
|---|---|---|---|
| *Cache Poisoning* | Підміна кешованих сторінок | *CDN*, кеш-плагіни | Підміна контенту |
| *CDN Misconfig* | Неправильні налаштування | *Cloudflare, Akamai* | Витік IP сервера |
| *Webhook Attack* | Модифікація *webhook*-запитів | *WooCommerce, CRM* | Масові фальшиві замовлення |
| *OAuth Misuse* | Некоректні *redirect_uri* | Інтеграції соцмереж | *Account takeover* |

Представлені структури класифікації доповнюють цей опис, деталізуючи конкретні підтипи атак, типові механізми їх експлуатації, потенційні наслідки та найефективніші заходи протидії. Разом вони створюють повноцінну карту ризиків, придатну для моделювання загроз, формування вимог до безпеки, оцінки вразливостей *WordPress*-сайтів і розробки комплексних рекомендацій щодо їхнього захисту.

## 3.2 Огляд *WPScan*

### 3.2.1 Загальна характеристика *WPScan*

*WPScan* є спеціалізованим інструментом автоматизованого аудиту безпеки вебсайтів, побудованих на платформі *WordPress* . На відміну від універсальних сканерів вебвразливостей, *WPScan* орієнтується саме на особливості архітектури *WordPress*, його плагінів, тем, стандартних директорій, *API*-взаємодії та серверних механізмів. Така спеціалізація забезпечує високу точність виявлення вразливостей і дозволяє використовувати *WPScan* як галузевий стандарт у сфері аналізу безпеки *CMS WordPress* [2-4].

Основою інструмента є *WPScan Vulnerability Database* – централізована база даних вразливостей, що містить тисячі записів про *CVE*-ідентифіковані та дослідницькі вразливості ядра *WordPress*, плагінів, тем і компонентів інфраструктури. База оновлюється щоденно і виконує роль знаннєвої моделі, яка забезпечує механізм зіставлення знайдених компонентів із наявними *CVE*-записами.

*WPScan* використовується у таких контекстах, як первинна оцінка безпеки вебресурсу, пентестинг, аудит відповідності політикам безпеки, регулярний моніторинг стану компонентів *WordPress*, оцінка вразливості до атак *muny brute-force* та аналіз конфігураційних недоліків. Завдяки цьому інструмент широко застосовується компаніями, які управляють сайтами на *WordPress*, фахівцями SOC, аудиторами безпеки, розробниками та адміністраторами вебресурсів.

3.2.2 Принципи функціонування *WPScan*.

Робота *WPScan* ґрунтується на спеціалізованому аналітичному підході, спрямованому на комплексне дослідження компонентів системи *WordPress*, виявлення їх технологічних характеристик і визначення рівня їхньої безпеки. Інструмент використовує багаторівневу модель аналізу, що включає сигнатурні методи, фінгерпринтинг, евристичну оцінку поведінки серверної частини та зіставлення результатів із централізованою базою знань про вразливості. Такий підхід дозволяє *WPScan* виконувати роль високоспеціалізованого сканера, що здатен набагато глибше та точніше аналізувати систему *WordPress*, ніж універсальні сканери вебвразливостей.

Одним із фундаментальних принципів функціонування *WPScan* є фінгерпринтинг, який передбачає визначення версій ядра, плагінів і тем *WordPress* на основі характерних структур, файлів і сигнатур. *WPScan* аналізує *HTML*-вивід вебсторінок, структуру каталогу /*wp-includes*, а також шляхи до статичних ресурсів, зокрема *JavaScript*- і *CSS*-файлів, які можуть містити специфічні ознаки певної версії компонента. Крім того, інструмент може використовувати контрольні суми файлів і поведінку серверних відповідей, що

дозволяє визначати версію навіть тоді, коли розробники намагалися приховати її шляхом видалення стандартних ідентифікаторів.

Важливу роль відіграє аналіз конфігураційної безпеки. *WPScan* не обмежується пошуком програмних вразливостей, а оцінює готову інфраструктуру. Під час роботи інструмент перевіряє наявність відкритих директорій, каталогу з увімкненою індексацією, доступних службових файлів, таких як *readme.html*, *license.txt*, файлів резервних копій або файлів журналів (*debug.log*). Подібні файли можуть містити конфіденційну інформацію або відкривати додаткові канали атаки для зловмисника. У більшості випадків їхня доступність є прямим порушенням вимог безпеки, а *WPScan* дозволяє швидко ідентифікувати такі недоліки.

Окрему увагу *WPScan* приділяє аналізу *REST API* та *XML-RPC* – двох механізмів, які можуть стати джерелом серйозних вразливостей у *WordPress*. Інструмент перевіряє, які *API*-методи доступні для сторонніх користувачів, чи можливий доступ до приватних даних, чи не можна здійснити небажані дії від імені користувача. У випадку з *XML-RPC WPScan* здатен визначити, чи можна через *API* виконувати масові *brute-force* атаки, здійснювати *pingback*-атаки або використовувати функціонал для обходу системних обмежень доступу.

Функціонування *WPScan* також передбачає моделювання ризиків, пов’язаних із системою автентифікації. Інструмент вміє визначати наявність користувачів через непрямі ознаки, включаючи авторські архіви, відповіді *REST API* та дані *RSS*-каналів. У багатьох випадках навіть сама наявність знання валідних логінів значно спрощує *brute-force* атаки. Після визначення імен користувачів *WPScan* може виконувати тест на стійкість паролів, перевіряючи їх складність і використання слабких словесних шаблонів (завжди лише за прямого дозволу користувача). Така перевірка є важливою складовою комплексного аудиту безпеки, оскільки велику кількість компрометацій *WordPress*-сайтів здійснюють саме через слабкі облікові записи.

Невід’ємним компонентом *WPScan* є його інтеграція з *WPScan Vulnerability Database*, яка містить тисячі записів про вразливості, знайдені в

ядрах *WordPress*, плагінах, темах та компонентах інфраструктури. Через *API* інструмент отримує актуальну інформацію про *CVE*-ідентифікатори, опис вразливості, рівень її критичності та методи експлуатації. Таким чином *WPScan* не просто визначає стан системи – він надає детальний контекст для прийняття управлінських рішень щодо оновлення, зміцнення захисту чи зміни конфігурацій.

3.2.3 Можливості *WPScan* як інструмента безпекового аудиту.

*WPScan* виступає комплексним інструментом для виконання широкого спектра завдань у сфері безпеки *WordPress*-сайтів (табл. 3.11). Його основні можливості зосереджені на виявленні вразливих компонентів, аналізі конфігураційних помилок, оцінюванні інфраструктурних ризиків та проведенні тестування стійкості механізмів автентифікації. Інструмент дозволяє здійснювати глибокий аналіз ядра *WordPress*, визначаючи точну версію та перевіряючи її на предмет відомих вразливостей. Ця функція є критично важливою, оскільки застарілі версії ядра часто містять серйозні уразливості, які дозволяють здійснювати віддалене виконання коду, *XSS*-атаки або підвищення привілеїв.

Таблиця 3.11 – Функціональні можливості *WPScan*

| № | Функція *WPScan* | Опис можливості | Практичне призначення |
|---|---|---|---|
| 1 | Фінгерпринтинг ядра *WordPress* | Визначення версії *CMS* за сигнатурами та ресурсами | Виявлення відомих вразливостей, оцінка актуальності оновлень |
| 2 | Ідентифікація плагінів | Аналіз директорій та ресурсів для виявлення інстальованих плагінів | Перевірка наявності вразливих або застарілих компонентів |
| 3 | Визначення версій плагінів | Аналіз кодових сигнатур та файлів компонентів | Коректне зіставлення з базою *CVE* |
| 4 | Ідентифікація тем | Виявлення активних та прихованих тем *WordPress* | Аналіз стану безпеки інтерфейсних компонентів |
| 5 | Виявлення конфігураційних помилок | Перевірка доступних файлів, резервних копій, debug-log | Зниження ризиків витоку конфіденційних даних |
| 6 | Аналіз *REST API* | Перевірка відкритих роутів і доступності ресурсів | Виявлення *API*-вразливостей і витоку інформації |
| 7 | Аналіз *XML-RPC* | Виявлення доступності небезпечних функцій | Запобігання *brute-force* та *pingback*-атакам |

Продовження таблиці 3.11

| | | | |
|---|---|---|---|
| 8 | Перерахування користувачів | Визначення валідних логінів різними методами | Підготовка до оцінки ризиків атаки *brute-force* |
| 9 | Тестування паролів | Перевірка стійкості автентифікації | Визначення слабких паролів і акаунтів із високим ризиком |
| 10 | Аналіз серверного оточення | Виявлення відкритих портів, конфігураційних недоліків | Підвищення загальної безпеки інфраструктури |
| 11 | Інтеграція з *Vulnerability DB* | Автоматичний доступ до актуальних *CVE* | Забезпечення точного й актуального аудиту |
| 12 | Генерація звітів | Формування структурованих результатів сканування | Використання у звітності, аудитах та документації |
| 13 | Інтеграція з *CI/CD* | Можливість автоматичного сканування в пайплайнах | Забезпечення безпеки на етапах розробки |

Окремий напрям можливостей *WPScan* пов’язаний з аналізом плагінів та тем. Плагіни є основним джерелом вразливостей у *WordPress*-екосистемі, оскільки багато з них створюються сторонніми розробниками та не проходять суворого контролю безпеки. *WPScan* ідентифікує кожен плагін, визначає його версію та звіряє її з базою *CVE*-вразливостей. Таким чином користувач може отримати інформацію про відомі недоліки, рівень критичності, потенційні шляхи експлуатації та рекомендації щодо оновлення або видалення компонентів. Аналогічний підхід застосовується до тем, які також можуть містити небезпечний код, незахищені файли або помилки в бізнес-логіці.

Інструмент також має засоби перевірки конфігураційної безпеки, що включає аналіз доступних службових файлів, неправильних дозволів на файловій системі, наявності резервних копій і тестових директорій. *WPScan* може виявляти помилки, які не є власне програмними вразливостями, але створюють потенційно небезпечні ситуації. Наприклад, відкритий файл *wp-config.php* може розкрити дані про базу даних, що призведе до повної компрометації сайту.

Крім того, *WPScan* має засоби для оцінювання безпеки механізмів автентифікації, включаючи визначення користувачів і тестування міцності їхніх паролів. Ця функція доповнює комплексний підхід до аналізу поверхні атаки, оскільки саме слабкі паролі є одним із найпоширеніших шляхів несанкціонованого доступу до *WordPress*-сайтів.

Інструмент також здатен генерувати звіти у форматах, придатних для використання у службових документах, таких як протоколи аудиту, аналітичні довідки та розділи дипломних робіт. Це значно підвищує його практичну цінність у процесі проведення організаційних та технічних оцінок безпеки.

Не менш важливою є можливість інтеграції *WPScan* у *CI/CD*-процеси, що дозволяє автоматизувати перевірку безпеки під час розробки чи оновлення вебресурсів. У рамках *DevSecOps*-підходу це дозволяє виявляти уразливості до розгортання коду на виробничих серверах, що знижує ризики появи уразливих або небезпечних компонентів у публічному середовищі.

Завдяки такому широкому набору можливостей *WPScan* є одним із найефективніших інструментів аудиту системи *WordPress*. Він дозволяє проводити комплексний аналіз без надмірного навантаження на сервер, забезпечує точну діагностику стану системи та надає рекомендації на основі актуальних відомостей про вразливості. Усі ці характеристики роблять *WPScan* стандартним інструментом для фахівців із кібербезпеки, які працюють з *WordPress*-сайтами.

Висновки до розділу

Ідентифіковано ключові класи загроз: ін’єкції, *XSS*-атаки, *CSRF*, неправильні конфігурації серверів, вразливості в плагінах і темах, проблеми з автентифікацією та сесіями, неконтрольовані завантаження файлів, атакованість *REST API* та *XML-RPC*. Особливу увагу приділено тому, що основною причиною більшості критичних інцидентів є не ядро *WordPress*, а сторонні плагіни та недоліки конфігурацій. Окремо розглянуто можливості *WPScan* як галузевого інструмента для виявлення відомих вразливостей, що дає змогу системно оцінювати рівень безпеки веб-ресурсу та проводити регулярні аудити.

# 4 РОЗРОБКА РЕКОМЕНДАЦІЙ ЗАХИСТУ

## 4.1 Заходи захисту для системи *WordPress*

### 4.1.1 Захист від *SQL Injection*.

Для мінімізації ризиків, пов'язаних із *SQL Injection*, у системі *WordPress* використовується механізм параметризованих запитів, реалізований через функцію $*wpdb->prepare*(). Такий підхід забезпечує розділення даних і команд, що унеможливлює впровадження шкідливого *SQL*-коду в структуру запиту. Додатковим етапом захисту є санітизація вхідних даних за допомогою внутрішніх функцій *WordPress*, зокрема *sanitize_text_field*(), *sanitize_email*(), *sanitize_key*(), які перевіряють і очищують дані користувача від небажаних символів. Застосування цих механізмів відповідає вимогам НД ТЗІ щодо забезпечення цілісності інформації та недопущення її несанкціонованої модифікації.

### 4.1.2 Захист від *XSS*-атак.

Ключовим елементом протидії *XSS* є фільтрація та екранування даних перед виведенням у браузер користувача. *WordPress* застосовує функції *esc_html*(), *esc_attr*(), *esc_url*(), які запобігають вставленню небезпечного *JavaScript*-коду у форму *HTML*-атрибутів чи текстових блоків. Крім того, політика *Content Security Policy* (*CSP*) забезпечує додатковий рівень захисту за рахунок блокування виконання сторонніх або *inline*-скриптів, обмежуючи можливість ін'єкції шкідливого коду. За нормами НД ТЗІ цей підхід гарантує конфіденційність і захищеність даних користувачів.

### 4.1.3 Запобігання *XXE*-атакам.

Захист від *XXE*-вразливостей полягає у відключенні можливості обробки зовнішніх ентитетів у стандартних *XML*-парсерах, таких як *libxml*. Додатково рекомендується обмежити використання *XML* у місцях, де можливе надсилання користувачем неконтрольованих файлів, застосовуючи перевірку та валідацію структури перед обробкою. Використання безпечних форматів, таких як *JSON*,

мінімізує ризики впровадження стороннього коду, що відповідає вимогам НД ТЗІ з технічного захисту інформації.

4.1.4 Захист від *CSRF*.

Протидія *CSRF* реалізується у *WordPress* через використання унікальних токенів, які генеруються для кожної дії, що потребує авторизації. Ці токени перевіряються під час виконання запиту і забезпечують підтвердження того, що дію ініціював саме автентифікований користувач. Додатковим захисним механізмом є перевірка значень заголовків *Origin* та *Referer*. Таке рішення повністю узгоджується з вимогами НД ТЗІ щодо запобігання несанкціонованим діям у системі.

4.1.5 Захист від *brute force*-атак.

Для запобігання атакам перебору паролів застосовуються кілька взаємодоповнювальних засобів: обмеження кількості спроб авторизації, *CAPTCHA*, а також двофакторна автентифікація (2*FA*). Ці механізми значно ускладнюють можливість автоматизованого підбору облікових даних, а також забезпечують надійний контроль над доступом до системи. Нормативні документи НД ТЗІ регламентують необхідність таких заходів у системах, де обробляється конфіденційна чи службова інформація.

4.1.6 Протидія *Credential Stuffing*.

*Credential Stuffing* – це атака, що використовує злиті бази паролів. Для захисту необхідно застосовувати політику складних паролів, двофакторну автентифікацію та механізми обмеження доступу до сторінки входу, такі як зміна шляху до *wp-login.php* та обмеження частоти запитів. Це мінімізує ризики використання зловмисником раніше скомпрометованих облікових даних.

4.1.7 Запобігання *User Enumeration*.

Зловмисники можуть визначити імена користувачів через механізми *REST API,* авторські архіви або запити до *URL*. Для усунення цієї уразливості рекомендується обмежити доступ до ендпоінтів, що повертають дані про користувачів, закрити авторські архіви та стандартизувати відповіді сервера

таким чином, щоб повідомлення про помилки не розкривали інформацію щодо наявності чи відсутності певного логіна.

4.1.8 Захист від *LFI/RFI*.

Запобігання включенню локальних чи віддалених файлів передбачає фільтрацію та валідацію параметрів, що містять шляхи до файлів, а також заборону виконання *PHP* у каталогах для завантаження файлів. Забезпечення правильних прав доступу та заборони на виконання скриптів у директорії /*wp-content/uploads*/ є обов'язковими заходами. Такий підхід виключає можливість розміщення та виконання шкідливого коду у файловій системі.

4.1.9 Протидія *Path Traversal*.

*Path Traversal* ґрунтується на використанні маніпуляцій із шляхами файлів. Для запобігання цьому застосовується whitelist дозволених файлів, а також нормалізація шляхів через *realpath*(). Забороняється використання параметрів, що містять конструкції типу ../. Ці дії забезпечують дотримання політики доступу та усувають можливість обходу файлових обмежень.

4.1.10 Захист від небезпечного завантаження файлів.

Захист від *Insecure File Upload* передбачає комплексний підхід, що включає перевірку *MIME*-типів, сигнатур файлів, а також обмеження допустимих форматів. Крім того, необхідно забезпечити розміщення завантажених файлів у директоріях, де виконання *PHP*-файлів заборонено. Це відповідає вимогам НД ТЗІ щодо контролю доступу до файлів і недопущення розміщення шкідливого коду.

4.1.11 Протидія *DoS/DDoS*.

Для забезпечення доступності системи застосовуються засоби захисту від *DoS/DDoS*-атак, такі як веб-додатковий екран (*WAF*), механізми кешування, обмеження інтенсивності запитів та часткове або повне блокування доступу до *xmlrpc.php*. Використання таких механізмів дозволяє знизити навантаження на сервер та запобігти порушенню доступності веб-ресурсу.

4.1.12 Захист від *Privilege Escalation*.

Зловживання правами користувачів запобігається шляхом регулярного аудиту ролей, обмеження доступу до адміністративних функцій та контролю над встановленими плагінами, які можуть змінювати політику доступу. НД ТЗІ підкреслює важливість регулярного перегляду прав у системах з багаторівневою структурою користувачів.

4.1.13 Обмеження *REST API*.

*REST API* може стати джерелом витоку даних або каналом атаки. Для запобігання цьому необхідно контролювати доступ до *API*, вимагати автентифікації, застосовувати rate limiting та вимикати непотрібні ендпоінти. Це дозволяє суттєво зменшити площину атаки та убезпечити систему від несанкціонованих запитів.

4.1.14. Захист плагінів від вразливостей.

Основна кількість критичних загроз у *WordPress* пов’язана з плагінами, тому важливим завданням є їх постійний моніторинг, оновлення та використання тільки з перевірених джерел. Застосування безпечних плагінів та видалення застарілих модулів зменшує ризики експлуатації відомих вразливостей. Це узгоджується з нормативними вимогами щодо контролю програмного забезпечення та актуальності його версій.

4.1.15 Захист конфігураційних файлів.

Конфігураційний файл *wp-config.php* містить критичну інформацію, тому необхідно обмежити доступ до нього та, за можливості, перенести за межі кореневої директорії веб-сервера. Також важливо заборонити доступ до службових файлів (*readme.html, debug.log*), які можуть розкрити структуру системи. Це відповідає вимогам НД ТЗІ щодо захисту службової інформації.

4.1.16 Усунення помилок конфігурації сервера.

Надійність функціонування системи залежить від правильності конфігурації веб-сервера. До обов’язкових заходів належать: використання протоколу *HTTPS* із сучасною версією *TLS*, впровадження механізму *HSTS*, обмеження доступу до службових директорій, правильне налаштування прав

файлів та приховування версій серверного ПЗ. Ці заходи повністю відповідають вимогам НД ТЗІ у частині забезпечення захищеності технологічного середовища.

4.1.17 Запобігання витокам конфіденційної інформації.

Для недопущення витоку даних необхідно вимкнути режим відлагодження та забезпечити зберігання логів у директоріях, недоступних з Інтернету. Також важливо обмежити доступ до резервних копій, які можуть містити чутливу інформацію. Це відповідає вимогам НД ТЗІ у частині контролю за доступом до інформаційних ресурсів.

4.1.18. Захист механізму *XML-RPC*

Файл *xmlrpc.php* може використовуватися для *DDoS, brute force* або несанкціонованої авторизації. Для підвищення рівня захищеності рекомендується обмежити доступ до нього або повністю його заблокувати, якщо цей механізм не використовується легітимними службами. Додатковий рівень безпеки забезпечує застосування *rate limiting* та перевірка автентичності запитів.

Загальні рекомендації зведено до таблиці 4.1 [18-19].

Таблиця 4.1 – Коротка характеристика рекомендацій захисту

| № | Тип вразливості | Засіб захисту |
|---|---|---|
| 1 | *SQL Injection* | Використання $*wpdb->prepare*(), санітизація введення |
| 2 | *XSS* | Екранування даних, *CSP*, фільтрація *HTML* |
| 3 | *XXE* | Вимкнення сторонніх ентитетів в *XML*-парсері |
| 4 | *CSRF* | Використання nonce-токенів, перевірка *Origin/Referer* |
| 5 | *Brute Force* | Обмеження спроб входу, *CAPTCHA*, 2*FA* |
| 6 | *Credential Stuffing* | 2*FA*, блокування *IP*, захист логін-сторінки |
| 7 | *User Enumeration* | Закриття *REST API*, приховування авторських сторінок |

Продовження таблиці 4.1

| | | |
|---|---|---|
| 8 | *LFI/RFI* | Заборона виконання *PHP* у /*uploads*/, валідація шляхів |
| 9 | *Path Traversal* | *Whitelist* файлів, фільтрація введення |
| 10 | *Insecure File Upload* | *MIME*-перевірка, обмеження типів файлів |
| 11 | *DoS/DDoS* | *WAF, rate limiting*, кешування |
| 12 | *Privilege Escalation* | Аудит прав доступу, *Role Editor* |
| 13 | *REST API Misconfiguration* | *Nonce, capability check* |
| 14 | *Plugin Vulnerabilities* | Регулярні оновлення, *WPScan*, офіційні джерела |
| 15 | *Configuration Disclosure* | Обмеження доступу до службових файлів |
| 16 | *Server Misconfigurations* | *HSTS, HTTPS*, правильні права на файли |
| 17 | *Data Leakage* | Вимкнення *debug*, контроль логів |
| 18 | *XML-RPC Abuse* | Обмеження або блокування *XML-RPC* |

## 4.2 Модель порушника

Модель порушника є базовим елементом комплексного забезпечення інформаційної безпеки веб-ресурсу на *WordPress*, оскільки дозволяє визначити характер потенційних загроз, оцінити ресурси та можливості зловмисників і сформувати адекватні заходи протидії. Аналіз різних типів порушників демонструє, що сучасні інформаційні системи стикаються з багаторівневою сукупністю ризиків, які походять як від зовнішніх акторів, так і від внутрішніх користувачів, що мають різні рівні доступу [20-22]. Тому формування комплексної моделі порушника (табл. 4.2) є необхідною умовою для створення ефективної системи управління інформаційною безпекою відповідно до вимог НД ТЗІ 2.5-004-99 та НД ТЗІ 2.5-010-03.

Таблиця 4.2 – Комплексна модель порушника

| Тип порушника | Можливості | Загрози | Заходи протидії |
|---|---|---|---|
| Зовнішній порушник з низьким рівнем підготовки | – Використання готових інструментів *WPScan, sqlmap, ботів brute force*<br>– Сканування публічних директорій<br>– Використання відомих *CVE* у плагінах і темах | – *Brute force* на */wp-login.php*<br>– Експлуатація старих вразливостей у плагінах<br>– Сканування та виявлення конфіденційних файлів (*wp-config.php.old*)<br>– Масові *XSS*-атаки | – Обмеження спроб входу, *CAPTCHA*, 2*FA*<br>– Регулярне оновлення ядра, плагінів і тем<br>– Хмарний *WAF* (*Cloudflare*, *Sucuri*)<br>– Вимкнення листингу директорій (*Options -Indexes*) |
| Зовнішній порушник середнього рівня | – Аналіз коду тем і плагінів<br>– Комбінування *SQLi/XSS/CSRF*<br>– Використання *REST API* та *XML-RPC* атак<br>– Спроби завантаження шкідливих файлів | – *SQL Injection* у плагінах<br>– *XSS* через форми коментарів<br>– *CSRF* у панелі керування<br>– Неавторизовані *POST*-запити через *XML-RPC*<br>– Завантаження *backdoor PHP*-файлів | – *REST API hardening*<br>– Обмеження *XML-RPC* або повне його вимкнення<br>– Перевірка MIME-типів і заборона виконання файлів в */uploads/*<br>– Налаштування *file permissions*: 644/755<br>– Використання *security*-плагінів (*Wordfence, iThemes Security*) |
| Зовнішній висококваліфікований атакувальник (*APT*, кримінальні групи) | – Створення власних експлойтів<br>– Атаки на рівні серверної інфраструктури<br>– Соціальна інженерія<br>– Компрометація хостингу або мережі<br>– Експлуатація 0-го дня | – Повний контроль над сервером<br>– Компрометація бази даних *MySQL*<br>– Підміна *DNS* або трафіку<br>– Масштабні атак на доступність (*DDoS*)<br>– Довготривале приховане проникнення (*backdoors, rootkits*) | – *IDS/IPS* на рівні сервера<br>– Контейнеризація *WordPress* (*Docker*)<br>– Двофакторна автентифікація з апаратними ключами (*U2F/FIDO*2)<br>– Регулярні аудит та пентест<br>– *DDoS protection* (*Cloudflare, Radware*) |
| Внутрішній порушник з обмеженими правами (Автор/Редактор) | – Доступ до панелі керування<br>– Завантаження файлів<br>– Робота з публікаціями<br>– Частковий доступ до *REST API* | – Завантаження шкідливих скриптів у вигляді зображень<br>– Використання *XSS* у редакторі<br>– Несанкціоновані зміни контенту<br>– Витік службової інформації | – Розмежування прав доступу (*RBAC*)<br>– Заборона виконання *PHP* у */uploads/*<br>– Моніторинг активності користувачів<br>– Вимкнення небезпечних *HTML*-тегів у редакторі |

Продовження таблиці 4.2

| | | | |
|---|---|---|---|
| Компрометований внутрішній користувач (угнаний акаунт) | – Повні можливості авторизованого користувача<br>– Можливість зміни пароля, *email*– Використання *REST API* від імені користувача | – Встановлення шкідливих плагінів<br>– Викрадення персональних даних<br>– Створення бекдорів<br>– Перехоплення діяльності адміністраторів | – Виявлення аномальної активності (*security log*)<br>– Обмеження встановлення плагінів лише для супер-адміністратора<br>– Жорсткі правила паролів<br>– 2*FA* для всіх ролей |
| Недобросовісний адміністратор | – Повний доступ до панелі, БД, *FTP, SSH*<br>– Можливість змін у конфігурації сервера<br>– Видалення журналів | – Видалення або зміна даних<br>– Створення прихованих акаунтів<br>– Встановлення бекдорів<br>– Повне блокування системи | – Розділення адміністративних повноважень<br>– Контроль доступу до серверів через ключі<br>– Апаратні журнальні системи (не підконтрольні адміну)<br>– Періодична ротація адміністраторів |
| Хостинг-провайдер | – Доступ до інфраструктури<br>– Доступ до фізичних серверів<br>– Доступ до даних у відкритому вигляді | – Модифікація файлів сайту<br>– Викрадення бази даних<br>– Розгортання шкідливих сервісів<br>– Порушення конфіденційності персональних даних | – Використання шифрування БД та резервних копій<br>– Контроль доступів провайдера через *SLA*<br>– Логічне розділення даних<br>– Моніторинг серверної активності |
| Автоматизовані боти та бот-мережі | – Масові атаки *brute force*<br>– Сканування портів, *URL*, директорій<br>– *DDoS*<br>– Впорскування спаму | – Виведення сайту з ладу<br>– Масові спроби входу<br>– Сканування вразливостей<br>– Зниження продуктивності системи | – *WAF + rate limiting*<br>– Honeypot на формах входу<br>– Заборона доступу з підозрілих країн/IP<br>– Автоматичне блокування ботів |
| Конкуренти або зацікавлені треті сторони | – Збір технічної інформації<br>– Аналіз структури сайту<br>– Спроби дискредитації | – Псування контенту<br>– Порушення доступності<br>– Зниження репутації організації | – Захист цілісності контенту<br>– Резервні копії<br>– Моніторинг змін файлів |
| Хакативісти, політично мотивовані групи | – Масові атаки<br>– Комбіновані вектори<br>– Пошук слабких місць у плагінах | – *Deface* сайту<br>– Поширення дезінформації<br>– Порушення роботи сервісу | – Захист від несанкціонованих змін<br>– Ізоляція адміністративної частини<br>– Свердловинний моніторинг подій безпеки (*SIEM*) |

Першою групою загроз є зовнішні порушники з низьким рівнем технічної підготовки. Ці особи, як правило, використовують доступні у відкритому доступі інструменти для сканування та атак, такі як *WPScan, sqlmap* та численні боти *brute force*. Вони не мають глибокого розуміння архітектури системи, проте завдяки автоматизації їхні дії можуть спричинити значні проблеми, особливо у випадку виявлення застарілих компонентів *WordPress* або плагінів із відомими вразливостями. Основна загроза з боку таких порушників полягає в масових атаках на механізми автентифікації, експлуатації давно відомих *CVE*, а також у доступі до конфіденційних файлів, які можуть бути залишені в публічних директоріях. Протидія цим загрозам ґрунтується на суворому контролі оновлень, впровадженні двофакторної автентифікації, застосуванні хмарних систем веб-захисту, а також належному контролі конфігурації веб-сервера.

Більш небезпечними є зовнішні порушники середнього рівня, які мають глибше розуміння структури *WordPress* і здатні аналізувати код плагінів, знаходити логічні вразливості та комбінувати кілька векторів атак. Вони можуть застосовувати *SQL Injection, XSS, CSRF*, атакувати *REST API* або *XML-RPC*, а також намагатися завантажити шкідливі файли у вигляді зображень чи архівів [23]. Такі актори часто використовують складніші схеми, включно з обходом авторизації через логічні помилки у плагінах. Загрози, які вони створюють, можуть призвести до модифікації бази даних, викрадення облікових записів, несанкціонованого оприлюднення інформації або встановлення бекдорів. Протистояти цим атакам можна шляхом комплексного контролю *API*, обмеження доступу до *XML-RPC*, фільтрації завантажень, правильного налаштування прав доступу в файловій системі, а також впровадження спеціалізованих модулів безпеки, що виконують моніторинг і блокування підозрілих дій.

Найсерйознішу групу загроз становлять висококваліфіковані зовнішні актори, до яких належать організовані кіберзлочинні групи, *APT*-оператори та інші суб’єкти, що мають значний досвід і ресурси [24-26]. Такі порушники здатні

створювати власні експлойти, застосовувати 0-day вразливості, здійснювати комбіновані багаторівневі атаки та атакувати не лише сам *WordPress*, а й серверну інфраструктуру, мережеві компоненти або навіть мережі провайдерів. Їхні атаки часто мають цілеспрямований характер, включають інженерію соціальних взаємодій, компрометацію хостингу, перехоплення трафіку або масовані *DDoS*. Такі порушники здатні утримувати довготривалий прихований доступ до системи, встановлюючи *rootkits*, бекдори та інструменти для приховування власних дій. Захист від таких загроз потребує не лише стандартних засобів посилення безпеки, але й глибокої інфраструктурної ізоляції, застосування *IDS/IPS*, контейнеризації *WordPress*, апаратної двофакторної автентифікації, повноцінного логування подій, а також регулярного професійного аудиту безпеки [27].

Окрему та досить небезпечну групу загроз становлять внутрішні користувачі з обмеженими правами. Хоча вони не мають доступу до критичних системних функцій, їхні можливості дозволяють їм завантажувати файли, редагувати контент, взаємодіяти з *REST API* та впливати на частину функціональності сайту. Компрометація або зловмисні дії таких користувачів можуть призвести до завантаження шкідливих скриптів, впровадження *XSS* через редактор, викривлення контенту або поширення конфіденційної інформації. Ефективна протидія потребує суворого розмежування прав доступу, обмеження виконання *PHP*-файлів у директоріях завантажень, ретельного моніторингу активності користувачів і контролю будь-яких *HTML*-функцій у редакторі.

Компрометовані користувачі (тобто ті, чиї облікові дані були викрадені) фактично набувають можливостей відповідної внутрішньої ролі. Вони можуть змінювати конфігурації, встановлювати шкідливі плагіни, викрадати інформацію або створювати бекдори. Оскільки на перший погляд їхні дії виглядають легітимними, саме такі порушники є найскладнішими для виявлення. Єдиною ефективною стратегією захисту є виявлення аномалій у

поведінці користувачів, застосування обмежень на встановлення плагінів, створення політик складності паролів та обов’язкове використання двофакторної автентифікації[28].

Особливе місце займає недобросовісний адміністратор, який має повний контроль над *WordPress*, базою даних, файловою системою, *SSH, FTP* і навіть конфігураціями сервера. Такий порушник може повністю вивести систему з ладу, викрасти дані, змінити конфігурацію системи безпеки, створити приховані акаунти, впровадити бекдори та повністю знищити журнали аудиту, що робить його дії практично непомітними. Протидія цьому типу загроз ґрунтується не на технічних, а на адміністративних заходах, таких як розділення повноважень, контроль доступу до серверів через апаратні ключі, запровадження журналювання на окремі недоступні адміну системи, а також регулярна ротація адміністраторів.

Додаткову групу ризиків становлять хостинг-провайдери та персонал центрів обробки даних, які мають фізичний або системний доступ до серверів і здатні переглядати або модифікувати дані на нижчому рівні. Їхні можливості дають змогу змінювати конфігурації, копіювати або перехоплювати інформацію, запускати приховані інструменти або модифікувати інфраструктуру сайту. Ефективний захист у цьому випадку передбачає використання шифрування, чітке регламентування *SLA*, контроль серверної активності та фізичну ізоляцію критичних даних.

Автоматизовані бот-мережі є одним з найпоширеніших сучасних типів порушників, оскільки вони здійснюють масові атаки *brute force, DDoS,* сканування вразливостей і спам-ін'єкції. Вони здатні швидко вивести сайт з ладу або виявити слабкі компоненти системи. Для протидії таким загрозам використовують *WAF*, обмеження частоти запитів, *honeypot*-технології та автоматичне блокування підозрілих IP-адрес [29-35].

Також слід розглядати загрози з боку конкурентів або третіх сторін, які можуть прагнути дискредитувати організацію або знизити її репутацію. Вони можуть намагатися змінювати контент сайту, поширювати недостовірну інформацію або впливати на доступність ресурсу. Для протидії необхідний моніторинг змін файлів, резервне копіювання та захист цілісності контенту.

Ще одним видом порушників є хактивісти та політично мотивовані групи, дії яких часто спрямовані не на отримання вигоди, а на привернення уваги або завдання репутаційної шкоди організації. Їхні атаки можуть бути масштабними та координованими, часто включаючи комбіновані вектори впливу. Захист від таких атак потребує ізоляції адміністративної частини сайту, контролю подій за допомогою SIEM та дотримання рекомендацій кіберстійкості.

Таким чином, комплексна модель порушника демонструє, що система *WordPress* повинна бути захищена від широкого спектра загроз, що походять від різних суб'єктів з різним рівнем технічної підготовки та різними можливостями. Розуміння ресурсів порушників, їхньої мотивації та потенційних дій дозволяє побудувати багаторівневу систему захисту, яка включає технічні, організаційні та адміністративні заходи і відповідає вимогам нормативних документів у сфері технічного захисту інформації.

## 4.3 Оцінювання ризиків використання системи

Оцінка ризиків для веб-системи *WordPress* виконується згідно з міжнародними та національними методологічними підходами до аналізу загроз та вразливостей. Зокрема, використано положення таких стандартів та практик управління безпекою *ISO/IEC* 27005:2018 – міжнародний підхід до управління ризиками інформаційної безпеки у рамках системи менеджменту ІБ.

4.3.1 Методика розрахунку ризиків

У підходах ISO 27005 ризик визначається за формулою:

$$Risk = Likelihood \times Impact$$

де:

*Likelihood* (*L*) — ймовірність реалізації загрози;

*Impact* (*I*) — потенційний збиток у разі реалізації загрози.

Для *WordPress*-інфраструктури використана 5-бальна шкала ймовірності виникнення (табл. 4.3).

Таблиця 4.3 – Шкала ймовірності

| Рівень | Значення | Опис | Типові приклади ситуацій |
|---|---|---|---|
| Дуже низький | 1 | Вкрай малоймовірна подія | Атака потребує унікального *zero-day*, який не знаходиться у вільному доступі; доступ можливий лише за збігу рідкісних умов |
| Низький | 2 | Можлива, але відбувається рідко | Поодинокі автоматизовані сканування, низькоякісний спам, випадкові технічні збої |
| Середній | 3 | Реалістична ймовірність | Типові спроби *brute force*, використання поширених вразливостей плагінів, фішинг |
| Високий | 4 | Часто можлива | Регулярні бот-атаки, систематичні сканування *WPScan*, експлуатація нових *CVE* у популярних плагінах |
| Дуже високий | 5 | Ймовірно станеться | Активні цілеспрямовані атаки, *APT*-діяльність, масові автоматизовані атаки по всьому інтернету |

Шкала збитку (I) відображає, які наслідки має успішна атака на систему *WordPress* (табл. 4.4).

Таблиця 4.4 – Шкала збитку

| Рівень | Оцінка | Характеристика | Приклади збитків |
|---|---|---|---|
| Дуже низький | 1 | Несуттєві незручності, мінімальні втрати | Тимчасове уповільнення сайту, поодинокі спроби спаму |
| Низький | 2 | Незначні порушення роботи, локальні інциденти | Модифікація окремих постів, дрібні *XSS*, частковий доступ до публічних сторінок |
| Середній | 3 | Відчутні наслідки, частковий збій роботи | Компрометація облікових записів, доступ до внутрішнього інтерфейсу, масові спам-атаки |
| Високий | 4 | Серйозні збитки, суттєва шкода діяльності | Успішний *SQLi*, витік частини БД, зупинка роботи сайту, *DDoS* високої інтенсивності |
| Дуже високий (критичний) | 5 | Катастрофічні наслідки, повна компрометація | Знищення БД, повний контроль над сервером, *root*-доступ, витік персональних даних |

4.3.2 Розрахунок ризиків в залежності від типу порушника

Для кожного типу порушника та відповідних загроз визначено початкову ймовірність ($L_0$) та початковий вплив ($I_0$). В такому разі ризик буде розраховуватись за формулою:

$$R_0 = L_0 \times I_0.$$

Після впровадження КСЗІ визначається зменшена ймовірність ($L_1$), а залишковий вплив лишається, але в певних випадках може бути зниженим ($I_1$). Залишковий ризик буде розраховуватись за формулою:

$$R_1 = L_1 \times I_1$$

Враховуючи вищесказане проведемо оцінку початкового та залишкового ризику, а результати зведемо в таблицю 4.5.

Таблиця 4.5 – Розрахунок ризиків *WordPress* за *ISO* 27005 / *NIST* 800-30

| Тип порушника | $L_0$ | $I_0$ | $R_0$ | $L_1$ | $I_1$ | $R_1$ | Загальна оцінка |
|---|---|---|---|---|---|---|---|
| Зовнішній низький рівень | 4 | 2 | 8 | 2 | 2 | 4 | Ризик істотно знижений |
| Зовнішній середній рівень | 4 | 3 | 12 | 2 | 3 | 6 | Потрібен моніторинг |
| Висококваліфікований APT | 5 | 5 | 25 | 3 | 5 | 15 | Частково контрольований |
| Внутрішній користувач | 3 | 2 | 6 | 1 | 2 | 2 | Добре контрольований |
| Компрометований акаунт | 4 | 4 | 16 | 2 | 4 | 8 | Ризик залишається середнім |
| Недобросовісний адміністратор | 5 | 5 | 25 | 4 | 5 | 20 | Один із найвищих |
| Хостинг-провайдер | 4 | 5 | 20 | 3 | 5 | 15 | Високий залишковий |
| Автоматизовані боти | 4 | 3 | 12 | 1 | 2 | 2 | Низький після КСЗІ |
| Конкуренти | 3 | 3 | 9 | 1 | 3 | 3 | Залишковий низький |
| Хактивісти | 5 | 4 | 20 | 3 | 4 | 12 | Суттєвий, потребує *WAF + SIEM* |

4.3.3 Аналіз результатів розрахунку

4.3.3.1 Зовнішній порушник з низьким рівнем підготовки

Цей тип порушника діє переважно за допомогою масових автоматизованих інструментів (*WPScan*, прості боти). Початкова ймовірність для таких атак оцінюється як висока (4), оскільки ботнети та сканери постійно здійснюють пошук вразливих сайтів. Однак шкода зазвичай не критична, оскільки такий порушник не створює власних експлойтів і не здатний експлуатувати складні механізми.

Після впровадження КСЗІ ризик значно зменшується завдяки: *WAF, rate limiting,* 2*FA,* регулярним оновленням ядра та плагінів.

Залишковий ризик знижується до низького, оскільки більшість таких атак відсікається на ранньому етапі.

4.3.3.2 Зовнішній порушник середнього рівня кваліфікації

Цей порушник здатний:

– аналізувати код плагінів;

– комбінувати *SQLi/XSS/CSRF*;

– використовувати *REST API* або *XML-RPC*.

Початковий вплив від його атак може досягати рівня 3 (середній), оскільки компрометація контенту чи доступ до внутрішнього інтерфейсу може мати суттєві наслідки. Ймовірність оцінюється як 4, оскільки плагіни *WordPress* часто містять нові вразливості.

Після впровадження КСЗІ ризик зменшується завдяки:

– обмеженню *REST API*;

– посиленню перевірки файлів;

– використанню *security*-плагінів.

Залишковий ризик залишається середнім (6), але керованим.

4.3.3.3 Висококваліфіковані групи *APT*

Цей порушник – найнебезпечніший. Він має:

– власні експлойти;

– глибокі технічні знання;

– ресурси та тривалу мотивацію;

– здатність атакувати інфраструктуру (*DNS*, мережі, сервери);

– можливість використання соціальної інженерії.

Початковий вплив максимальний (5), оскільки атака може призвести до повної втрати керування системою. Ймовірність також висока (5), особливо якщо інфраструктура є цінною цілю.

Після впровадження КСЗІ ризик знижується завдяки:

– *IDS/IPS*;

– контейнеризації *WordPress*;

– апаратним ключам автентифікації;

– *SIEM*-моніторингу.

Однак залишковий ризик залишається високим (15), що відповідає світовим стандартам: *APT* неможливо усунути повністю, їх можна лише стримувати.

4.3.3.4 Внутрішній порушник з обмеженими правами

Ймовірність атаки не максимальна (3), але вплив обмежений (2), оскільки користувач не має прав на зміну системних налаштувань. Основні загрози:

– XSS у редакторі,

– завантаження шкідливих файлів,

– витік матеріалів.

Після реалізації КСЗІ (моніторинг, фільтрація *HTML*) ризик зменшується до 2 – найнижчої з реалістичних оцінок.

4.3.3.5 Компрометований акаунт

Такий тип порушника небезпечніший, ніж звичайний внутрішній користувач, оскільки порушник може:

– змінювати *email* та пароль;

– створювати бекдори;

– встановлювати шкідливі плагіни;

– викрадати персональні дані.

Початковий ризик високий (16) через можливість значного впливу (4) та високу ймовірність (4) у випадку фішингу або слабких паролів.

Після КСЗІ ризик зменшується до 8, але повністю не усувається, оскільки людина завжди є слабкою ланкою.

4.3.3.6. Недобросовісний адміністратор

Це один із найважчих сценаріїв. Адміністратор має:

– повні права;

– доступ до БД, *FTP, SSH*;

– здатність видаляти журнали.

Ймовірність низька, але не нульова (5 для критичності, оскільки шкала визначає не моральність, а потенціал).

Вплив = 5 (повна компрометація системи).

Після КСЗІ ризик знижується лише частково (до 20), оскільки технічні засоби не можуть повністю компенсувати людський фактор.

4.3.3.7 Порушники з боку хостинг-провайдера

Мають фізичний доступ до інфраструктури.

Навіть за наявності *SLA*, шифрування та моніторингу ризик залишається середньо-високим. Такі інциденти трапляються рідко, але масштаби збитків можуть бути критичними.

4.3.3.8 Автоматизовані боти і бот-мережі

Основні наслідки: спам, *brute force*, інтенсивні *DDoS*.

Шкода від *brute force* невелика, але загроза постійна.

Після КСЗІ (*WAF*, *rate limit*, *honeypots*) ризик зменшується до мінімуму (2).

4.3.3.9 Конкуренти або треті сторони

Мотивація – дискредитація.

Вплив на роботу системи середній, але після резервних копій та моніторингу вмісту ризик знижується майже до нуля.

4.3.3.10 Хактивісти

Їхні атаки часті, гучні, спрямовані на *deface*, *DDoS*, дискредитацію.

Шкода може бути значною (4), а ймовірність підвищеною (5).

Після КСЗІ ризик зменшується до 12, але все ще потребує постійного моніторингу з боку *SIEM*.

Висновки до розділу

Розкрито комплекс організаційних і технічних заходів, спрямованих на мінімізацію ризиків, серед яких використання параметризованих запитів для запобігання *SQL*-ін’єкціям, екранування даних та налаштування *CSP* для протидії *XSS*, контроль прав доступу, обмеження спроб входу, впровадження двофакторної автентифікації, налаштування *HTTPS* з актуальним *TLS*, обмеження доступу до *wp-config.php* та службових директорій, регулярне оновлення компонентів і моніторинг через системи безпеки.

Також запропоновано модель порушника, що включає зовнішніх атакувальників, внутрішніх користувачів, хактивістів, автоматизовані бот-мережі та недобросовісних адміністраторів, і проведено оцінку ризиків для кожної категорії. На основі аналізу визначено, що найвищий рівень загроз виходить від компрометованих облікових записів і внутрішніх осіб з розширеними правами.

# ВИСНОВКИ

У ході дослідження було виконано комплексний аналіз безпеки вебресурсу на базі *WordPress*, включаючи огляд архітектури системи, типових вразливостей, загроз та механізмів їх експлуатації. Розроблено розширену таблицю вразливостей із підтипами атак, проведено аналіз відповідності вимогам НД ТЗІ 2.5-004 та НД ТЗІ 2.5-010, сформовано модель порушника та визначено його можливості, ймовірні загрози й заходи протидії.

На основі отриманих результатів було сформовано комплексну систему захисту інформації для *WordPress*, що включає технічні, криптографічні, мережеві та організаційні заходи. Значну увагу приділено засобам моніторингу, *WAF/IDS*, контролю доступу, 2*FA*, регулярному аудиту та інструментам сканування, зокрема *WPScan*.

# ПЕРЕЛІК ПОСИЛАНЬ

1. *McDonald B. WordPress Plugin Development Cookbook. Packt Publishing*, 2021.

2. *WordPress Foundation. WordPress Documentation. URL: https://wordpress.org/support/* (дата звернення: 10.12.2025).

3. *WordPress Security Team. WordPress Security Whitepaper. URL: https://wordpress.org/about/security/* (дата звернення: 10.12.2025).

4. *Wordfence. WordPress Threat Intelligence Reports. URL: https://www.wordfence.com/* (дата звернення: 10.12.2025).

5. Закон України «Про захист інформації в інформаційно-телекомунікаційних системах». – К., 2020.

6. Закон України «Про кібербезпеку України». – К., 2018.

7. НД ТЗІ 2.5-010-03. Захист інформації. Вимоги до захисту веб-ресурсів. – К.: Адміністрація Держспецзв'язку, 2003.

8. НД ТЗІ 2.5-004-99. Захист інформації. Порядок проведення робіт із створення КСЗІ. – К.: Адміністрація Держспецзв'язку, 2009.

9. ДСТУ *ISO/IEC* 27001 Системи управління інформаційною безпекою. Вимоги. – К.: ДП «УкрНДНЦ».

10. *OWASP Foundation. OWASP Top* 10 – 2021. *OWASP, 2021*.

11. *OWASP. Cheat Sheet Series. URL: https://cheatsheetseries.owasp.org* (дата звернення: 20.12.2025).

12. *Bishop M. Computer Security: Art and Science. Addison-Wesley*, 2019.

13. Sucuri Security. Website Hacked Trend Report 2022. Sucuri, 2022.

14. *Vieira M., Antunes N., Madeira H. Using Web Security Scanners to Detect Vulnerabilities in Web Applications // IEEE Security & Privacy.* – 2009.

15. *Balduzzi M., Platzer C., Holz T. A Security Analysis of WordPress Plugins // Journal of Computer Security*. – 2021.

16. *WPScan. WordPress Vulnerability Database. URL: https://wpscan.com* (дата звернення: 10.12.2025).

17. *Et cetera Docs*. Сканування вразливостей сайту на WordPress за допомогою WPScan *URL: https://etc.hneu.edu.ua/docs/wpscan/* (дата звернення: 20.12.2025).

18. Захист сайту на базі WordPress: просунуті заходи безпеки *URL: https://yak.dslua.org/articles/bezpeka-saytu-na-bazi-wordpress/* (дата звернення: 20.12.2025).

19. Повний Посібник з Безпеки WordPress – Захистіть сайт у 2025 *URL: https://cyberset.com.ua/advanced/protection/wordpress-security-guide/* (дата звернення: 20.12.2025).

20. ДСТУ ISO/IEC 27005:2018. Інформаційні технології. Методи захисту. Керування ризиками інформаційної безпеки. – К.: ДП «УкрНДНЦ», 2018.

*21. NIST SP* 800-30. *Guide for Conducting Risk Assessments. Gaithersburg: NIST*, 2012.

*22. NIST SP* 800-53. *Security and Privacy Controls for Information Systems and Organizations. Gaithersburg: NIST*, 2020.

23. *Alberts C., Dorofee A. OCTAVE Method Implementation Guide. – CMU/SEI*, 2002. *URL: https://resources.sei.cmu.edu/library/*

24. Wordfence, "Wordfence Security – Firewall & Malware Scan," Wordfence.com, URL: https://www.wordfence.com/.

25. *Getting Started with WordPress: What is WordPress?. WordPress.org, URL: https://wordpress.org/support/article/new-to-wordpress-where-to-start/*.

*26. WordPress Plugins," WordPress.org URL: https://wordpress.org/plugins*

27. *Kinsta. How to Optimize WordPress Performance, URL: https://kinsta.com/wordpress-performance/*

*28. Joomla! Documentation, Joomla.org, URL: https://docs.joomla.org*

29. *Dries, B. Mastering Drupal 9, Packt Publishing, UK*. 2022

*30. Squarespace Help Center. "Getting Started with Squarespace", URL: https://support.squarespace.com*

*31. Wix Blog. Creating a Website with Wix: A Step-by-Step Guide, URL: https://wix.com/blog*

32. *Tilda Education. How to Build a Website with Tilda, URL:https://tilda.cc/education*.

33. Чикунов П.О., Касян П.А. Розробка вебсайту факультету кібербезпеки та інформаційних технологій / Матеріали IX Всеукраїнської науково-практичній конференції: Інформаційне суспільство: проблеми та перспективи». Одеса, ОНЮА, 2024.

34. Логінова Н.І., Касян П.А. Wordpress та його безпека / Матеріали IX Всеукраїнської науково-практичній конференції: Інформаційне суспільство: проблеми та перспективи». Одеса, ОНЮА, 2024