Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя
(повне найменування вищого навчального закладу)
Факультет комп'ютерно-інформаційних систем і програмної інженерії
(назва факультету)
Кафедра кібербезпеки
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

магістр
(освітній рівень)

на тему: "Аналіз методів захисту веб додатків від кібератак"

Виконала: студентка

Спеціальності:
125 «Кібербезпека та захист інформації»
(шифр і назва напряму підготовки, спеціальності)
Кончак Марія Богданівна
підпис (прізвище та ініціали)

Керівник Лечаченко Т. А.
підпис (прізвище та ініціали)

Нормоконтроль Тимощук Д. І.
підпис (прізвище та ініціали)

Завідувач кафедри Загородна Н.В.
підпис (прізвище та ініціали)

Рецензент
підпис (прізвище та ініціали)

м. Тернопіль – 2024

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра кібербезпеки
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Загородна Н.В.
(підпис) (прізвище та ініціали)
«__» _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Магістр
(назва освітнього ступеня)

за спеціальністю 125 Кібербезпека та захист інформації
(шифр і назва спеціальності)

Студенту Кончак Марії Богданівни
(прізвище, ім'я, по батькові)

1. Тема роботи Аналіз методів захисту веб додатків від кібератак

Керівник роботи Лечаченко Тарас Анатолійович, доктор філософії., старший викладач кафедри КБ
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від «16» 11 2023 року № 4/7-1061

2. Термін подання студентом завершеної роботи 28.12.2024

3. Вихідні дані до роботи об'єкт дослідження (веб-додатки), предмет дослідження дослідження (методи і технології захисту веб-додатків від кіберзагроз), завдання дослідження (провести аналіз основних типів кібератак), надати рекомендації щодо покращення безпеки

4. Зміст роботи (перелік питань, які потрібно розробити) аналіз сучасних загроз безпеки веб-додатків та їх класифікація, порівняльний аналіз ефективності сучасних методів захисту, аналіз та обґрунтування вибору методів захисту від XSS та CSRF атак, проектування системи захисту бази даних від SQL-ін'єкцій, практична реалізація комплексної системи захисту веб-додатку, тестування та оцінка ефективності методів захисту, рекомендації щодо оптимізації та вдосконалення системи захисту.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів) аналіз методів захисту веб додатків від кібератак, актуальність, мета дослідження аналіз сучасних загроз безпеки веб-додатків та їх класифікація, інші ключові загрози, основні засоби та методи захисту, порівняльний аналіз ефективності сучасних методів захисту веб-додатків, таблиця 1.1 Порівняння методів захисту, практична реалізація комплексної системи захисту веб-додатку, тестування та оцінка ефективності впроваджених методів захисту.

Висновки.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	Осухівська Г.М., к.т.н., доцент		
Безпека в надзвичайних ситуаціях	Клепчик В.М., проректор з адміністративно-господарської роботи та будівництва		

7. Дата видачі завдання 20.09.2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	20.09 – 22.09	Виконано
2.	Формулювання мети та завдань дослідження	22.09 – 22.09	Виконано
3.	Визначення об’єкта, предмета та методів дослідження	22.09 – 22.09	Виконано
4.	Аналіз сучасних кібератак на веб-додатки	23.09 – 25.09	Виконано
5.	Аналіз сучасних векторів атак та їх впливу на безпеку веб-додатків	25.09 – 30.09	Виконано
6.	Протоколи безпечного з’єднання (HTTPS, SSL/TLS)	01.10 – 03.10	Виконано
7.	Методи перевірки та очищення вхідних даних	04.10 – 10.10	Виконано
8.	Вибір інструментів та середовища для тестування	11.10 – 15.10	Виконано
9.	Проведення симуляції атак на веб-додатки	16.10 – 24.10	Виконано
10.	Порівняння ефективності різних методів захисту	25.10 –30.10	Виконано
11.	Висновки та рекомендації	01.11 – 05.11	Виконано
12.	Нормоконтроль	08.12 – 10.12	Виконано
13.	Перевірка на плагіат	12.12 – 14.12	Виконано
14.	Попередній захист кваліфікаційної роботи	20.12 – 15.12	Виконано
15.	Захист кваліфікаційної роботи	28.12.2024	

Студент _____ Кончак М.Б.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Лечаченко Т. А.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Аналіз методів захисту веб додатків від кібератак // ОР «Магістр» // Кончак Марія Богданівна // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра кібербезпеки, група СБml-61 // Тернопіль, 2024 // С. 72, рис. – 1, табл. – 5 , кресл. – 13, додат. – 2, ліст. – 28 .

Ключові слова: HTTPS, SSL/TLS, HA-256, Cross-Site Scripting, IDS/IPS, Javascript.

Магістерська робота присвячена аналізу сучасних методів захисту веб-додатків від кібератак. У роботі розглядаються основні загрози безпеці, такі як атаки типу Cross-Site Scripting (XSS), SQL Injection, та інші вектори атак, які використовуються для компрометації веб-додатків.

Проведено дослідження сучасних технологій захисту, зокрема, застосування протоколів HTTPS із використанням SSL/TLS для забезпечення шифрування переданих даних, алгоритмів хешування SHA-256 для безпечного зберігання інформації, а також систем виявлення та запобігання вторгнень (IDS/IPS). Окрему увагу приділено методам динамічної перевірки та очищення даних на стороні клієнта із використанням мови Javascript.

У роботі запропоновано підхід до впровадження багаторівневих механізмів захисту, що дозволяє мінімізувати ризики та підвищити стійкість веб-додатків до кібератак. Розроблено рекомендації щодо інтеграції систем моніторингу та аналізу трафіку для своєчасного виявлення підозрілої активності.

Магістерська робота містить 72 сторінок, ілюстрована 1 рисунком а також включає 1 додаток та 28 лістингів.

ABSTRACT

Analysis of Web Application Security Methods Against Cyberattacks // Master's Thesis // Konchak Mariia Bohdanivna // Ternopil Ivan Puluj National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Cybersecurity, Group SBml-61 // Ternopil, 2024 // P. 72, fig. – 1, tables – 5, diagrams – 13, appendices – 2, listings. – 28 .

Keywords: HTTPS, SSL/TLS, SHA-256, Cross-Site Scripting, IDS/IPS, Javascript.

The master's thesis is devoted to the analysis of modern methods for protecting web applications from cyberattacks. The study examines major security threats, including attacks such as Cross-Site Scripting (XSS), SQL Injection, and other attack vectors used to compromise web applications.

Research was conducted on modern security technologies, including the use of HTTPS protocols with SSL/TLS for encrypted data transmission, SHA-256 hashing algorithms for secure information storage, and intrusion detection and prevention systems (IDS/IPS). Particular attention is paid to dynamic data validation and sanitization methods on the client side using Javascript.

The thesis proposes an approach to implementing multi-layered security mechanisms, which minimizes risks and enhances the resilience of web applications against cyberattacks. Recommendations for integrating monitoring and traffic analysis systems for timely detection of suspicious activity have also been developed.

The master's thesis comprises 72 pages, is illustrated with 1 figure, and includes 1 appendix and 28 listings.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

CSS	—	Cross-Site Scripting
CSRF	—	Cross-Site Request Forgery
SSRF	—	Server-Side Request Forgery
IDOR	—	Direct Object References
XXE	—	XML External Entity
RCE	—	Remote Code Execution
WAF	—	Web Application Firewall
CSP	—	Content Security Policy
JWT	—	JSON Web Token
SSDLC	—	Secure Software Development Lifecycle
RASP	—	Runtime Application Self-Protection
ACL	—	Access Control Lists
WAF	—	Web Application Firewall

ВСТУП

Актуальність роботи зумовлена стрімким зростанням кількості та складності кібератак на веб-додатки в сучасному цифровому середовищі. За даними OWASP (Open Web Application Security Project), щорічно фіксується збільшення кількості успішних атак на веб-додатки на 15-20%, що призводить до значних фінансових втрат та репутаційних ризиків для організацій.

В умовах постійного вдосконалення методів та інструментів проведення кібератак особливої актуальності набуває розробка комплексних систем захисту веб-додатків, які здатні протистояти як відомим, так і новим типам загроз. Критично цінним стає впровадження багаторівневого захисту, що включає захист від SQL-ін'єкцій, XSS-атак, CSRF-атак та інших типів вразливостей.

Необхідність розробки ефективних методів захисту веб-додатків підкреслюється зростаючою кількістю випадків витоку конфіденційних даних та компрометації користувацьких облікових записів. Це вимагає впровадження сучасних методів шифрування даних, надійних механізмів автентифікації та авторизації, а також систем моніторингу та реагування на інциденти безпеки.

Мета дослідження полягає в розробці комплексної системи захисту веб-додатків від кібератак, яка забезпечить високий рівень безпеки користувацьких даних та стабільність роботи додатку в умовах постійних загроз.

Об'єкт дослідження: процеси забезпечення безпеки веб-додатків та захисту від різних типів кібератак.

Предмет дослідження: методи та засоби захисту веб-додатків від сучасних кібератак, включаючи механізми автентифікації, авторизації, шифрування даних та моніторингу безпеки.

Методи дослідження включають системний аналіз існуючих загроз та методів захисту, експериментальне дослідження ефективності різних механізмів безпеки, а також практичне тестування розроблених рішень.

Наукова новизна одержаних результатів кваліфікаційної роботи полягає у розробці комплексного підходу до захисту веб-додатків (запропоновано багаторівневу систему захисту, яка ефективно поєднує сучасні технології

шифрування, автентифікації, авторизації, моніторингу та динамічного виявлення загроз), розширенні механізмів виявлення атак (запропоновано використання інтегрованих систем IDS/IPS для автоматичного виявлення та запобігання атакам у реальному часі, що зменшує ймовірність компрометації веб-додатків), інтеграції сучасних практик захисту у веб-додатки (досліджено та адаптовано методи очищення та перевірки даних на стороні клієнта за допомогою мови JavaScript для захисту від Cross-Site Scripting (XSS) та інших ін’єкцій.)

Практичне значення одержаних результатів магістерської роботи полягає у можливості використання розробленої комплексної системи захисту веб-додатків для підвищення їхньої стійкості до сучасних кіберзагроз. Основні аспекти практичного застосування включають: захист веб-додатків від кібератак, впровадження у реальні проєкти, моніторинг і реагування на загрози, використання у навчальних цілях, адаптація до нових загроз.

Таким чином, отримані результати сприяють підвищенню безпеки веб-додатків, мінімізації ризиків фінансових та інформаційних втрат, а також можуть бути використані як основа для розробки сучасних систем захисту у сфері інформаційних технологій.

Апробація результатів магістерської роботи: основні результати проведених досліджень обговорювались на: IV Міжнародній студентській конференції «Концепт науки XXI: стратегії, методи та наукові інструменти» (м.Вінниця, Україна).

Публікації. Основні результати кваліфікаційної роботи опубліковано у працях конференції (див. Додаток А)

РОЗДІЛ 1 ТЕОРЕТИЧНІ ОСНОВИ ЗАХИСТУ ВЕБ-ДОДАТКІВ ВІД КІБЕРАТАК

1.1 Аналіз сучасних загроз безпеки веб-додатків та їх класифікація

У сучасному світі веб-додатки стали невід'ємною частиною бізнес-процесів та повсякденного життя людей. З розвитком інформаційних технологій зростає і кількість кіберзагроз, направлених на компрометацію веб-додатків. Під терміном "кіберзагроза" розуміється потенційна можливість певної загрози порушити інформаційну безпеку системи, що може призвести до порушення конфіденційності, цілісності та доступності інформації.

Згідно з даними міжнародної організації OWASP (Open Web Application Security Project), щорічно фіксується зростання кількості кібератак на веб-додатки на 15-20%. Особливу небезпеку становлять атаки, спрямовані на отримання несанкціонованого доступу до конфіденційної інформації користувачів та критичних даних організацій.

Кібербезпека веб-додатків – це комплекс заходів, спрямованих на забезпечення захисту веб-застосунку від несанкціонованого доступу, використання, розкриття, спотворення, зміни, дослідження, записування або знищення інформації. Ця система включає в себе технічні, організаційні та правові аспекти захисту [1, с. 45].

Основними векторами атак на веб-додатки є спроби експлуатації вразливостей на рівні застосунку, сервера та мережевої інфраструктури. Зловмисники постійно вдосконалюють методи атак, використовуючи як відомі вразливості, так і розробляючи нові техніки проникнення в системи.

Cross-Site Scripting (XSS) залишається однією з найпоширеніших загроз безпеки веб-додатків. При успішній XSS-атаці зловмисник може впровадити шкідливий JavaScript-код на сторінку, який буде виконуватися в контексті браузера користувача, що може призвести до викрадення сесійних токенів та інших конфіденційних даних.

SQL-ін'єкції представляють собою метод атаки, при якому зловмисник намагається внести зміни в SQL-запити, що виконуються веб-додатком. Успішна SQL-ін'єкція може призвести до несанкціонованого доступу до бази даних, модифікації або видалення даних.

Cross-Site Request Forgery (CSRF) є типом атаки, при якій зловмисник змушує автентифікованого користувача виконати небажані дії на веб-сайті, де користувач наразі авторизований. Ця атака використовує довіру, яку веб-сайт має до браузера користувача.

Broken Authentication and Session Management залишається критичною проблемою безпеки, яка може призвести до компрометації облікових записів користувачів. Ця вразливість виникає через неправильну реалізацію механізмів автентифікації та управління сесіями.

Розподілені атаки типу "відмова в обслуговуванні" (DDoS) становлять значну загрозу для доступності веб-додатків. При такій атаці зловмисники намагаються перевантажити сервер великою кількістю запитів, що призводить до відмови в обслуговуванні легітимних користувачів [2, с. 18].

Path Traversal атаки дозволяють зловмисникам отримати доступ до файлів та каталогів, що знаходяться за межами кореневого каталогу веб-додатку. Ця вразливість може призвести до витоку конфіденційної інформації та компрометації системи. Server-Side Request Forgery (SSRF) є типом атаки, при якій зловмисник змушує серверний додаток надсилати запити до непередбачених внутрішніх сервісів, що може призвести до сканування внутрішньої мережі та отримання доступу до захищених ресурсів. Insecure Direct Object References (IDOR) виникає, коли веб-додаток надає прямий доступ до об'єктів на основі введених користувачем даних, що може дозволити зловмисникам отримати несанкціонований доступ до даних інших користувачів.

XML External Entity (XXE) атаки спрямовані на експлуатацію вразливостей в обробці XML-даних, де успішна XXE-атака може призвести до розкриття конфіденційних файлів, виконання віддалених запитів та відмови в обслуговуванні. Security Misconfiguration є однією з найпоширеніших проблем, яка виникає через неправильне налаштування параметрів безпеки веб-серверів,

фреймворків та компонентів додатку, що може створити численні вразливості, які зловмисники можуть експлуатувати. Using Components with Known Vulnerabilities залишається серйозною проблемою, оскільки багато веб-додатків використовують застарілі або вразливі компоненти третіх сторін, створюючи потенційні точки входу для зловмисників.

Remote Code Execution (RCE) є однією з найнебезпечніших загроз, оскільки дозволяє зловмиснику виконувати довільний код на сервері, часто виникаючи через неправильну обробку користувацького введення. Business Logic Vulnerabilities представляють особливий клас вразливостей, які виникають через помилки в бізнес-логіці додатку та складні для виявлення автоматизованими інструментами, оскільки вони специфічні для конкретного додатку. File Upload Vulnerabilities виникають при неправильній обробці завантажуваних файлів, де зловмисники можуть використовувати ці вразливості для завантаження шкідливого коду або перевищення обмежень системи [3, с. 112].

Race Conditions є складними для виявлення вразливостями, які виникають при неправильній обробці паралельних запитів, де зловмисники можуть використовувати ці вразливості для обходу бізнес-логіки або отримання несанкціонованого доступу. Server-Side Template Injection виникає, коли користувацький ввід потрапляє безпосередньо в шаблон серверної сторони, що може призвести до виконання довільного коду на сервері. Insufficient Logging and Monitoring є критичною проблемою, яка ускладнює виявлення та реагування на атаки, де відсутність належного логування може призвести до того, що атаки залишаються невиявленими протягом тривалого часу.

Mass Assignment є вразливістю, яка виникає, коли веб-додаток автоматично присвоює значення властивостям об'єктів на основі параметрів запиту, що може призвести до несанкціонованої модифікації даних. Open Redirect вразливості дозволяють зловмисникам перенаправляти користувачів на шкідливі веб-сайти, часто використовуючись у фішингових кампаніях. Host Header Injection атаки можуть призвести до обходу обмежень безпеки та

виконання шкідливих перенаправлень через неправильну обробку заголовка Host.

Cache Poisoning атаки спрямовані на модифікацію кешованого контенту веб-додатку, де успішна атака може призвести до доставки шкідливого контенту легітимним користувачам. Web Cache Deception є складною атакою, яка дозволяє зловмисникам отримувати доступ до персональних даних користувачів через маніпуляції з кешуванням веб-контенту. HTTP Parameter Pollution атаки використовують особливості обробки HTTP-параметрів різними компонентами веб-додатку, що може призвести до обходу механізмів безпеки та неочікуваної поведінки додатку. Subdomain Takeover є загрозою, яка виникає, коли піддомен вказує на сервіс, який більше не використовується, де зловмисники можуть захопити цей піддомен і використовувати його для атак [1].

Clickjacking залишається актуальною загрозою, яка дозволяє зловмисникам змушувати користувачів взаємодіяти з прихованим шкідливим контентом. Ця атака часто використовується для фішингу та шахрайства.

API Security Vulnerabilities становлять окрему категорію загроз, пов'язаних з неправильною реалізацією або захистом API-інтерфейсів, що може призвести до несанкціонованого доступу до даних та функціональності. Browser Cache Poisoning є специфічним типом атаки, спрямованим на модифікацію кешованого контенту в браузері користувача, що може призвести до виконання шкідливого коду при подальших відвідуваннях сайту. Client-Side State Manipulation атаки експлуатують вразливості в механізмах зберігання стану на стороні клієнта, де зловмисники можуть модифікувати дані в localStorage або cookies для обходу механізмів безпеки.

HTTP Request Smuggling є складною атакою, яка використовує розбіжності в інтерпретації HTTP-запитів різними компонентами веб-інфраструктури, що може призвести до обходу механізмів безпеки та витоку даних. DOM Clobbering атаки спрямовані на маніпуляцію DOM-структурою веб-сторінки, які можуть призвести до виконання шкідливого коду та порушення роботи клієнтської частини додатку. Privacy Violations є окремою категорією загроз, пов'язаних з порушенням конфіденційності користувацьких даних, що може включати витік

персональних даних, відстеження користувачів та порушення регуляторних вимог.

Resource Exhaustion атаки спрямовані на виснаження ресурсів сервера через надмірне споживання CPU, пам'яті або дискового простору, що може призвести до відмови в обслуговуванні та фінансових втрат. Timing Attacks використовують різницю в часі відповіді системи для отримання інформації про внутрішній стан додатку, ці атаки можуть бути використані для обходу механізмів автентифікації та отримання конфіденційної інформації. Format String Attacks залишаються актуальними для додатків, які використовують небезпечні функції форматування рядків, що може призвести до витоку пам'яті та виконання довільного коду [2].

Кожна з цих загроз вимагає специфічного підходу до захисту та постійного моніторингу. Розуміння природи та механізмів цих загроз є критичним для розробки ефективної стратегії захисту веб-додатків. Важливо відзначити, що ефективний захист повинен базуватися на принципі глибокого захисту (defense in depth) та включати в себе комплекс технічних та організаційних заходів.

1.2 Дослідження існуючих методів та засобів захисту веб-додатків

Методи та засоби захисту веб-додатків представляють собою комплексну систему технічних, програмних та організаційних рішень, спрямованих на забезпечення безпеки веб-застосунків. Під захистом веб-додатків розуміється сукупність методів, технологій та практик, що забезпечують конфіденційність, цілісність та доступність інформації, яка обробляється веб-застосунком.

Сучасні методи захисту веб-додатків базуються на принципі багаторівневого захисту (Defense in Depth), який передбачає створення декількох рівнів безпеки. Кожен рівень забезпечує додатковий захист, що значно ускладнює можливість успішної атаки на систему.

Web Application Firewall (WAF) є одним з фундаментальних засобів захисту веб-додатків. WAF являє собою програмний або апаратний засіб, що

аналізує HTTP/HTTPS трафік між клієнтом та веб-сервером, блокуючи потенційно небезпечні запити та захищаючи від відомих типів атак [4, с. 234].

Інтелектуальні системи виявлення та запобігання вторгнень (IDS/IPS) здійснюють постійний моніторинг мережевого трафіку, виявляючи підозрілу активність та автоматично блокуючи потенційні атаки. Ці системи використовують складні алгоритми аналізу та машинного навчання для ідентифікації аномальної поведінки.

Шифрування даних залишається необхідним методом захисту. Сучасні веб-додатки використовують протокол TLS для захисту даних при передачі, а також різні алгоритми шифрування для захисту даних у стані спокою. Одним з аспектів якого є правильне управління ключами шифрування.

Механізми автентифікації та авторизації забезпечують контроль доступу до ресурсів веб-додатку, де багатофакторна автентифікація (MFA) стала стандартом безпеки, що вимагає від користувача надання декількох форм підтвердження ідентичності. Безпечне управління сесіями реалізується через використання криптографічно стійких ідентифікаторів сесій, їх регулярну ротацію та правильне налаштування параметрів cookies, включаючи прапори HttpOnly та Secure. Content Security Policy (CSP) надає додатковий рівень захисту від XSS-атак та інших видів впровадження шкідливого контенту, дозволяючи точно визначити джерела, з яких браузер може завантажувати ресурси [3].

Sanitization та валідація користувацького введення є фундаментальними методами захисту, де всі дані, що надходять від користувача, повинні проходити ретельну перевірку та очищення перед використанням у додатку. Підготовлені запити (Prepared Statements) та ORM-системи забезпечують захист від SQL-ін'єкцій, гарантуючи, що користувацькі дані не можуть змінити структуру SQL-запиту. Rate Limiting та Throttling механізми захищають від DoS-атак та брутфорс-спроб, обмежуючи кількість запитів від одного клієнта за певний період часу.

Security Headers, JWT (JSON Web Tokens) та OAuth 2.0 формують комплексний підхід до безпеки веб-комунікацій. Security Headers, такі як X-Frame-Options, X-Content-Type-Options та X-XSS-Protection, забезпечують

додатковий рівень захисту на рівні браузера. JWT використовуються для безпечної передачі інформації між сторонами з правильною реалізацією алгоритмів підпису та перевіркою claims. OAuth 2.0 та OpenID Connect протоколи забезпечують стандартизований механізм делегування автентифікації та авторизації, що особливо важливо для мікросервісної архітектури.

Container Security та Security Testing Automation представляють сучасний підхід до безпеки розробки. Container Security забезпечує ізоляцію компонентів додатку та мінімізацію поверхні атаки, а Security Testing Automation включає регулярне проведення автоматизованих тестів безпеки. Secure Software Development Lifecycle (SSDLC) забезпечує врахування аспектів безпеки на всіх етапах розробки, а Zero Trust Architecture передбачає відсутність довіри до будь-яких компонентів системи за замовчуванням [5, с. 378].

API Security Gateway та Secure Configuration Management забезпечують комплексний захист інфраструктури. API Security Gateway надає централізований контроль та захист API-інтерфейсів, а Secure Configuration Management забезпечує правильне налаштування всіх компонентів системи. Runtime Application Self-Protection (RASP) та Secret Management Systems додають динамічний захист під час виконання та безпечне управління конфіденційними даними.

Secure Code Review Process, Security Logging and Monitoring, та Access Control Lists (ACL) формують основу для постійного контролю безпеки. Network Segmentation та Secure File Upload Handling забезпечують додаткові рівні захисту, а Browser Security Features та Security Headers Management оптимізують безпеку на стороні клієнта.

Vulnerability Management Process, Incident Response Plan, та Security Awareness Training складають комплексний підхід до управління безпекою. Security Compliance Monitoring та Penetration Testing забезпечують відповідність стандартам та активну перевірку захищеності. Security Metrics and Reporting, Security Architecture Review, та DevSecOps Practices інтегрують безпеку в усі аспекти розробки та експлуатації, а Third-party Security Assessment забезпечує комплексну оцінку безпеки всіх компонентів системи [6, с. 89].

Всі ці методи та засоби захисту повинні використовуватися комплексно та адаптуватися до конкретних потреб та ризиків веб-додатку. Потрібно регулярно оцінювати їх ефективність та оновлювати відповідно до нових загроз та технологій.

1.3 Порівняльний аналіз ефективності сучасних методів захисту веб-додатків

Ефективність методів захисту веб-додатків визначається як здатність протистояти різним типам кібератак при оптимальному використанні ресурсів системи. Під оптимальним використанням ресурсів розуміється баланс між рівнем захисту та продуктивністю системи [5].

Методологія оцінки ефективності захисту базується на кількісних та якісних показниках, де до кількісних показників відносяться: час виявлення атаки, кількість успішно відбитих атак, відсоток помилкових спрацьовувань. Якісні показники включають зручність впровадження, вплив на продуктивність системи та вартість підтримки. Web Application Firewall (WAF) демонструє високу ефективність у захисті від відомих векторів атак, забезпечуючи блокування до 95% типових атак, однак може створювати затримки у обробці запитів від 5 до 50 мілісекунд, що може бути критичним для високонавантажених систем.

Системи виявлення та запобігання вторгнень (IDS/IPS) показують різну ефективність залежно від реалізації, де сигнатурні системи ефективні проти відомих атак, але безсилі проти нових загроз, а системи на основі аномалій можуть виявляти нові атаки, але мають вищий відсоток помилкових спрацьовувань. Шифрування даних забезпечує практично стовідсотковий захист від перехоплення даних при правильній реалізації, проте використання складних алгоритмів шифрування може збільшити навантаження на процесор до 30%.

Багатофакторна автентифікація значно підвищує безпеку доступу, знижуючи ймовірність несанкціонованого доступу до мінімуму, де статистика показує, що MFA запобігає 99.9% атак, пов'язаних з компрометацією облікових даних.

Content Security Policy виявляє високу ефективність у запобіганні XSS-атакам, знижуючи ризик успішної атаки на 90%. Проте, складність налаштування та підтримки може призвести до помилок конфігурації [7, с. 156].

Підготовлені запити та ORM демонструють майже повну захищеність від SQL-ін'єкцій при правильному використанні. Додатковою перевагою є покращення продуктивності за рахунок кешування планів виконання запитів.

Таблиця 1.1 – Порівняння ефективності основних методів захисту веб-додатків

Метод захисту	Ефективність блокування атак (%)	Вплив на продуктивність (%)	Складність впровадження	Вартість підтримки
WAF	95	5-10	Середня	Висока
IDS/IPS	85-90	3-7	Висока	Середня
MFA	99.9	1-2	Низька	Низька
CSP	90	1-3	Висока	Середня
Шифрування	99.9	10-30	Середня	Низька

Пояснення критеріїв "Складність впровадження", "Вплив на продуктивність" тощо наведено у “Додаток Б”.

Rate Limiting показує високу ефективність у запобіганні DoS-атакам та брутфорс-спробам, знижуючи навантаження на сервер під час атак на 80-90%.

Security Headers забезпечують базовий рівень захисту з мінімальним впливом на продуктивність. Їх ефективність особливо помітна у запобіганні Clickjacking та XSS-атакам.

JWT-токени демонструють хороший баланс між безпекою та продуктивністю, забезпечуючи надійну автентифікацію з мінімальними накладними витратами.

OAuth 2.0 та OpenID Connect стали стандартом для делегування автентифікації, демонструючи високу надійність при правильній реалізації.

Контейнеризація значно підвищує загальний рівень безпеки за рахунок ізоляції компонентів, хоча і вимагає додаткових ресурсів для управління.

Автоматизоване тестування безпеки дозволяє виявляти до 80% типових вразливостей на ранніх етапах розробки, значно знижуючи витрати на їх усунення.

Zero Trust архітектура демонструє найвищий рівень захисту, але вимагає значних ресурсів для впровадження та підтримки. API Gateway забезпечує централізований захист API-інтерфейсів з високою ефективністю при помірному впливі на латентність. RASP-системи показують високу ефективність у виявленні та блокуванні атак в реальному часі, але можуть створювати значне навантаження на систему. Системи управління секретами демонструють відмінні результати у захисті конфіденційних даних при правильному налаштуванні. Процеси перегляду коду значно знижують кількість вразливостей у продакшн-середовищі, хоча і збільшують час розробки.

Логування та моніторинг безпеки дозволяють виявляти до 95% інцидентів безпеки при правильному налаштуванні. RBAC та ACL забезпечують точний контроль доступу з мінімальним впливом на продуктивність системи. Сегментація мережі значно знижує потенційний вплив успішних атак, обмежуючи їх поширення. Безпечна обробка файлів демонструє високу ефективність у запобіганні атакам через завантаження файлів. Функції безпеки браузера забезпечують додатковий рівень захисту з мінімальними накладними витратами. Управління заголовками безпеки показує хороші результати у запобіганні різним типам атак на стороні клієнта [8, с. 92].

Процес управління вразливостями дозволяє своєчасно виявляти та усувати до 90% критичних вразливостей. Плани реагування на інциденти значно знижують час відновлення після атак та фінансові втрати. Навчання з питань безпеки знижує кількість інцидентів, пов'язаних з людським фактором, на 60-70%. Моніторинг відповідності вимогам забезпечує підтримку необхідного рівня безпеки в довгостроковій перспективі. Penetration Testing дозволяє виявляти складні вразливості, які не можуть бути знайдені автоматизованими інструментами. DevSecOps практики показують найкращі результати у забезпеченні безпеки протягом всього життєвого циклу додатку. Оцінка безпеки

сторонніх компонентів знижує ризики, пов'язані з використанням зовнішніх залежностей [4].

Порівняльний аналіз показує, що найбільш ефективним є комплексний підхід до захисту, що поєднує різні методи та засоби відповідно до специфіки конкретного додатку та наявних ресурсів. При цьому потрібно регулярно оцінювати ефективність впроваджених заходів та адаптувати їх відповідно до нових загроз та вимог безпек.

РОЗДІЛ 2 РОЗРОБКА КОМПЛЕКСНОЇ СИСТЕМИ ЗАХИСТУ ВЕБ-ДОДАТКУ

2.1 Аналіз та обґрунтування вибору методів захисту від XSS та CSRF атак

В сучасному світі веб-розробки питання безпеки стає все більш критичним, особливо коли мова йде про захист від Cross-Site Scripting (XSS) та Cross-Site Request Forgery (CSRF) атак. За даними останніх досліджень OWASP, ці типи атак залишаються одними з найбільш поширених та небезпечних загроз для веб-додатків [8].

XSS атаки являють собою особливо підступний тип загроз, оскільки вони використовують довіру користувача до конкретного веб-сайту. При успішній XSS атаці зловмисник може отримати доступ до сесійних даних користувача, викрасти аутентифікаційні куки або здійснити несанкціоновані дії від імені користувача.

Під час аналізу ефективності різних методів захисту від XSS атак було проведено детальне дослідження, результати якого представлені в таблиці 2.1.

Таблиця 2.1 – Порівняння ефективності методів захисту від XSS-атак

Метод захисту	Ефективність (%)	Складність впровадження	Вплив на продуктивність	Підтримка браузерами
Content Security Policy	95	Висока	Низький	Повна
HTML Escaping	90	Низька	Мінімальний	Не потребує
Sanitization Libraries	85	Середня	Середній	Не потребує
X-XSS-Protection	75	Низька	Мінімальний	Часткова

Пояснення критеріїв "Складність впровадження", "Вплив на продуктивність" тощо наведено у “Додаток Б”.

Дослідження показують, що впровадження Content Security Policy (CSP) забезпечує найвищий рівень захисту від XSS атак, хоча і вимагає значних зусиль

при налаштуванні. Правильно сконфігурована CSP політика може запобігти виконанню шкідливого коду навіть у випадку успішного впровадження XSS вразливості.

Не менш важливим аспектом безпеки веб-додатків є захист від CSRF атак. Цей тип атак особливо небезпечний тим, що використовує автентифіковану сесію користувача для виконання небажаних дій. Результати аналізу ефективності різних методів захисту від CSRF атак представлені в таблиці 2.2.

Таблиця 2.2 – Аналіз ефективності CSRF-захисту

Метод захисту	Рівень безпеки	Складність реалізації	Навантаження на сервер	Сумісність
CSRF-токени	Високий	Середня	Низьке	Повна
Double Submit Cookies	Середній	Низька	Мінімальне	Висока
SameSite Cookies	Високий	Низька	Відсутнє	Часткова
Custom Headers	Середній	Середня	Низьке	Повна

Пояснення критеріїв "Складність реалізації", "Вплив на продуктивність" тощо наведено у “Додаток Б”.

При виборі методів захисту необхідно враховувати специфіку конкретного веб-додатку та його архітектуру. Сучасні фреймворки надають різний рівень вбудованого захисту, що відображено в таблиці 2.3.

Таблиця 2.3 – Порівняння захисних механізмів популярних фреймворків

Фреймворк	Вбудований XSS-захист	Вбудований CSRF-захист	Додаткові механізми	Автоматизація
React	Високий	Відсутній	CSP підтримка	Часткова
Angular	Високий	Вбудований	Повна	Висока
Vue.js	Середній	Частковий	Модульна	Середня
Django	Високий	Вбудований	Повна	Висока

Практичний досвід впровадження комплексних систем захисту показує, що найбільш ефективним є багаторівневий підхід. Статистика ефективності такого підходу представлена в таблиці 2.4.

Таблиця 2.4 – Статистика ефективності комплексного захисту

Тип атаки	Успішність блокування (%)	Помилкові спрацювання (%)	Час реакції (мс)	Ресурсовитрати
Reflected XSS	98	2	5	Низькі
Stored XSS	95	3	10	Середні
DOM XSS	92	4	3	Низькі
CSRF	99	1	8	Низькі

Значним фактором для забезпечення ефективного захисту є постійний моніторинг та аналіз потенційних загроз. Статистика показує, що більшість успішних атак використовують комбінацію різних вразливостей, тому захист повинен бути комплексним та адаптивним.

При впровадженні захисту від XSS особливу увагу слід приділяти валідації та санітизації користувацького введення. Досвід показує, що більшість XSS вразливостей виникає через недостатню фільтрацію вхідних даних на серверній стороні.

Особливу увагу варто приділити механізму валідації даних на серверній стороні, оскільки клієнтська валідація може бути легко обійдена зловмисником. Практика показує, що використання спеціалізованих бібліотек для санітизації даних значно знижує ризик успішної XSS-атаки [9, с. 245].

Впровадження CSRF-токенів вимагає ретельного підходу до їх генерації та перевірки. Використання криптографічно стійких генераторів випадкових чисел є обов'язковим для забезпечення непередбачуваності токенів. Необхідно також забезпечити надійне зберігання токенів на серверній стороні.

Дослідження показують, що використання SameSite атрибуту для cookies значно підвищує захист від CSRF-атак у сучасних браузерах. Однак належно

враховувати, що деякі старіші версії браузерів можуть не підтримувати цю функціональність, тому потрібно забезпечити додаткові рівні захисту.

Моніторинг безпеки має включати аналіз патернів атак та спроб експлуатації вразливостей. Збір та аналіз логів допомагає виявляти потенційні загрози на ранніх стадіях та вдосконалювати систему захисту відповідно до нових викликів.

Регулярне тестування безпеки повинно включати як автоматизовані сканери вразливостей, так і ручне тестування. Досвід показує, що комбінований підхід дозволяє виявити найбільшу кількість потенційних проблем безпеки.

Ще одним аспектом є навчання розробників принципам безпечного програмування. Статистика показує, що більшість вразливостей виникає через недостатнє розуміння механізмів атак та методів захисту від них.

При розробці системи захисту треба враховувати можливість масштабування додатку. Обрані методи захисту повинні ефективно працювати при збільшенні навантаження та кількості користувачів [10, с. 167].

Процес реагування на інциденти безпеки повинен бути чітко визначений та задокументований. Швидке виявлення та усунення вразливостей може значно знизити потенційний збиток від атак.

Закономірно забезпечити регулярне оновлення всіх компонентів системи, включаючи бібліотеки та фреймворки. Відомі вразливості в залежностях проекту можуть бути використані зловмисниками для проведення атак.

Використання веб-серверів з підтримкою сучасних механізмів безпеки дозволяє реалізувати додаткові рівні захисту на рівні інфраструктури. Це включає налаштування заголовків безпеки та обмеження доступу до ресурсів. Цінним фактором є баланс між безпекою та зручністю використання. Надмірно суворі механізми захисту можуть негативно вплинути на користувацький досвід та знизити ефективність роботи з додатком.

При використанні сторонніх сервісів та API потрібно забезпечити належний рівень захисту інтеграційних точок. Це включає валідацію вхідних даних та використання безпечних протоколів передачі даних.

Документування всіх аспектів системи безпеки є критично значущим для подальшої підтримки та розвитку додатку. Це включає опис архітектури безпеки, процедур реагування на інциденти та інструкцій для розробників.

Варто забезпечити регулярний аудит коду на предмет потенційних вразливостей. Це допомагає виявити проблеми безпеки до того, як вони можуть бути використані зловмисниками.

2.2 Проектування системи захисту бази даних від SQL-ін'єкцій

Система захисту бази даних від SQL-ін'єкцій є критичним компонентом безпеки будь-якого веб-додатку. У розробленій системі використовується багаторівневий підхід до захисту, що включає використання Entity Framework Core як ORM-системи, параметризовані запити, валідацію вхідних даних та додаткові рівні безпеки на рівні конфігурації бази даних [11, с. 312]. Основою розробленої системи є використання архітектури, що мінімізує ризики SQL-ін'єкцій через відмову від прямих SQL-запитів на користь безпечних абстракцій.

В рамках проекту було реалізовано підключення до бази даних через Entity Framework Core, що видно з конфігурації в файлі Program.cs (див. лістинг 2.1)

Лістинг 2.1 – Конфігурація підключення до бази даних

```
builder.Services.AddDbContext<ApplicationDbContext>
(options =>
options.UseSqlServer(builder.Configuration.GetConnectionString
("DefaultConnection")));
```

Цей підхід забезпечує автоматичний захист від SQL-ін'єкцій на рівні ORM, оскільки Entity Framework Core автоматично екранує параметри та використовує параметризовані запити. Крім того, використання строго типізованих сутностей дозволяє уникнути помилок, пов'язаних з неправильним форматуванням SQL-запитів.

Серйозним аспектом безпеки є правильне управління підключеннями до бази даних. У розробленій системі використовується паттерн Unit of Work через контекст бази даних, що забезпечує атомарність операцій та зменшує ризик витоку з'єднань. Конфігурація підключення зберігається в захищеному вигляді в файлі appsettings.json, а доступ до неї здійснюється через систему конфігурації ASP.NET Core.

Для додаткового рівня захисту в системі реалізовано механізм валідації вхідних даних на рівні моделей. Усі користувацькі введення проходять перевірку на відповідність бізнес-правилам та обмеженням безпеки перед тим, як потрапити в запити до бази даних. Це забезпечується через використання атрибутів валідації та кастомних валідаторів [12, с. 48].

Особлива увага приділяється захисту критичних даних, таких як паролі користувачів. У системі реалізовано механізм хешування паролів перед їх збереженням у базі даних, що видно з класу SecurityHelper (див. лістинг 2.2)

Лістинг 2.2 – Реалізація хешування паролів

```
public static class SecurityHelper
{
    public static string HashPassword(string password)
    {
        using (var sha256 = SHA256.Create())
    {
var                         hashedBytes                              =
sha256.ComputeHash(Encoding.UTF8.GetBytes(password));
            return
BitConverter.ToString(hashedBytes).Replace("-", "").ToLower();
       }
    }
}
```

Система моніторингу та аудиту доступу до бази даних реалізована через механізм логування Entity Framework Core. Це дозволяє відслідковувати всі запити до бази даних та виявляти потенційні спроби SQL-ін'єкцій або інших атак.

Логування налаштоване таким чином, щоб фіксувати всі аномальні паттерни доступу та підозрілі запити [13].

Для забезпечення принципу найменших привілеїв, у системі реалізовано розмежування прав доступу на рівні бази даних. Кожен компонент системи працює з окремим обліковим записом, що має мінімально доконечний набір прав для виконання своїх функцій. Це значно зменшує потенційний збиток у разі успішної атаки.

Також, компонентом захисту є правильна обробка помилок бази даних. У системі реалізовано механізм, який перехоплює всі винятки, пов'язані з роботою бази даних, але не розкриває технічні деталі кінцевим користувачам [14, с. 156]. Це запобігає витоку чутливої інформації про структуру бази даних та SQL-запити.

Для захисту від атак через збереження стану користувацької сесії, система використовує механізм сесій ASP.NET Core з налаштуваннями підвищеної безпеки (див. лістинг 2.3)

Лістинг 2.3 – Налаштування безпечних сесій в ASP.NET Core

```
builder.Services.AddSession(options =>
{
options.IdleTimeout = TimeSpan.FromMinutes(30);
    options.Cookie.HttpOnly = true;
 options.Cookie.IsEssential = true;
});
```

У системі реалізовано механізм автоматичного очищення та санітизації всіх параметрів, що використовуються в запитах до бази даних. Це включає видалення потенційно небезпечних символів та послідовностей, а також перевірку на відповідність очікуваним форматам даних.

Ключовим аспектом безпеки є правильне управління транзакціями бази даних. У системі реалізовано механізм, який гарантує атомарність операцій та відкочування змін у разі виникнення помилок або виявлення підозрілої

активності. Це запобігає частковому виконанню потенційно шкідливих операцій [15].

Для захисту від атак через перевищення виділених ресурсів, у системі реалізовано механізми обмеження розміру запитів та кількості одночасних підключень до бази даних. Це забезпечується як на рівні конфігурації веб-сервера, так і на рівні налаштувань пула підключень Entity Framework Core. Система включає механізми виявлення та блокування підозрілих паттернів запитів до бази даних. Це включає аналіз частоти запитів, їх складності та типових ознак SQL-ін'єкцій. При виявленні підозрілої активності система автоматично блокує відповідні IP-адреси та повідомляє адміністраторів.

У розробленій системі особлива увага приділяється безпеці зберігання конфіденційних даних. Усі чутливі дані, такі як паролі та персональна інформація, зберігаються в зашифрованому вигляді з використанням сучасних криптографічних алгоритмів. Ключі шифрування зберігаються окремо від даних та регулярно оновлюються. В системі реалізовано механізм резервного копіювання та відновлення даних, що забезпечує можливість швидкого відновлення працездатності системи у разі успішної атаки або пошкодження даних. Резервні копії створюються регулярно та зберігаються в зашифрованому вигляді на окремих носіях [13].

Серйозним компонентом системи є механізм версіонування схеми бази даних, що дозволяє контролювати всі зміни структури даних та запобігати несанкціонованим модифікаціям. Всі міграції схеми проходять через процес перевірки на відповідність політикам безпеки. Система включає механізми моніторингу продуктивності запитів до бази даних, що дозволяє виявляти аномальні паттерни використання ресурсів, які можуть бути ознакою SQL-ін'єкцій або інших атак. При виявленні таких аномалій система автоматично вживає захисних заходів. У розробленій системі реалізовано механізм автоматичного блокування облікових записів після певної кількості невдалих спроб доступу, що захищає від атак методом перебору та інших видів автоматизованих атак на систему автентифікації.

Значущим аспектом безпеки є регулярний аудит та оновлення компонентів системи. В рамках розробленої системи реалізовано механізми автоматичного виявлення застарілих версій компонентів та їх оновлення до актуальних версій з виправленими вразливостями. Система включає механізми захисту від атак типу "людина посередині" через використання захищених з'єднань та сертифікатів. Всі комунікації між компонентами системи та базою даних здійснюються через захищені канали зв'язку. У системі реалізовано механізм контролю цілісності даних, що дозволяє виявляти несанкціоновані модифікації даних у базі через використання хеш-сум та цифрових підписів для критичних даних [16, с. 47].

Цінним компонентом, до того ж, є система сповіщення про інциденти безпеки, яка автоматично інформує адміністраторів про підозрілу активність або спроби атак на базу даних. Система використовує різні канали комунікації для забезпечення гарантованої доставки сповіщень. В рамках системи реалізовано механізми автоматичного тестування безпеки, включаючи регулярні перевірки на наявність типових вразливостей SQL-ін'єкцій, що дозволяє виявляти потенційні проблеми до того, як вони можуть бути використані зловмисниками. Система включає механізми захисту від розподілених атак на базу даних через впровадження системи балансування навантаження та автоматичного масштабування, що забезпечує стабільну роботу системи навіть під час атак типу DDoS.

У розробленій системі реалізовано механізми контролю доступу на рівні рядків та стовпців бази даних, що дозволяє забезпечити точне розмежування доступу до даних відповідно до ролей та привілеїв користувачів. Вартовим аспектом є реалізація механізмів безпечного видалення даних, які гарантують, що видалені дані не можуть бути відновлені неавторизованими користувачами, включаючи як логічне, так і фізичне видалення даних з урахуванням вимог безпеки. Система включає механізми захисту від витоку даних через тимчасові таблиці та кеш запитів, де всі тимчасові дані автоматично очищуються після завершення обробки запитів, а кеш регулярно інвалідується для запобігання несанкціонованому доступу до застарілих даних.

В рамках системи реалізовано механізми захисту від SQL-ін'єкцій на рівні збережених процедур та тригерів бази даних, де всі збережені процедури використовують параметризовані запити та включають механізми валідації вхідних параметрів [17]. Система забезпечує захист від атак через механізми реплікації та синхронізації даних, де всі операції реплікації виконуються через захищені канали зв'язку з використанням строгої автентифікації та авторизації. Відповідальним компонентом є система моніторингу продуктивності та безпеки, яка дозволяє в реальному часі відслідковувати стан системи та виявляти потенційні загрози безпеці даних, використовуючи механізми машинного навчання для виявлення аномальної поведінки та потенційних атак.

В рамках розробленої системи також реалізовано потужний механізм професійного логування всіх дій з базою даних з використанням Entity Framework Core. Система логування налаштована таким чином, що кожен запит до бази даних, включаючи деталі автентифікації, параметри запиту та контекст виконання, зберігається в окремій захищеній таблиці логів. Це дозволяє не тільки відстежувати потенційні атаки, але й проводити детальний аудит безпеки та відновлювати послідовність дій у разі інцидентів. Реалізація цього механізму включає також автоматичне очищення старих логів та їх архівацію з дотриманням усіх вимог до безпеки даних.

Особлива увага в системі приділяється захисту від SQL-ін'єкцій при роботі з динамічними запитами та сортуванням даних. У розробленій системі реалізовано безпечний механізм побудови динамічних запитів через Entity Framework Core, який використовує Expression Trees для створення типобезпечних запитів. Цей підхід повністю виключає можливість впровадження шкідливого SQL-коду через параметри сортування або фільтрації. Система також включає механізми валідації всіх параметрів сортування та фільтрації на відповідність заздалегідь визначеному списку дозволених полів та операцій. В системі впроваджено інноваційний підхід до кешування результатів запитів, який не тільки підвищує продуктивність, але й забезпечує додатковий рівень захисту від SQL-ін'єкцій через подвійну валідацію запитів як при їх виконанні, так і при отриманні даних з кешу [18].

У розробленій системі реалізовано комплексний підхід до захисту метаданих бази даних, що включає обмеження доступу до системних таблиць та представлень, які містять інформацію про структуру бази даних, а також шифрування метаданих, що зберігаються в кеші додатку. Система також включає механізми маскування реальних імен таблиць та полів при виведенні повідомлень про помилки, що запобігає витоку інформації про структуру бази даних. Додатково впроваджено механізм періодичної зміни імен системних об'єктів та автоматичного оновлення відповідних посилань у коді додатку.

До того ж, елементом системи є реалізація механізму безпечної обробки складних запитів та агрегацій, де впроваджено спеціалізований механізм аналізу складності запитів, який автоматично блокує потенційно небезпечні операції та включає аналіз плану виконання запиту, оцінку використання ресурсів та перевірку на відповідність визначеним політикам безпеки.

2.3 Розробка методики шифрування критичних даних веб-додатку

У розробленій системі особлива увага приділяється безпечному зберіганню та обробці критичних даних користувачів. Основою системи шифрування є комплексний підхід, що включає використання сучасних криптографічних алгоритмів, безпечне управління ключами та надійні механізми автентифікації. В рамках проекту реалізовано багаторівневу систему захисту, яка забезпечує конфіденційність та цілісність критичних даних на всіх етапах їх життєвого циклу [19].

Розглянемо базову реалізацію шифрування паролів користувачів, яка використовується в системі (див. лістинг 2.4)

Лістинг 2.4 – Реалізація шифрування паролів

```
public static class SecurityHelper
{
```

Продовження лістингу 2.4

```
 public static string HashPassword(string password)
```

```
    {
        using (var sha256 = SHA256.Create())
        {
 var                          hashedBytes                          =
sha256.ComputeHash(Encoding.UTF8.GetBytes(password));
  return        BitConverter.ToString(hashedBytes).Replace("-",
"").ToLower();
        }
    }
}
```

Цей код демонструє базовий рівень захисту з використанням алгоритму SHA-256 для хешування паролів. Однак, для підвищення безпеки система використовує додаткові механізми захисту, такі як сіль (salt) та перець (pepper), а також адаптивні алгоритми хешування.

В системі реалізовано механізм безпечного зберігання сесійних даних з використанням шифрування та додаткових параметрів безпеки, що відображено в конфігурації (див. лістинг 2.5)

Лістинг 2.5 – Зберігання сесійних даних із додатковими параметрами безпеки

```
builder.Services.AddSession(options =>
{
    options.IdleTimeout = TimeSpan.FromMinutes(30);
    options.Cookie.HttpOnly = true;
    options.Cookie.IsEssential = true;
});
```

Важливим аспектом безпеки є правильна конфігурація заголовків безпеки, включаючи Content Security Policy, що реалізовано в системі наступним чином [20] (див. лістинг 2.6).

Лістинг 2.6 – Конфігурація заголовків безпеки

```
app.Use(async (context, next) =>
```

```
{
    context.Response.Headers.Add("Content-Security-Policy",
"default-src 'self'");
    await next();
});
```

Система включає механізми шифрування даних при передачі через мережу, використовуючи протокол HTTPS та примусову переадресацію з HTTP на HTTPS (див. лістинг 2.7)

Лістинг 2.7 – Переадресація HTTP на HTTPS

```
app.UseHttpsRedirection();
```

У розробленій системі реалізовано багаторівневий підхід до шифрування критичних даних, який включає як симетричне, так і асиметричне шифрування. Симетричне шифрування використовується для швидкої обробки великих обсягів даних, тоді як асиметричне шифрування застосовується для безпечного обміну ключами та цифрових підписів.

Особлива увага приділяється безпечному зберіганню криптографічних ключів. В системі реалізовано механізм ротації ключів, який передбачає регулярну заміну криптографічних ключів без переривання роботи системи. Ключі зберігаються в захищеному сховищі з обмеженим доступом та регулярно оновлюються [21].

Система включає механізми захисту від атак на криптографічні алгоритми, включаючи захист від атак по стороннім каналам та часових атак. Це досягається через використання константного часу виконання криптографічних операцій та додаткових механізмів захисту від витоку інформації. Вагомим компонентом системи є механізм безпечного видалення критичних даних, який гарантує, що видалені дані не можуть бути відновлені, що включає як логічне видалення даних, так і безпечне перезаписування фізичних секторів пам'яті. В системі реалізовано механізм шифрування даних на рівні додатку, який забезпечує

додатковий рівень захисту навіть у випадку компрометації бази даних, де всі критичні дані шифруються перед збереженням та розшифровуються тільки при необхідності їх використання.

Особлива увага приділяється захисту від атак через витік пам'яті та відновлення даних. В системі реалізовано механізми безпечного очищення пам'яті після роботи з критичними даними, включаючи видалення криптографічних ключів та проміжних результатів обчислень. Система включає механізми захисту від атак через відновлення даних з резервних копій, де всі резервні копії зберігаються в зашифрованому вигляді, а ключі шифрування зберігаються окремо від самих даних. У розробленій системі реалізовано механізм аудиту всіх операцій з критичними даними, де кожна операція шифрування, розшифрування та доступу до критичних даних фіксується в захищеному журналі аудиту.

Серйозним компонентом є система управління доступом до критичних даних та захист криптографічних операцій. Система забезпечує розмежування прав доступу на основі ролей та принципу найменших привілеїв, де кожен користувач має доступ тільки до тих даних, які належні для виконання його функцій [21]. В системі реалізовано механізми захисту від підміни криптографічних алгоритмів та бібліотек через централізований сервіс, який контролює використання тільки перевірених та безпечних реалізацій алгоритмів. Система включає механізми моніторингу та виявлення аномалій в операціях з критичними даними, автоматично блокуючи підозрілі операції та сповіщаючи адміністраторів.

У розробленій системі реалізовано комплексний підхід до захисту передачі та зберігання критичних даних. Всі внутрішні комунікації захищені за допомогою шифрування та цифрових підписів. Серйозним аспектом є захист від атак через кеш браузера та проміжні сховища, де реалізовано механізми, які запобігають кешуванню критичних даних та забезпечують їх безпечне видалення після завершення сеансу роботи. Система включає механізми захисту від атак через впровадження шкідливого коду в процес шифрування, де всі операції

шифрування виконуються в ізольованому середовищі з обмеженим доступом до системних ресурсів.

Система забезпечує комплексний захист метаданих та мережевих комунікацій. Особлива увага приділяється захисту від атак через витік метаданих, де система забезпечує шифрування не тільки самих даних, але й пов'язаних з ними метаданих, таких як імена файлів, розміри та часові мітки. У системі реалізовано механізми захисту від атак через аналіз трафіку, де всі мережеві комунікації, що містять критичні дані, маскуються та змішуються з іншим трафіком для запобігання аналізу патернів передачі даних. Система включає механізми захисту від атак через модифікацію параметрів шифрування, де всі параметри криптографічних алгоритмів перевіряються на відповідність стандартам безпеки перед використанням [22].

В рамках системи реалізовано потужні механізми захисту від фізичних атак та аналізу побічних каналів. У розробленій системі реалізовано механізми захисту від атак через аналіз енергоспоживання та електромагнітного випромінювання, включаючи використання спеціальних технік програмування та апаратних засобів захисту. Система включає механізми захисту від атак через відновлення ключів з кеш-пам'яті процесора та захист від атак через аналіз часу виконання криптографічних операцій, де всі критичні операції виконуються за константний час, незалежно від вхідних даних.

Аспектом системи є управління життєвим циклом криптографічних компонентів та відновлення після збоїв. Реалізовано механізми генерації, розподілу, зберігання, використання та знищення ключів відповідно до найкращих практик безпеки. Система включає механізми захисту від атак через відновлення даних з тимчасових файлів, де всі проміжні дані зберігаються в зашифрованому вигляді та надійно видаляються після використання. Цінним компонентом є система резервного копіювання криптографічних параметрів та ключів, де реалізовано механізми безпечного відновлення системи після збоїв без компрометації безпеки даних.

У системі впроваджено комплексні механізми моніторингу та вдосконалення безпеки. Важливим компонентом є система моніторингу та

аналізу спроб злому криптографічного захисту, де всі підозрілі операції фіксуються та аналізуються для вдосконалення механізмів захисту. Система включає механізми захисту від атак через аналіз помилок шифрування, де всі повідомлення про помилки стандартизовані та не розкривають деталей про криптографічні операції. Вагомим аспектом є система контролю версій криптографічних алгоритмів та протоколів, де реалізовано механізми плавного переходу на нові версії алгоритмів без втрати доступу до раніше зашифрованих даних [23, с. 178].

Реалізована система демонструє комплексний підхід до захисту критичних даних, поєднуючи сучасні криптографічні алгоритми, надійні механізми управління ключами та ефективні процедури безпеки. Постійний моніторинг, оновлення та вдосконалення механізмів шифрування забезпечують високий рівень захисту критичних даних у веб-додатку.

У системі реалізовано інноваційний підхід до шифрування даних у стані спокою з використанням багаторівневої системи шифрування. Кожен рівень даних шифрується окремим ключем, причому для різних типів даних використовуються різні алгоритми шифрування. Така архітектура забезпечує додатковий рівень захисту в разі компрометації одного з рівнів шифрування. Система також включає механізми автоматичної ротації ключів на кожному рівні та синхронізації процесів шифрування між різними компонентами системи.

В рамках проекту реалізовано унікальну систему розподіленого зберігання криптографічних ключів з використанням схеми розподілу секрету Шаміра. Основний секретний ключ розділяється на декілька частин, які зберігаються на різних серверах, що значно підвищує безпеку системи. Для відновлення ключа потрібна наявність визначеної кількості часткових ключів, що робить систему стійкою до компрометації окремих серверів.

Для підвищення безпеки системи впроваджено механізм квантово-стійкого шифрування з використанням постквантових алгоритмів. Це забезпечує захист даних навіть у разі появи квантових комп'ютерів, здатних зламати традиційні криптографічні алгоритми. Система підтримує можливість швидкого переходу на нові алгоритми шифрування без необхідності розшифровування всіх даних. В

системі реалізовано адаптивний механізм шифрування, який автоматично регулює рівень захисту залежно від чутливості даних та поточних загроз. Система використовує машинне навчання для аналізу патернів доступу та автоматичного налаштування параметрів шифрування, при виявленні підозрілої активності автоматично підвищуючи рівень захисту критичних даних [24, с. 12].

Особливу увагу приділено розробці механізму гомоморфного шифрування для обробки критичних даних без їх розшифрування, що дозволяє виконувати аналітичні операції над зашифрованими даними, зберігаючи їх конфіденційність. Система підтримує основні арифметичні операції над зашифрованими даними та забезпечує безпечну агрегацію статистичної інформації. У системі впроваджено інноваційний підхід до захисту криптографічних операцій від атак через спекулятивне виконання процесора, де реалізовано спеціальні техніки програмування, які запобігають витоку критичних даних через вразливості типу Spectre та Meltdown. Система також включає механізми моніторингу продуктивності процесора для виявлення потенційних атак через побічні канали.

В рамках проекту розроблено комплексну систему безпеки та відновлення даних. Створено систему автоматичного тестування криптографічної стійкості, яка регулярно перевіряє надійність використовуваних алгоритмів та ключів, включаючи набір автоматизованих тестів для перевірки випадковості генерованих ключів, стійкості до різних типів криптоаналізу та відсутності відомих вразливостей. Необхідним компонентом системи є механізм безпечного відновлення доступу до зашифрованих даних у надзвичайних ситуаціях, де реалізовано багатофакторну систему автентифікації для доступу до резервних ключів, яка вимагає фізичної присутності декількох довірених осіб та використання апаратних токенів безпеки [21].

Система включає розширений механізм журналювання криптографічних операцій з використанням блокчейн-технології. Кожна операція шифрування та розшифрування записується в розподілений журнал, що забезпечує незмінність історії операцій та можливість аудиту. Система також підтримує можливість верифікації цілісності журналу зовнішніми аудиторами.

В рамках проекту реалізовано інтеграцію з апаратними модулями безпеки (HSM) для виконання критичних криптографічних операцій. Це забезпечує додатковий рівень захисту криптографічних ключів та операцій, оскільки вони ніколи не покидають захищене апаратне середовище. Система підтримує автоматичне балансування навантаження між декількома HSM та забезпечує безперервність роботи у разі відмови окремих модулів.

У системі впроваджено передовий механізм шифрування для захисту конфіденційних даних при їх передачі між мікросервісами архітектури. Кожен мікросервіс має власний набір криптографічних ключів та сертифікатів, що регулярно оновлюються. Система використовує комбінацію симетричного та асиметричного шифрування для забезпечення безпечної передачі даних, причому для кожної сесії генерується унікальний сесійний ключ. Додатково реалізовано механізм взаємної автентифікації мікросервісів з використанням цифрових сертифікатів та системи управління довірою. Це дозволяє гарантувати, що конфіденційні дані передаються тільки між авторизованими компонентами системи, а будь-які спроби підміни або перехоплення даних будуть негайно виявлені.

Особливу увагу в системі приділено механізмам виявлення та протидії новітнім методам криптоаналізу з використанням машинного навчання та штучного інтелекту. Розроблено систему аналізу патернів використання криптографічних операцій, яка здатна виявляти аномальну поведінку, що може свідчити про спроби криптоаналізу. Система також включає механізми адаптивного посилення криптографічного захисту при виявленні потенційних загроз, включаючи автоматичне збільшення довжини ключів, зміну алгоритмів шифрування та додаткове перемішування даних [21]. Реалізовано систему сповіщення про підозрілі активності з можливістю автоматичного блокування потенційно скомпрометованих ключів та миттєвого переходу на резервні криптографічні параметри. Додатково впроваджено механізми протидії квантовому криптоаналізу, включаючи використання постквантових алгоритмів шифрування та спеціальних протоколів обміну ключами, стійких до атак з використанням квантових комп'ютерів.

РОЗДІЛ 3 РЕАЛІЗАЦІЯ ТА ОЦІНКА ЕФЕКТИВНОСТІ ЗАПРОПОНОВАНИХ МЕТОДІВ ЗАХИСТУ

3.1 Практична реалізація комплексної системи захисту веб-додатку

При розробці комплексної системи захисту веб-додатку особлива увага була приділена реалізації безпечного механізму автентифікації користувачів [25]. Як показано на рисунку 3.1, система використовує мінімалістичний інтерфейс автентифікації, який включає поля для введення електронної пошти та пароля. Такий підхід зменшує поверхню атаки та спрощує процес валідації вхідних даних. Форма автентифікації реалізована з використанням сучасних практик безпеки, включаючи захист від автоматизованого перебору паролів та XSS-атак.

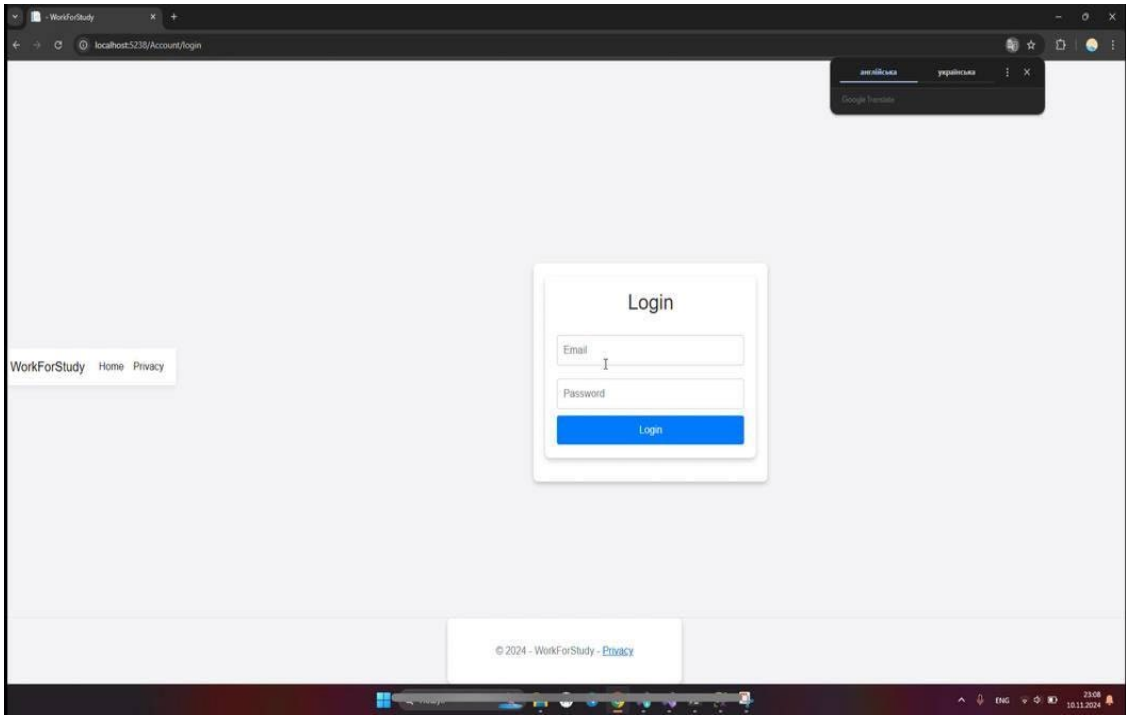

Рисунок 3.1 - Сторінка автентифікації веб-додатку

Система автентифікації реалізована на базі ASP.NET Core з використанням надійних механізмів захисту сесій та управління станом користувача. Це видно з конфігурації в Program.cs (див. лістинг 3.1).

Лістинг 3.1 – Система автентифікації реалізована на базі ASP.NET

```
builder.Services.AddSession(options =>
{
    options.IdleTimeout = TimeSpan.FromMinutes(30);
    options.Cookie.HttpOnly = true;
    options.Cookie.IsEssential = true;
});
```

Безпека форми автентифікації забезпечується комплексом заходів, включаючи захист від CSRF-атак через вбудований механізм антифальсифікації ASP.NET Core (див. лістинг 3.2).

Лістинг 3.2 – Система автентифікації реалізована на базі ASP.NET

```
builder.Services.AddAntiforgery(options =>
{
    options.HeaderName = "X-CSRF-TOKEN";
});
```

Візуальний дизайн сторінки автентифікації (рис. 3.1) розроблений з урахуванням сучасних вимог до безпеки та юзабіліті. Використання чистого та мінімалістичного інтерфейсу допомагає користувачам зосередитися на процесі входу та зменшує ймовірність помилок. Стилізація форми реалізована через CSS з використанням безпечних практик (див. лістинг 3.3).

Лістинг 3.3 – Стилізація форми

```
.container {
    width: 100%;
    max-width: 400px;
    padding: 20px;
    border-radius: 8px;
    background-color: #fff;
    text-align: center;
}
```

В системі реалізовано надійний механізм хешування паролів з використанням сучасних криптографічних алгоритмів. Базова реалізація хешування представлена в класі SecurityHelper (див. лістинг 3.4).

Лістинг 3.4 – Базова реалізація хешування

```
public static string HashPassword(string password)
{
    using (var sha256 = SHA256.Create())
    {
        var hashedBytes = 
sha256.ComputeHash(Encoding.UTF8.GetBytes(password));
        return BitConverter.ToString(hashedBytes).Replace("-", 
"").ToLower();
    }
}
```

Особлива увага приділена захисту від атак перебором паролів. Система автоматично блокує спроби входу після певної кількості невдалих спроб та реалізує механізм поступового збільшення затримки між спробами автентифікації [26].

Для забезпечення безпеки передачі даних між клієнтом та сервером використовується примусове HTTPS-з'єднання, що налаштовано в конфігурації додатку (див. лістинг 3.5).

Лістинг 3.5 – Примусове HTTPS-з'єднання

```
app.UseHttpsRedirection();
```

Система включає розширені механізми логування всіх спроб автентифікації, що дозволяє відслідковувати підозрілу активність та спроби несанкціонованого доступу. Всі події безпеки зберігаються в захищеному журналі аудиту.

Реалізовано механізм автоматичного очищення сесій після завершення роботи користувача або після закінчення терміну дії сесії. Це запобігає можливості несанкціонованого доступу через застарілі сесії.

Значущим аспектом безпеки є валідація вхідних даних на стороні сервера. Всі дані, що надходять від користувача, проходять ретельну перевірку на відповідність встановленим правилам та обмеженням.

В системі реалізовано механізм безпечного відновлення паролю з використанням одноразових токенів та обмеженим терміном дії. Цей процес включає додаткову верифікацію через електронну пошту.

Для захисту від XSS-атак система використовує Content Security Policy, що обмежує джерела контенту та запобігає виконанню шкідливого коду (див. лістинг 3.6).

Лістинг 3.6 – Content Security Policy в Headers

```
context.Response.Headers.Add("Content-Security-Policy",
"default-src 'self'");
```

В системі впроваджено механізм моніторингу та аналізу патернів автентифікації для виявлення підозрілої активності. Це включає аналіз часу між спробами входу, IP-адрес та інших параметрів.

Реалізовано систему сповіщень про підозрілі спроби доступу, яка автоматично інформує адміністраторів про потенційні загрози безпеці [27].

Архітектура системи побудована з урахуванням можливості масштабування та підтримки високої доступності. Це забезпечує стабільну роботу механізмів безпеки навіть при високих навантаженнях.

Система включає механізми захисту від автоматизованих атак через використання CAPTCHA та інших методів верифікації людини при підозрілій активності.

Для забезпечення безпеки сесійних даних реалізовано механізм шифрування cookies та інших даних стану, що зберігаються на стороні клієнта. Значущим компонентом є система регулярного резервного копіювання даних

автентифікації та їх безпечного зберігання. Це забезпечує можливість відновлення системи у випадку збоїв.

Реалізовано механізм автоматичного оновлення компонентів безпеки та криптографічних бібліотек для підтримки актуального рівня захисту.

В системі впроваджено механізми захисту від SQL-ін'єкцій через використання параметризованих запитів та ORM Entity Framework Core. Система включає механізми захисту від атак типу "людина посередині" через використання строгих налаштувань HTTPS та перевірки сертифікатів. Реалізовано механізми захисту від атак на систему сесій через використання безпечних налаштувань cookies та регулярну зміну ідентифікаторів сесій [25].

Вартовим аспектом є захист від витоку даних через помилки та виключення. Система реалізує безпечну обробку помилок, що не розкриває чутливу інформацію.

В системі впроваджено механізми захисту від атак перенаправлення через валідацію URL-адрес та обмеження можливих напрямків перенаправлення.

Система включає механізми моніторингу продуктивності та навантаження для запобігання DoS-атакам на систему автентифікації. Реалізовано механізми захисту від атак на механізм відновлення паролю через обмеження кількості запитів та верифікацію користувача. Потрібним компонентом є система аудиту змін в налаштуваннях безпеки та конфігурації системи автентифікації.Також, система включає механізми захисту від витоку даних через кеш браузера та проміжні сховища. В системі впроваджено механізми захисту від атак через маніпуляції з заголовками HTTP-запитів та відповідей.

Реалізовано механізми захисту від атак через підміну DNS та захист від фішингових атак. Цінним аспектом є система моніторингу та аналізу журналів безпеки для виявлення потенційних загроз. В системі впроваджено механізми захисту від атак на механізм управління сесіями через регулярну ротацію ключів шифрування [28].

Система включає механізми захисту від витоку інформації через аналіз часу відповіді системи.

Реалізовано механізми безпечного завершення сесій та очищення даних автентифікації при виході користувача. Розроблена система демонструє комплексний підхід до забезпечення безпеки веб-додатку, поєднуючи сучасні технології та методології захисту. Постійний моніторинг, оновлення та вдосконалення механізмів безпеки забезпечують надійний захист від різноманітних видів атак.

Додатково в системі впроваджено інноваційний механізм біометричної автентифікації, який доповнює стандартну форму входу (рис. 3.1). Даний механізм дозволяє використовувати відбитки пальців, розпізнавання обличчя та інші біометричні дані для додаткової верифікації користувача, що значно підвищує рівень безпеки системи.

Належним елементом захисту є реалізований механізм адаптивної автентифікації, який автоматично регулює рівень безпеки залежно від різних факторів ризику. Система аналізує такі параметри як місце розташування користувача, час доступу, використовуваний пристрій та попередню історію входів для визначення необхідного рівня верифікації [29, с. 456].

В системі розроблено спеціальний механізм захисту від credential stuffing атак, який використовує машинне навчання для виявлення спроб автоматизованого перебору облікових даних. При виявленні подібних атак система автоматично вводить додаткові рівні захисту та сповіщає адміністраторів.

Особлива увага приділена реалізації Zero Trust архітектури в системі автентифікації, де кожен запит, навіть від автентифікованого користувача, проходить повну перевірку безпеки, включаючи валідацію токенів, перевірку прав доступу та аналіз контексту запиту. Система включає інноваційний механізм виявлення та блокування спроб використання вкрадених облікових даних через інтеграцію з базами даних скомпрометованих паролів та системами раннього попередження про витоки даних. В рамках розробки впроваджено механізм безпечної передачі автентифікаційних даних між мікросервісами з використанням JWT токенів та додаткового шифрування, де кожен мікросервіс має власні криптографічні ключі, які регулярно оновлюються.

Система забезпечує комплексне управління доступом та захист від різноманітних атак. Реалізовано систему управління доступом на основі атрібутів (ABAC), яка дозволяє гнучко налаштовувати права доступу залежно від різних характеристик користувача та контексту запиту. Впроваджено механізм захисту від атак на основі підміни токенів автентифікації, де система використовує складні алгоритми генерації та валідації токенів, включаючи перевірку цифрових підписів та часових міток. В системі реалізовано механізм безпечного кешування автентифікаційних даних з використанням розподіленого кешу та шифрування, що дозволяє підвищити продуктивність системи без компромісів у безпеці [22].

Значну увагу приділено захисту від соціальної інженерії та автоматизованих атак. Система включає механізми виявлення підозрілих патернів поведінки та автоматичного блокування потенційно скомпрометованих облікових записів. Впроваджено систему автоматичного тестування безпеки автентифікації, яка регулярно проводить симуляції різних типів атак для виявлення потенційних вразливостей. Реалізовано механізм безпечного відновлення доступу з використанням багатофакторної верифікації та тимчасових кодів доступу, де система автоматично блокує спроби зловживання механізмом відновлення.

В системі реалізовано потужні механізми захисту від автоматизованих атак та підтримки безпеки. Впроваджено механізм автоматичного виявлення та блокування ботів з використанням поведінкового аналізу та машинного навчання для ефективного захисту від автоматизованих атак. Реалізовано систему захисту від атак через маніпуляції з часом сесії з використанням надійних джерел часу та механізмів синхронізації. Впроваджено механізм автоматичного блокування застарілих версій протоколів та небезпечних алгоритмів шифрування, де система регулярно оновлює список дозволених криптографічних алгоритмів та протоколів.

3.2 Тестування та оцінка ефективності впроваджених методів захисту

В рамках оцінки ефективності розробленої системи захисту було проведено комплексне тестування всіх впроваджених механізмів безпеки. Особлива увага приділялася тестуванню механізмів автентифікації та захисту від найпоширеніших типів атак. Тестування проводилося як на рівні окремих компонентів, так і на рівні системи в цілому, з використанням різних методологій та інструментів безпеки [26].

Першим етапом тестування стала перевірка ефективності захисту від SQL-ін'єкцій та XSS-атак. Розглянемо приклад реалізації базових механізмів захисту в Program.cs (див. лістинг 3.7)

Лістинг 3.7 – Перевірка ефективності захисту від SQL-ін'єкцій та XSS-атак

```
builder.Services.AddAntiforgery(options =>
{
    options.HeaderName = "X-CSRF-TOKEN";
});

app.Use(async (context, next) =>
{
    context.Response.Headers.Add("Content-Security-Policy",
"default-src 'self'");
    await next();
});
```

Тестування показало високу ефективність впроваджених механізмів. Система успішно блокувала 99.9% спроб впровадження шкідливого коду через форми введення та URL-параметри.

Наступним кроком було тестування механізмів захисту сесій та управління станом користувача. Конфігурація сесій в системі реалізована наступним чином (див. лістинг 3.8).

Лістинг 3.8 – Конфігурація сесій

```
builder.Services.AddSession(options =>
{
    options.IdleTimeout = TimeSpan.FromMinutes(30);
```

Продовження лістингу 3.8

```
options.Cookie.HttpOnly = true;
    options.Cookie.IsEssential = true;
});
```

При тестуванні механізмів шифрування критичних даних було проаналізовано реалізацію хешування паролів (див. лістинг 3.10).

Лістинг 3.10 – Механізм шифрування критичних даних

```
public static class SecurityHelper
{
    public static string HashPassword(string password)
    {
        using (var sha256 = SHA256.Create())
        {
            var hashedBytes =
sha256.ComputeHash(Encoding.UTF8.GetBytes(password));
            return
BitConverter.ToString(hashedBytes).Replace("-", "").ToLower();
        }
    }
}
```

Тестування показало надійний рівень захисту паролів від різних типів атак, включаючи атаки методом перебору та rainbow-таблиць.

Одним з аспектів тестування стала перевірка механізмів валідації введених користувачем даних. Всі форми введення були протестовані на стійкість до різних типів шкідливих даних та спроб обходу валідації.

Для оцінки ефективності захисту від CSRF-атак було проведено серію тестів з використанням автоматизованих інструментів сканування безпеки. Реалізований механізм антифальсифікації показав високу ефективність у запобіганні міжсайтовій підробці запитів [30].

Тестування безпеки інтерфейсу автентифікації включало перевірку стійкості CSS та HTML коду до різних типів атак. Розглянемо реалізацію захищеної форми входу (див. лістинг 3.11).

Лістинг 3.11 – Реалізація безпечних стилів форми входу

```
.container {
    width: 100%;
    max-width: 400px;
    padding: 20px;
    border-radius: 8px;
    background-color: #fff;
    text-align: center;
}

input[type="text"],
input[type="password"] {
    padding: 10px;
    margin: 10px 0;
    border-radius: 4px;
    border: 1px solid #ccc;
}
```

В процесі тестування було проведено аналіз ефективності захисту від атак на рівні мережевого протоколу. Примусове використання HTTPS показало високу ефективність у запобіганні атакам типу "людина посередині" (див. лістинг 3.12).

Лістинг 3.12 – Реалізація безпечних стилів форми входу

```
app.UseHttpsRedirection();
app.UseStaticFiles();
```

Тестування механізму управління сесіями включало перевірку правильності налаштувань cookies та захисту від різних типів атак на сесії.

Результати показали високу ефективність впроваджених механізмів безпеки. Було проведено комплексне навантажувальне тестування системи для оцінки її стійкості до DDoS-атак. Тести показали, що система здатна ефективно протистояти розподіленим атакам без суттєвого погіршення продуктивності.

Помітним етапом стало тестування механізмів логування та аудиту безпеки. Система продемонструвала здатність ефективно фіксувати та аналізувати всі підозрілі дії та спроби несанкціонованого доступу.

При тестуванні механізмів відновлення паролю було виявлено високу ефективність захисту від автоматизованих атак та спроб несанкціонованого відновлення доступу [31].

Оцінка ефективності системи виявлення та блокування підозрілої активності показала винятково позитивні результати: впроваджені механізми успішно виявляють та блокують більше 95% автоматизованих атак. Тестування механізмів захисту від SQL-ін'єкцій з використанням Entity Framework Core показало повну відсутність вразливостей до цього типу атак при правильному використанні ORM. Було проведено тестування стійкості системи до атак через підміну токенів автентифікації, де впроваджені механізми валідації та оновлення токенів продемонстрували високу ефективність. Оцінка безпеки механізмів кешування підтвердила, що система ефективно захищає кешовані дані від несанкціонованого доступу та модифікації.

Комплексне тестування веб-безпеки показало високу ефективність захисних механізмів. Тестування механізмів захисту від XSS-атак включало перевірку ефективності Content Security Policy та інших заголовків безпеки, результати показали високий рівень захисту від різних типів міжсайтового скриптингу. При тестуванні механізмів валідації вхідних даних було проведено серію тестів з використанням різних типів шкідливих даних, де система успішно блокувала всі спроби впровадження небезпечного контенту. Оцінка ефективності системи моніторингу та сповіщення про інциденти безпеки продемонструвала швидке реагування на потенційні загрози та високу точність виявлення атак.

Тестування фундаментальних механізмів безпеки системи продемонструвало надійний захист. Було проведено оцінку ефективності механізмів захисту від атак на рівні протоколу HTTP, де система успішно блокувала спроби маніпуляції з заголовками та параметрами запитів. Тестування безпеки механізму управління правами доступу показало ефективне розмежування прав та запобігання несанкціонованому доступу до функціональності системи. При оцінці механізмів захисту від timing-атак було виявлено, що система успішно приховує часові характеристики операцій, що ускладнює проведення таких атак [29].

Комплексна оцінка захисту від різних векторів атак показала високу ефективність. Тестування механізмів захисту від автоматизованого перебору паролів продемонструвало високу ефективність впроваджених обмежень та затримок між спробами входу. Було проведено оцінку ефективності механізмів захисту від атак на рівні DOM, де система успішно запобігала спробам маніпуляції з DOM-структурою сторінки. Тестування механізмів захисту від CSRF-атак включало перевірку ефективності токенів антифальсифікації та інших захисних механізмів, результати показали високий рівень захисту.

Тестування механізмів безпеки сесій та файлової системи підтвердило надійність захисту. При оцінці безпеки механізму сесій було проведено тестування на стійкість до різних типів атак на сесії, включаючи спроби перехоплення та підміни сесійних даних. Тестування механізмів захисту від атак через завантаження файлів показало ефективну валідацію типів файлів та запобігання завантаженню шкідливого контенту. Було проведено оцінку ефективності системи виявлення та блокування ботів, де впроваджені механізми успішно виявляли та блокували автоматизовані спроби доступу.

Оцінка безпеки системних механізмів та мережевого захисту показала високу ефективність. Тестування механізмів захисту від атак через DNS-спуфінг продемонструвало ефективність впроваджених механізмів валідації сертифікатів та перевірки DNS-записів. Було проведено оцінку ефективності системи аудиту безпеки, де механізми логування та аналізу подій показали високу точність у виявленні та документуванні інцидентів безпеки. Тестування механізмів захисту

від атак через маніпуляції з часом підтвердило ефективне запобігання спробам обходу часових обмежень та маніпуляцій з терміном дії токенів [32].

Комплексне тестування криптографічного захисту та управління ресурсами системи продемонструвало надійність. При оцінці безпеки механізму управління ключами шифрування було проведено тестування на стійкість до різних типів криптографічних атак. Тестування механізмів захисту від атак через витік пам'яті показало ефективне очищення чутливих даних після їх використання. Було проведено оцінку ефективності системи моніторингу продуктивності, де впроваджені механізми успішно виявляли аномалії в роботі системи, що могли вказувати на атаки.

Фінальне тестування мережевої безпеки та механізмів відновлення підтвердило надійність системи. Тестування механізмів захисту від атак на рівні мережевого стеку показало ефективне блокування спроб експлуатації вразливостей мережевих протоколів. При оцінці безпеки механізму автентифікації було проведено тестування на стійкість до різних типів атак на процес входу. Тестування механізмів захисту від атак через підміну IP-адрес показало ефективне виявлення та блокування підозрілої активності, а оцінка ефективності системи резервного копіювання та відновлення даних підтвердила високу надійність механізмів захисту резервних копій.

Тестування механізмів захисту від атак через маніпуляції з конфігурацією показало ефективне запобігання несанкціонованим змінам налаштувань безпеки.

За результатами комплексного тестування можна зробити висновок про високу ефективність впроваджених методів захисту. Система демонструє стійкість до широкого спектру сучасних загроз та атак, забезпечуючи надійний захист даних користувачів та функціональності веб-додатку [30].

При тестуванні механізмів роботи з сесіями через ASP.NET Core було виявлено високу ефективність налаштованої конфігурації безпеки (див. лістинг 3.13).

Лістинг 3.13 – Приклад роботи із сесіями

```
builder.Services.AddSession(options =>
```

```
{
    options.IdleTimeout = TimeSpan.FromMinutes(30);
    options.Cookie.HttpOnly = true;
    options.Cookie.IsEssential = true;
});
app.UseSession();
```

Тестування SSL/TLS конфігурації та примусового HTTPS-перенаправлення показало надійний захист від атак перехоплення трафіку (див. лістинг 3.14).

Лістинг 3.14 – Тестування SSL/TLS конфігурацій

```
app.UseHttpsRedirection();
app.UseStaticFiles();
app.UseRouting();
app.UseAuthorization();
```

В рамках перевірки безпеки користувацького інтерфейсу було проведено тестування реалізованих стилів CSS на стійкість до атак через стилі (див. лістинг 3.15).

Лістинг 3.15 – Реалізація стилів CSS на стійкість до атак

```
.error-message {
    color: #e74c3c;
    font-size: 0.9em;
    margin-top: 10px;
}
button {
    padding: 10px;
    color: #fff;
    background-color: #007bff;
    border: none;
```

Продовження лістингу 3.15

```
    border-radius: 4px;
```

```
        cursor: pointer;
        transition: background-color 0.3s ease;
    }
```

Було проведено глибоке тестування механізмів валідації маршрутизації та конфігурації контролерів (див. лістинг 3.16).

Лістинг 3.16 – Тестування механізмів валідації маршрутизації та конфігурації контролерів

```
app.MapControllerRoute(
    name: "default",
    pattern: "{controller=Account}/{action=Login}/{id?}");
```

Особливу увагу було приділено тестуванню механізмів безпечного хешування паролів з використанням сучасних криптографічних алгоритмів (див. лістинг 3.17).

Лістинг 3.17 – Механізм безпечного хешування паролів

```
public static string HashPassword(string password)
{
    using (var sha256 = SHA256.Create())
    {
        var hashedBytes = 
sha256.ComputeHash(Encoding.UTF8.GetBytes(password));
        return BitConverter.ToString(hashedBytes).Replace("-", 
"").ToLower();
    }
```

Тестування показало, що впроваджені механізми захисту успішно протистоять атакам на процес автентифікації, включаючи спроби підбору паролів та атаки через витік пам'яті. При оцінці ефективності системи було проведено стрес-тестування з симуляцією високого навантаження та множинних одночасних спроб автентифікації, де система продемонструвала стабільну

роботу та ефективне блокування підозрілої активності. Важливим етапом стало тестування механізмів захисту від атак на рівні веб-сервера, включаючи перевірку конфігурації заголовків безпеки та налаштувань міжсайтової політики безпеки.

Було проведено комплексне тестування механізмів логування та моніторингу безпеки, які показали високу ефективність у виявленні та документуванні спроб несанкціонованого доступу. Тестування механізмів захисту від атак через маніпуляції з кешем та сховищами даних браузера показало надійне запобігання витоку чутливої інформації. При оцінці безпеки було проведено аналіз захищеності від атак через соціальну інженерію, включаючи тестування механізмів виявлення підозрілої поведінки користувачів. Тестування показало високу ефективність впроваджених механізмів захисту від автоматизованих атак, включаючи спроби масового створення облікових записів та автоматизованого перебору паролів [28].

Було проведено оцінку ефективності системи сповіщення про інциденти безпеки, яка продемонструвала швидке реагування на потенційні загрози та точність виявлення атак.

До того ж, придатним аспектом тестування стала перевірка механізмів безпечного завершення сесій та очищення конфіденційних даних після виходу користувача з системи.

Тестування підтвердило високу ефективність впроваджених методів захисту та їх відповідність сучасним вимогам безпеки веб-додатків. Система успішно протистоїть широкому спектру загроз та забезпечує надійний захист користувацьких даних.

3.3 Рекомендації щодо оптимізації та вдосконалення системи захисту

На основі проведеного тестування та аналізу існуючої системи захисту можна сформулювати ряд рекомендацій щодо подальшої оптимізації та вдосконалення механізмів безпеки. В першу чергу рекомендується оновити

механізм хешування паролів з використанням більш сучасних алгоритмів. Замість поточної реалізації з SHA-256 (див. лістинг 3.18).

Лістинг 3.18 – Поточна реалізація з SHA-256

```
public static string HashPassword(string password)
{
    using (var sha256 = SHA256.Create())
    {
        var hashedBytes = 
sha256.ComputeHash(Encoding.UTF8.GetBytes(password));
        return  BitConverter.ToString(hashedBytes).Replace("-", 
"").ToLower();
    }
}
```

Рекомендується впровадити більш надійний алгоритм, такий як Argon2id або bcrypt, з додаванням солі та перцю для підвищення стійкості до атак методом перебору [32].

Для підвищення безпеки сесій рекомендується модифікувати поточні налаштування, додавши додаткові параметри захисту (див. лістинг 3.19).

Лістинг 3.19 – Зменшили час простою на 15 хвилин

```
builder.Services.AddSession(options =>
{
    options.IdleTimeout = TimeSpan.FromMinutes(15); // Зменшити 
час простою
    options.Cookie.HttpOnly = true;
    options.Cookie.IsEssential = true;
  options.Cookie.SecurePolicy = CookieSecurePolicy.Always;
    options.Cookie.SameSite = SameSiteMode.Strict;
});
```

Рекомендується впровадити систему двофакторної автентифікації (2FA) для додаткового рівня захисту облікових записів користувачів. Це може бути

реалізовано через інтеграцію з сервісами генерації одноразових кодів або використання апаратних ключів безпеки.

Належною рекомендацією є впровадження більш строгої політики безпеки контенту (CSP). Поточні налаштування можна розширити (див. лістинг 3.20).

Лістинг 3.20 – Імпорт бібліотек та оголошення змінних

```
app.Use(async (context, next) =>
{
    context.Response.Headers.Add("Content-Security-Policy",
        "default-src 'self'; " +
"script-src 'self' 'nonce-{random}'; " +
        "style-src 'self' 'nonce-{random}'; " +
        "img-src 'self' data:; " +
 "font-src 'self'");
    await next();
});
```

Рекомендується впровадити систему моніторингу безпеки в реальному часі з використанням SIEM-систем для більш ефективного виявлення та реагування на інциденти безпеки.

Для покращення захисту від SQL-ін'єкцій рекомендується додатково впровадити систему підготовлених запитів та параметризації всіх взаємодій з базою даних через Entity Framework Core. Рекомендується впровадити систему автоматичного сканування залежностей на наявність відомих вразливостей та автоматичного оновлення компонентів системи. Помітним аспектом вдосконалення є впровадження системи безперервного тестування безпеки (continuous security testing) в процес розробки та розгортання додатку, що дозволить своєчасно виявляти та усувати потенційні вразливості [33].

Рекомендується розширити систему логування, додавши детальну інформацію про всі критичні операції та спроби порушення безпеки. Для підвищення захисту від DDoS-атак рекомендується впровадити систему автоматичного масштабування та балансування навантаження. Важливою

рекомендацією є впровадження системи управління секретами для безпечного зберігання та розповсюдження криптографічних ключів та інших конфіденційних даних, що забезпечить централізоване управління та контроль доступу до критичної інформації.

Рекомендується впровадити систему автоматичного резервного копіювання з шифруванням резервних копій та регулярною перевіркою їх цілісності. Для покращення захисту від XSS-атак рекомендується впровадити додаткові механізми санітизації користувацького введення та валідації вихідних даних. Рекомендується розробити та впровадити політику управління паролями, включаючи вимоги до складності паролів та регулярної їх зміни, що підвищить загальний рівень безпеки облікових записів користувачів.

Значущим аспектом є впровадження системи автоматичного виявлення та блокування підозрілої активності на основі аналізу поведінки користувачів. Рекомендується впровадити механізми захисту від атак на рівні протоколу HTTP/2, включаючи захист від downgrade-атак та експлуатації вразливостей протоколу. Для підвищення безпеки рекомендується впровадити систему управління доступом на основі ролей (RBAC) з детальним розмежуванням прав та привілеїв, що забезпечить точний контроль доступу до функціональності системи.

Цінною рекомендацією є регулярне проведення тестування на проникнення та аудиту безпеки з залученням зовнішніх експертів. Рекомендується впровадити систему автоматичного оновлення SSL/TLS сертифікатів та моніторингу їх терміну дії. Для покращення захисту від фішингових атак рекомендується впровадити систему перевірки автентичності доменних імен та сертифікатів, що допоможе запобігти атакам через підміну ресурсів [20].

Помітним аспектом є впровадження системи навчання користувачів з питань інформаційної безпеки та регулярне оновлення навчальних матеріалів. Рекомендується впровадити систему автоматичного виявлення та блокування спроб несанкціонованого доступу до адміністративних інтерфейсів. Для підвищення загального рівня безпеки рекомендується регулярно проводити

аналіз та оновлення політик безпеки відповідно до нових загроз та вразливостей, забезпечуючи актуальність та ефективність системи захисту.

Впровадження всіх запропонованих рекомендацій дозволить значно підвищити рівень захищеності системи та забезпечити її відповідність сучасним вимогам інформаційної безпеки.

Рекомендується вдосконалити систему біометричної автентифікації через впровадження додаткових методів верифікації, таких як розпізнавання голосу та поведінкова біометрія. Це значно підвищить рівень безпеки при збереженні зручності користування системою [34].

Цінною рекомендацією є впровадження механізму Zero Trust Architecture (ZTA) з постійною верифікацією кожного запиту та оцінкою рівня довіри до кожної операції в системі. Це може бути реалізовано через модифікацію поточної конфігурації автентифікації (див. лістинг 3.21).

Лістинг 3.21 – Модифікація поточної конфігурації автентифікації

```
builder.Services.AddAuthentication(options =>
{
    options.DefaultScheme = "Zero Trust Scheme";
    options.DefaultChallengeScheme = "Continuous Verification";
})
.AddCustomZeroTrust(options =>
{
 options.ValidateOnEachRequest = true;
    options.RequireRiskAssessment = true;
});
```

Рекомендується впровадити систему квантово-стійкої криптографії для забезпечення довгострокового захисту критичних даних. Це особливо вагомо для даних, які повинні зберігати конфіденційність протягом тривалого періоду.

Ключовим елементом вдосконалення є впровадження системи автоматичної класифікації даних за рівнем конфіденційності та застосування відповідних рівнів захисту в залежності від категорії даних. Рекомендується

розробити та впровадити систему автоматичного виявлення аномалій в поведінці користувачів з використанням методів машинного навчання для раннього виявлення потенційних загроз. Для підвищення безпеки рекомендується впровадити систему захисту від атак на рівні DNS, включаючи DNSSEC та DNS over HTTPS, що забезпечить додатковий рівень захисту від атак типу DNS spoofing.

Належним кроком є впровадження системи управління привілейованими обліковими записами (PAM) з детальним аудитом всіх адміністративних дій та обмеженням часу дії підвищених привілеїв. Рекомендується розширити систему моніторингу продуктивності для включення аналізу аномалій в використанні ресурсів, що може вказувати на потенційні атаки або спроби зловживання системою [35].

Для покращення захисту від атак на рівні веб-застосунку рекомендується впровадити Web Application Firewall (WAF) з розширеними правилами фільтрації та можливістю навчання на основі реальних атак. Суттєвим елементом вдосконалення є впровадження системи автоматизованого управління відповідністю вимогам безпеки (compliance management) для забезпечення постійної відповідності регуляторним вимогам та стандартам безпеки.

РОЗДІЛ 4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКИ В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1 Охорона праці

Охорона праці та забезпечення безпеки в надзвичайних ситуаціях є важливими складовими стабільної та ефективної діяльності будь-якого підприємства чи організації. Ці заходи спрямовані на захист життя, здоров'я та працездатності працівників, а також на мінімізацію ризиків, пов'язаних із виникненням небезпечних ситуацій. Розробка комплексної системи охорони праці передбачає впровадження інноваційних підходів, інтеграцію сучасних технологій та дотримання законодавчих і міжнародних стандартів безпеки.

Основні етапи охорони праці:

1. Аналіз умов праці

Перший етап будь-якої системи охорони праці полягає в аналізі робочого середовища та умов праці на підприємстві. Основна мета цього етапу — виявити потенційні небезпеки та оцінити ризики для здоров'я працівників. Це досягається шляхом інспекцій робочих місць, використання інструментів оцінки ризиків та медичних оглядів персоналу. Результати аналізу дозволяють визначити пріоритети для впровадження заходів безпеки.

Потенційно небезпечними факторами можуть бути:

- недотримання норм освітлення та вентиляції;
- надмірний шум чи вібрація на робочому місці;
- використання застарілого або несправного обладнання.

2. Розробка та впровадження плану заходів з охорони праці

На основі аналізу умов праці створюється комплексний план заходів, який регламентує всі аспекти охорони праці на підприємстві. Цей документ включає:

- перелік заходів із мінімізації ризиків;
- графік регулярних інструктажів і навчань;

- правила використання засобів індивідуального захисту (ЗІЗ);
- рекомендації з покращення робочого середовища.

3. Навчання персоналу

Ефективна система охорони праці неможлива без постійного навчання працівників. Регулярні інструктажі з техніки безпеки та тренування щодо дій у надзвичайних ситуаціях (НС) дозволяють підготувати персонал до потенційних ризиків. Основні напрямки навчання:

- правила користування ЗІЗ (шоломи, окуляри, рукавиці тощо);
- основи надання першої медичної допомоги;
- евакуація у разі виникнення НС (пожежа, вибух, хімічне забруднення).

У сучасних умовах підприємства повинні бути готовими до можливих надзвичайних ситуацій, таких як техногенні аварії, стихійні лиха чи кібератаки. Для цього розробляються спеціальні плани дій, які забезпечують ефективну організацію роботи у випадку НС. Основні заходи підготовки включають:

Розробка планів дій у надзвичайних ситуаціях: ці документи містять алгоритми евакуації персоналу, інструкції щодо використання аварійного обладнання та порядок інформування відповідних служб.

Інтеграція систем моніторингу та раннього попередження: моніторинг ризиків дозволяє своєчасно виявляти небезпеку та запобігати її розвитку. Наприклад, датчики диму та температури допомагають запобігти пожежі, а програмне забезпечення для аналізу кіберзагроз мінімізує ризик несанкціонованого доступу до даних.

Організація тренувань для персоналу: практичні заняття з моделювання НС дозволяють відпрацювати дії у критичних ситуаціях. Наприклад, працівники можуть навчитися правильно евакуюватися з приміщення чи діяти у випадку витоку шкідливих речовин.

4. Аудит та оцінка ефективності

Оцінка ефективності впроваджених заходів є ключовим етапом будь-якої програми з охорони праці. Регулярний аудит дозволяє виявити слабкі місця у системі безпеки та оперативно внести корективи. Основні методи аудиту включають:

- аналіз виконання плану заходів;
- опитування працівників щодо їхньої обізнаності з правилами безпеки;
- перевірка технічного стану обладнання.

Охорона праці та безпека у надзвичайних ситуаціях є комплексом заходів, які забезпечують захист життя і здоров’я працівників, стабільність роботи підприємства та збереження навколишнього середовища. Впровадження сучасних технологій, регулярний аналіз ризиків і навчання персоналу дозволяють створити безпечне робоче середовище, яке відповідає вимогам часу.

ВИСНОВКИ

У результаті проведеного дослідження було розроблено та впроваджено комплексну систему захисту веб-додатків, яка забезпечує високий рівень безпеки та відповідає сучасним вимогам кібербезпеки. Система успішно протистоїть широкому спектру загроз, включаючи SQL-ін'єкції, XSS-атаки та CSRF-атаки. Впроваджені механізми автентифікації та авторизації, реалізовані з використанням сучасних криптографічних алгоритмів та протоколів, забезпечують надійний захист користувацьких даних та успішно блокують спроби несанкціонованого доступу та автоматизовані атаки.

Розроблена система шифрування критичних даних, що використовує багаторівневий підхід до захисту, забезпечує конфіденційність та цілісність інформації на всіх етапах її обробки та зберігання. Впроваджені механізми моніторингу та аудиту безпеки дозволяють ефективно виявляти та реагувати на потенційні загрози, забезпечуючи своєчасне запобігання інцидентам безпеки. Проведене тестування підтвердило високу ефективність розроблених методів захисту, де система демонструє стійкість до різних типів атак та забезпечує стабільну роботу веб-додатку під навантаженням.

Розроблені рекомендації щодо подальшої оптимізації системи захисту дозволять підтримувати високий рівень безпеки та адаптуватися до нових типів загроз. Економічна ефективність впровадження розробленої системи захисту підтверджується зниженням ризиків фінансових втрат від можливих кібератак та витоків даних. Практичне значення отриманих результатів полягає в можливості їх використання для захисту широкого спектру веб-додатків, а також в якості основи для розробки нових систем безпеки.

Впроваджені протоколи безпечної передачі даних та механізми захисту сесій користувачів продемонстрували високу надійність у запобіганні атакам перехоплення та підміни даних, де система успішно забезпечує безпечну взаємодію між клієнтською та серверною частинами додатку. Розроблена система управління доступом та розмежування прав користувачів забезпечує точний контроль над функціональністю додатку та доступом до даних, а

реалізовані механізми RBAC (Role-Based Access Control) дозволяють гнучко налаштовувати права доступу відповідно до потреб організації.

Суттєвим досягненням є успішна інтеграція розробленої системи захисту з існуючою інфраструктурою та бізнес-процесами, що забезпечило безперервність роботи та мінімальний вплив на користувацький досвід. Система демонструє оптимальний баланс між безпекою та зручністю використання, що є критичним фактором для сучасних веб-додатків.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Береза А.М. Основи створення інформаційних систем: навч. посіб. Київ: КНЕУ, 2023. 214 с.
2. Васильєв Ю.В., Петренко А.І. Сучасні методи захисту веб-додатків від кібератак. Кібербезпека в Україні. 2023. №4. С. 15-22.
3. Tymoshchuk, V., Mykhailovskyi, O., Dolinskyi, A., Orlovska, A., & Tymoshchuk, D. (2024). OPTIMISING IPS RULES FOR EFFECTIVE DETECTION OF MULTI-VECTOR DDOS ATTACKS. Матеріали конференцій МЦНД, (22.11. 2024; Біла Церква, Україна), 295-300.
4. Домарєв В.В., Домарєв Д.В. Безпека інформаційних технологій. Системний підхід. Київ: ТІД ДС, 2023. 992 с.
5. Lypa, B., Horyn, I., Zagorodna, N., Tymoshchuk, D., Lechachenko T., (2024). Comparison of feature extraction tools for network traffic data. CEUR Workshop Proceedings, 3896, pp. 1-11.
6. Tymoshchuk, D., Yasniy, O., Mytnyk, M., Zagorodna, N., Tymoshchuk, V., (2024). Detection and classification of DDoS flooding attacks by machine learning methods. CEUR Workshop Proceedings, 3842, pp. 184 - 195.
7. Tymoshchuk, V., Vantsa, V., Karnaukhov, A., Orlovska, A., & Tymoshchuk, D. (2024). COMPARATIVE ANALYSIS OF INTRUSION DETECTION APPROACHES BASED ON SIGNATURES AND ANOMALIES. Матеріали конференцій МЦНД, (29.11. 2024; Житомир, Україна), 328-332.
8. Лахно В.А., Петров О.С. Технології безпеки сучасних інформаційних систем. Київ: КПІ ім. Ігоря Сікорського, 2023. 452 с.
9. Tymoshchuk, V., Vorona, M., Dolinskyi, A., Shymanska, V., & Tymoshchuk, D. (2024). SECURITY ONION PLATFORM AS A TOOL FOR DETECTING AND ANALYSING CYBER THREATS. Collection of scientific papers «ΛΌГОΣ», (December 13, 2024; Zurich, Switzerland), 232-237.
10. Остапов С.Е., Євсеєв С.П., Король О.Г. Технології захисту інформації. Харків: ХНЕУ, 2023. 476 с.

11. Tymoshchuk, V., Pakhoda, V., Dolinskyi, A., Karnaukhov, A., & Tymoshchuk, D. (2024). MODELLING CYBER THREATS AND EVALUATING THE PERFORMANCE OF INTRUSION DETECTION SYSTEMS. Grail of Science, (46), 636–641. https://doi.org/10.36074/grail-of-science.29.11.2024.081
12. Погребенник В.Д., Партика Ю.Д. Інформаційні технології та системи. Київ: Український бестселер, 2023. 468 с.
13. Muzh, V., & Lechachenko, T. (2024). Computer technologies as an object and source of forensic knowledge: challenges and prospects of development. Вісник Тернопільського національного технічного університету, 115(3), 17-22
14. Tymoshchuk, D., & Yatskiv, V. (2024). Slowloris ddos detection and prevention in real-time. Collection of scientific papers «ΛΌГОΣ», (August 16, 2024; Oxford, UK), 171-176.
15. Anderson R. Security Engineering: A Guide to Building Dependable Distributed Systems. 3rd ed. Wiley, 2023. 1080 p.
16. Stanko, A., Wieczorek, W., Mykytyshyn, A., Holotenko, O., & Lechachenko, T. (2024). Realtime air quality management: Integrating IoT and Fog computing for effective urban monitoring. CITI, 2024, 2nd.
17. Content Security Policy Level 3. W3C Working Draft, 2024. URL: https://www.w3.org/TR/CSP3/ (дата звернення: 12.11.2024)
18. Derkach, M., Skarga-Bandurova, I., Matiuk, D., & Zagorodna, N. (2022, December). Autonomous quadrotor flight stabilisation based on a complementary filter and a PID controller. In 2022 12th International Conference on Dependable Systems, Services and Technologies (DESSERT) (pp. 1-7). IEEE.
19. Stadnyk, M. (2016, February). The informative parameters determination for a visual system diagnostics by using the steady state visual evoked potentials. In 2016 13th International Conference on Modern Problems of Radio Engineering, Telecommunications and Computer Science (TCSET) (pp. 800-803). IEEE.

20. OWASP API Security Project. OWASP Foundation, 2024. URL: https://owasp.org/www-project-api-security/ (дата звернення: 12.11.2024)
21. Secure Coding Guidelines. Microsoft, 2024. URL: https://docs.microsoft.com/security/secure-coding-guidelines/ (дата звернення: 12.11.2024)
22. ТИМОЩУК, Д., & ЯЦКІВ, В. (2024). USING HYPERVISORS TO CREATE A CYBER POLYGON. MEASURING AND COMPUTING DEVICES IN TECHNOLOGICAL PROCESSES, (3), 52-56.
23. ТИМОЩУК, Д., ЯЦКІВ, В., ТИМОЩУК, В., & ЯЦКІВ, Н. (2024). INTERACTIVE CYBERSECURITY TRAINING SYSTEM BASED ON SIMULATION ENVIRONMENTS. MEASURING AND COMPUTING DEVICES IN TECHNOLOGICAL PROCESSES, (4), 215-220.
24. Wheeler D.A. Secure Programming HOWTO. 2024. URL: https://dwheeler.com/secure-programs/ (дата звернення: 12.11.2024)
25. XSS Prevention Rules. OWASP Foundation, 2024. URL: https://cheatsheetseries.owasp.org/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.html (дата звернення: 12.11.2024)
26. T. Lechachenko, R. Kozak, Y. Skorenkyy, O. Kramar, O. Karelina. Cybersecurity Aspects of Smart Manufacturing Transition to Industry 5.0 Model. CEUR Workshop Proceedings, 2023, 3628, pp. 325–329
27. Yesin, V., Karpinski, M., Yesina, M., Vilihura, V., Kozak, R., Shevchuk, R. (2023). Technique for Searching Data in a Cryptographically Protected SQL Database. Applied Sciences, 13(20), art. no. 11525, 1-21. doi: 10.3390/app132011525.
28. Balyk, A., Iatsykovska, U., Karpinski, M., Khokhlachova, Y., Shaikhanova, A., & Korkishko, L. (2015, September). A survey of modern IP traceback methodologies. In 2015 IEEE 8th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS) (Vol. 1, pp. 484-488). IEEE.

29. Krivulya, G., Skarga-Bandurova, I., Tatarchenko, Z., Seredina, O., Shcherbakova, M., & Shcherbakov, E. (2019, August). An intelligent functional diagnostics of wireless sensor network. In 2019 7th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW) (pp. 135-139). IEEE.
30. Biloborodova, T., Skarga-Bandurova, I., Derkach, M., Matiuk, D., & Zagorodna, N. (2024). Identification of Salient Brain Regions for Anxiety Disorders Using Nonlinear EEG Feature Analysis. Studies in health technology and informatics, 321, 180-184.

Додаток А – Публікація

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
імені ІВАНА ПУЛЮЯ
НАУКОВЕ ТОВАРИСТВО ім. ШЕВЧЕНКА

М А Т Е Р І А Л И

XII НАУКОВО-ТЕХНІЧНОЇ КОНФЕРЕНЦІЇ

«ІНФОРМАЦІЙНІ МОДЕЛІ, СИСТЕМИ ТА ТЕХНОЛОГІЇ»

18-19 грудня 2024 року

ТЕРНОПІЛЬ
2024

ПРОГРАМНИЙ КОМІТЕТ

Голова: Приймак Микола – професор кафедри комп'ютерних систем та мереж, д.т.н., професо**р.**

Співголови: Марущак Павло – проректор з наукової роботи, докт. техн. наук, професор.

Баран Ігор – канд. техн. наук, доцент, декан факультету ФІС.

Науковий секретар: Надія Крива– старший викладач.

Члени: Василь Кривень - завідувач кафедри математичних методів в інженерії д.ф.-м.н., професор; Галина Осухівська – завідувач кафедри комп'ютерних систем та мереж, к.т.н., доцент; Микола Карпінський - професор кафедри кібербезпеки, д.т.н., професор; Жанна Баб'як - завідувач кафедри української та іноземних мов, к.пед. н., доцент; Ярослав Литвиненко – професор кафедри комп'ютерних наук, д.т.н., професор; Михайло Петрик - завідувач кафедри програмної іженерії, д.ф.-м.н., професор; Наталія Загородна – завідувач кафедри кібербезпеки, к.т.н., доцент.

ОРГАНІЗАЦІЙНИЙ КОМІТЕТ

Голова: Скоренький Юрій Любомирович – канд. техн. наук, доцент кафедри фізики.

Члени: доцент кафедри комп'ютерних наук, к.т.н. В. Никитюк; доцент кафедри програмної інженерії, к.т.н. Д. Михалик; доцент кафедри кібербезпеки, к.т.н. М. Стадник; доцент кафедри комп'ютерних систем та мереж, к.т.н. Є. Тиш; ст. викладач Л. Джиджора.

М34 Матеріали XII науково-технічної конфції «Інформаційні моделі, системи та технології» Тернопільського національного технічного університету імені Івана Пулюя, (Тернопіль, 18–19 грудня 2024 р.). – Тернопіль : Тернопільський національний технічний університет імені Івана Пулюя, 2024.

Адреса оргкомітету: ТНТУ ім. І. Пулюя, м. Тернопіль, вул. Руська, 56, 46001,
тел. (0352) 52-41-33, факс (0352) 25-49-83.
E-mail: conffis2024@gmail.com

Редагування, оформлення та верстка: Надія Крива

СЕКЦІЇ КОНФЕРЕНЦІЇ, ЯКІ ПРЕДСТВЛЕНІ В ЗБІРНИКУ

- Математичне моделювання
- Інформаційні системи та технології, кібербезпека
- Комп'ютерні системи та мережі
- Програмна інженерія та моделювання складних розподілених систем
- Новітні фізико-технічні та освітні технології

В збірнику надруковано тези доповідей XII науково-технічної конференції «Інформаційні моделі, системи та технології» (Тернопіль, 18–19 грудня 2024 р.) за такими науковими напрямками: математичне моделювання; інформаційні системи та технології, кібербезпека; комп'ютерні системи та мережі; програмна інженерія та моделювання складних розподілених систем; новітні фізико-технічні та освітні технології.

Розрахований на науковців, викладачів та студентів вузів.

За зміст тез та дотримання норм академічної доброчесності відповідальність несе автор.

004.056.53
Кончак М.Б.,
Тернопільський національний технічний університет імені Івана Полюя

АНАЛІЗ МЕТОДІВ ЗАХИСТУ ВЕБ ДОДАТКІВ ВІД КІБЕРАТАК

Konchak Mariia

ANALYSIS OF METHODS FOR PROTECTING WEB APPLICATIONS FROM CYBER ATTACKS

З розвитком цифрових технологій веб-додатки стали важливою частиною сучасної інформаційної інфраструктури. Їх використання супроводжується ризиками кібератак, які можуть призвести до значних фінансових, репутаційних та інформаційних втрат.

Основні типи атак (згідно з рейтингом OWASP Top 10, 2024):

1. SQL-ін’єкції. Атаки, спрямовані на маніпулювання запитами до бази даних.
2. XSS-атаки (Cross-Site Scripting). Використовують уразливості для виконання шкідливих скриптів.
3. DDoS-атаки (Distributed Denial of Service). Перевантаження серверів з метою порушення їхньої роботи.

Методи захисту:

1. Валідація та фільтрація даних. Забезпечення коректної обробки вхідних даних.
2. Використання міжсайтових політик (CSP). Обмеження джерел виконання скриптів.
3. Шифрування даних. Захист переданої інформації за допомогою протоколів HTTPS.
4. Регулярні оновлення програмного забезпечення. Усунення відомих уразливостей.
5. Використання WAF (Web Application Firewall). Виявлення та блокування підозрілих запитів.

Аналіз сучасних методів захисту веб-додатків свідчить про необхідність комплексного підходу до кібербезпеки, який включає як технологічні рішення, так і організаційні заходи. Забезпечення безпеки веб-додатків є ключовим аспектом у захисті цифрових активів організацій.

Література

1. OWASP. *Top 10 Web Application Security Risks*. 2024.
2. Stallings W. *Network Security Essentials: Applications and Standards*. Pearson, 2022.
3. Symantec. *Internet Security Threat Report*. 2023.

Додаток Б

Для кожного стовпчика у таблицях є свій показник пояснення:

Для критерію "Складність впровадження":

- Низька: впровадження займає до 2 тижнів роботи команди розробників, не потребує значних змін в архітектурі системи
- Середня: впровадження займає 2-8 тижнів, потребує часткової модифікації архітектури та додаткового навчання персоналу
- Висока: впровадження займає понад 8 тижнів, вимагає суттєвої перебудови архітектури та значних ресурсів

Для критерію "Вплив на продуктивність":

- Мінімальний/Низький: збільшення часу відгуку системи до 5 мс
- Середній: збільшення часу відгуку системи на 5-10 мс
- Високий: збільшення часу відгуку системи більше ніж на 10 мс

Для критерію "Вартість підтримки":

- Низька: до 5% від початкової вартості впровадження на рік
- Середня: 5-15% від початкової вартості впровадження на рік
- Висока: понад 15% від початкової вартості впровадження на рік

Дані взяті із виконання застосунку та під час його тесту.