Вінницький національний технічний університет

Факультет менеджменту та інформаційної безпеки

Кафедра менеджменту та безпеки інформаційних систем

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Підвищення захищеності авторизації користувачів вдосконаленою системою з двоетапною генерацією та перевіркою токенів доступу на основі метаданих користувача»

Виконав: ст. 2-го курсу, групи КІТС-22 мз
спеціальності 125– Кібербезпека
Освітня програма – Кібербезпека
інформаційних технологій та систем
(шифр і назва напряму підготовки, спеціальності)

_____ Грабар Р.І._____
(прізвище та ініціали)

Керівник: д.т.н., проф. каф. МБІС
_____ Яремчук Ю.Є._____
(прізвище та ініціали)
«____» _____ 2024 р.

Опонент: к.т.н., доцент, каф. ОТ
_____ Колесник І.С.._____
(прізвище та ініціали)
«____» _____ 2024 р.

Допущено до захисту
Голова секції УБ кафедри МБІС

_____Юрій ЯРЕМЧУК
“_____” _____2024 р.

Вінниця ВНТУ – 2024 рік

Вінницький національний технічний університет
Факультет менеджменту та інформаційної безпеки
Кафедра менеджменту та безпеки інформаційних систем

Рівень вищої освіти ІІ-й (магістерський)
Галузь знань 12 – Інформаційні технології
Спеціальність 125 – Кібербезпека
Освітньо-професійна програма - Кібербезпека інформаційних технологій та систем

ЗАТВЕРДЖУЮ
Голова секції УБ, кафедра МБІС

_____**Юрій ЯРЕМЧУК**
“_____”_____2024 р.

З А В Д А Н Н Я
на магістерську кваліфікаційну роботу студенту
Грабар Руслан Іванович
(прізвище, ім`я, по-батькові)

1. Тема роботи:

«Підвищення захищеності авторизації користувачів вдосконаленою системою з двоетапною генерацією та перевіркою токенів доступу на основі метаданих користувача»
Керівник роботи: Яремчук Ю.Є.д.т.н., професор каф. МБІС
(прізвище, ім`я, по-батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від “ 11” березня 2024 року № 81
2. Строк подання студентом роботи_за тиждень до захисту.__
3. Вихідні дані до роботи:
Стандарти, електронні джерела, підручники та наукові статті по темі, які стосуються теми магістерської кваліфікаційної роботи.
4. Зміст текстової частини:
У вступі викладено актуальність теми та сформульовано мету роботи. У першому розділі проведено теоретичний аналіз підвищення захищеності авторизації користувачів вдосконаленою системою з двоетапною генерацією та перевіркою токенів доступу на основі метаданих користувача. У другому розділі описано проектування системи підвищення захищеності авторизації користувачів, включаючи механізми збору та аналізу метаданих, розробку архітектури системи та алгоритмів збору й аналізу метаданих, а також алгоритмів двоетапної генерації та перевірки токенів. Третій розділ присвячено програмній реалізації розробленої системи, обґрунтовано вибір програмної мови та середовища розробки, описано

програмну реалізацію додатку та його інтерфейс. Четвертий розділ містить економічне обґрунтування розробки програмного забезпечення

5. Перелік ілюстративного матеріалу (з точним зазначенням обов`язкових креслень)
У дугому розділі магістерської кваліфікаційної роботи наведено 3 рисунка, у третьому розділі – 15 рисунків

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Основна частина			
I	Яремчук Ю.Є. д.т.н., професор каф. МБІС		
II	Яремчук Ю.Є. д.т.н., професор каф. МБІС		
III	Яремчук Ю.Є. д.т.н., професор каф. МБІС		
Економічна частина			
IV	Причепа І.В., к.е.н., доц. каф. ЕПВМ		

7.Дата видачі завдання _____2024 р.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів магістерської кваліфікаційної роботи	Строк виконання етапів роботи		Примітка
1.	Визначення напрямку магістерської роботи, формулювання теми	1.03.2024	15.03.2024	
2.	Аналіз предметної області обраної теми	15.03.2024	1.04.2024	
3.	Розробка роботи	1.04.2024	10.04.2024	
4.	Написання магістерської роботи на основі розробленої теми	10.04.2024	20.04.2024	
5.	Передзахист магістерської кваліфікаційної роботи	20.04.2024	10.05.2024	
6.	Виправлення, уточнення, корегування магістерської кваліфікаційної роботи	10.05.2024	4.06.2024	
7.	Захист магістерської кваліфікаційної роботи	10.06.2024	10.06.2024	

Студент _____ Грабар Р.І.
(підпис)

Керівник роботи _____ Яремчук Ю.Є.
(підпис)

АНОТАЦІЯ

УДК 004.56.5(043.2)

Грабар Р.І. Підвищення захищеності авторизації користувачів вдосконаленою системою з двоетапною генерацією та перевіркою токенів доступу на основі метаданих користувача

Магістерська кваліфікаційна робота зі спеціальності 125 – «Кібербезпека», освітня програма «Кібербезпека інформаційних технологій та систем». Вінниця: ВНТУ, 2024. 108с.

На укр.мові. Бібліогр.: 41 назв; рис.: 21; табл. 16

У магістерській кваліфікаційній роботі розглянуто питання підвищення захищеності авторизації користувачів через впровадження вдосконаленої системи з двоетапною генерацією та перевіркою токенів доступу на основі метаданих користувача. У першому розділі проведено теоретичний аналіз сучасних методів авторизації, розглянуто принципи побудови безпечних систем авторизації та проаналізовано вразливості цих систем. Особливу увагу приділено ролі метаданих користувача в авторизаційних процесах та перевагам використання двоетапної генерації і перевірки токенів.

Другий розділ присвячений проектуванню системи. Описано механізми збору та аналізу метаданих, розроблено архітектуру системи, алгоритми збору та аналізу метаданих, а також алгоритми двоетапної генерації та перевірки токенів.

У третьому розділі представлено програмну реалізацію розробленої системи. Обґрунтовано вибір програмної мови та середовища розробки, описано програмну реалізацію додатку та його інтерфейсу, а також проведено тестування та аналіз роботи реалізованого додатку.

Четвертий розділ присвячено економічній частині розробки. Оцінено комерційний потенціал програмного забезпечення, прогнозовано витрати на виконання наукової роботи та впровадження її результатів.

Ключові слова: токени доступу, двоетапна генерація, метадані

ABSTRACT

Grabar R.I. Increased security of user authorization with an improved system with two-step generation and verification of access tokens based on user metadata

Master's qualification thesis on specialty 125 - "Cybersecurity", educational program "Cybersecurity of information technologies and systems". Vinnytsia: VNTU, 2024. 130p.

In the Ukrainian language. Bibliography: 41 titles; Fig.: 21; table 16

The master's qualification work considered the issue of increasing the security of user authorization through the implementation of an improved system with two-stage generation and verification of access tokens based on user metadata. In the first chapter, a theoretical analysis of modern authorization methods was carried out, the principles of building secure authorization systems were considered, and the vulnerabilities of these systems were analyzed. Special attention is paid to the role of user metadata in authorization processes and the advantages of using two-stage token generation and verification.

The second section is devoted to system design. Metadata collection and analysis mechanisms are described, system architecture, metadata collection and analysis algorithms, as well as two-stage token generation and verification algorithms are developed.

The third section presents the software implementation of the developed system. The choice of the programming language and development environment is substantiated, the software implementation of the application and its interface is described, and testing and analysis of the implementation of the application is performed.

The fourth chapter is devoted to the economic part of development. The commercial potential of the software was assessed, the costs of carrying out scientific work and the implementation of its results, as well as the commercial effects of the implementation of the development, were forecasted.

Keywords: access tokens, two-stage generation, metadata

ЗМІСТ

ВСТУП

Актуальність. У сучасному світі інформаційних технологій обсяг персональних даних, що обробляються та зберігаються в різних інформаційних системах, зростає з кожним роком. Це створює значні виклики для забезпечення їх безпеки, оскільки кількість кіберзагроз та атак на інформаційні системи постійно збільшується. За даними різних досліджень, кількість інцидентів, пов'язаних з несанкціонованим доступом до конфіденційної інформації, зростає, що вимагає розробки та впровадження нових методів і засобів захисту.

Традиційні методи авторизації, такі як паролі або одноразові коди, все частіше виявляються недостатніми для забезпечення необхідного рівня безпеки. Атаки типу фішинг, підміна сесій, атаки на посередника та інші стають більш витонченими, що підвищує ризик компрометації облікових даних користувачів. У цьому контексті виникає потреба у впровадженні більш ефективних механізмів авторизації, які забезпечують підвищений рівень захисту.

Одним із перспективних напрямків у цій сфері є використання двоетапної генерації та перевірки токенів доступу на основі метаданих користувача. Метадані, такі як геолокація, час доступу, інформація про пристрій тощо, дозволяють значно підвищити точність ідентифікації користувачів та забезпечити додатковий рівень захисту. Впровадження такої системи дозволяє зменшити ризики несанкціонованого доступу та забезпечити більш надійний захист конфіденційної інформації.

Крім того, розвиток технологій машинного навчання та штучного інтелекту відкриває нові можливості для аналізу метаданих та виявлення підозрілої активності у реальному часі.

Таким чином, розробка та впровадження вдосконаленої системи авторизації з двоетапною генерацією та перевіркою токенів доступу на основі метаданих користувача є актуальним та важливим завданням, що сприяє підвищенню рівня захищеності інформаційних систем і забезпеченню конфіденційності та цілісності персональних даних.

Мета і задачі дослідження. Метою даної роботи є підвищення захищеності авторизації користувачів шляхом розробки та впровадження вдосконаленої системи з двоетапною генерацією та перевіркою токенів доступу на основі метаданих користувача.

Задачами дослідження є:

- проведення огляду сучасних методів авторизації.
- визначення принципів побудови безпечних систем авторизації.
- аналіз вразливостей існуючих систем авторизації.
- дослідження ролі метаданих користувача в процесі авторизації.
- розробка алгоритмів двоетапної генерації та перевірки токенів.
- проектування архітектури системи підвищення захищеності авторизації.
- реалізація програмного забезпечення та проведення тестування.

Об’єкт дослідження. Процеси авторизації користувачів у інформаційних системах.

Предмет дослідження. Методи підвищення захищеності авторизації користувачів за допомогою двоетапної генерації та перевірки токенів доступу на основі метаданих користувача.

Новизна роботи. Новизна роботи полягає у розробці та впровадженні вдосконаленої системи авторизації, яка використовує двоетапну генерацію та перевірку токенів доступу, що базується на метаданих користувача. Це дозволяє значно підвищити рівень безпеки, знижуючи ризик несанкціонованого доступу до системи та забезпечуючи більш точну ідентифікацію користувачів.

Практична цінність. Практична цінність роботи полягає у створенні ефективного інструменту для підвищення захищеності авторизації користувачів, що може бути впроваджений у різноманітні інформаційні системи. Розроблена система може бути використана у комерційних та державних установах для захисту конфіденційних даних, забезпечуючи високий рівень безпеки та надійності доступу.

1 ТЕОРЕТИЧНИЙ АНАЛІЗ ПІДВИЩЕННЯ ЗАХИЩЕНОСТІ АВТОРИЗАЦІЇ КОРИСТУВАЧІВ ВДОСКОНАЛЕНОЮ СИСТЕМОЮ З ДВОЕТАПНОЮ ГЕНЕРАЦІЄЮ ТА ПЕРЕВІРКОЮ ТОКЕНІВ ДОСТУПУ НА ОСНОВІ МЕТАДАНИХ КОРИСТУВАЧА

У першому розділі магістерської кваліфікаційної роботи буде проведений огляд сучасних методів авторизації, включаючи парольне входження, двофакторну аутентифікацію, біометричні методи та інші. Далі будуть визначені основні принципи побудови безпечних систем авторизації та проведений аналіз вразливостей, що можуть виникнути у процесі авторизації. Окремо буде розглянуто роль метаданих користувача у забезпеченні безпеки авторизації. Наступним кроком буде аналіз переваг та можливих використань двоетапної генерації та перевірки токенів доступу в системах авторизації.

Розділ завершиться висновками, у яких будуть сформульовані ключові висновки та поставлені завдання для подальшої розробки.

1.1 Огляд сучасних методів авторизації

У сучасному цифровому світі, де доступ до важливих даних та ресурсів є ключем до успіху, забезпечення безпеки ідентифікації користувачів стає пріоритетним завданням. Методи авторизації відіграють визначну роль у забезпеченні захищеного доступу до систем, програм та інформаційних ресурсів. Однак із зростанням кількості і складності цифрових загроз, необхідно постійно оновлювати та удосконалювати методи забезпечення ідентифікації.

Детально розглянемо такі методи авторизації як парольна, біометрична, двофакторна, мультифакторна, а також методи, що використовують токени доступу, смарт-карти та інші технології. Для кожного методу буде проведено аналіз його принципів, переваг і недоліків, а також відомостей про застосування в практичних системах та його відповідність вимогам безпеки [1].

Парольна авторизація ґрунтується на перевірці знання користувачем секретного пароля, який пов'язаний з його обліковим записом. При вході в систему

користувач вводить логін (зазвичай це адреса електронної пошти або ім'я користувача) та пароль. Система перевіряє, чи співпадає введений пароль з паролем, що зберігається в базі даних для даного логіну. Якщо паролі збігаються, користувачеві надається доступ до системи.

Важливим аспектом парольної авторизації є складність пароля. Складні паролі важче вгадати або зламати зловмисникам. Рекомендовано використовувати паролі, які:

– мають довжину не менше 12 символів;

– містять комбінацію великих і малих літер, цифр та символів;

– не містять особистої інформації (наприклад, дати народження, імен), слів словникового запасу або легко передбачуваних комбінацій (наприклад, "123456", "qwerty").

Паролі користувачів повинні зберігатися в безпечному місці. Не рекомендується записувати паролі на папері або зберігати їх у незашифрованому вигляді на комп'ютері. Більш безпечним способом є зберігання паролів у спеціальних менеджерах паролів, які використовують шифрування для захисту даних [2].

Парольна авторизація має ряд ризиків та вразливостей, таких як:

– слабкі паролі: багато користувачів обирають слабкі паролі, які легко вгадати;

– перебір паролів: зловмисники можуть використовувати атаки грубої сили для перебору паролів;

– фішинг: зловмисники можуть використовувати фішингові атаки, щоб обманом отримати від користувачів їхні паролі;

– витік паролів: паролі можуть бути скомпрометовані внаслідок витоків даних з веб-сайтів або програм.

Для підвищення безпеки парольної авторизації рекомендується використовувати такі заходи:

– створення складних паролів: використовувати довгі паролі, що містять комбінацію різних символів.

– зберігання паролів у безпечному місці: не записувати паролі на папері або зберігати їх у незашифрованому вигляді.

– використання двофакторної аутентифікації (2FA): 2FA додає додатковий рівень безпеки, окрім пароля.

– регулярна зміна паролів: регулярно змінювати паролі, особливо у випадку підозри на викрадення;

– використання надійних веб-сайтів та програм: уникати використання веб-сайтів та програм з сумнівною репутацією.

Парольна авторизація є простим і поширеним методом аутентифікації, але має ряд ризиків та вразливостей. Для підвищення безпеки рекомендується використовувати складні паролі, 2FA та інші заходи безпеки.

Двофакторна аутентифікація (2FA) – це метод підвищення безпеки авторизації користувачів, який додає додатковий рівень перевірки особи, окрім пароля. Замість того, щоб просто ввести пароль для входу в систему, користувачеві також необхідно надати другий фактор аутентифікації. Це робить процес входу більш складним для зловмисників, навіть якщо їм вдасться отримати пароль користувача [3].

2FA використовує два з трьох типів факторів аутентифікації:

– щось, що знає користувач (Knowledge Factor): Це зазвичай пароль або PIN-код, який користувач повинен пам'ятати;

– щось, що має користувач (Ownership Factor): Це фізичний об'єкт, який повинен мати користувач, наприклад, смартфон, смарт-карта або апаратний ключ безпеки;

– щось, чим є користувач (Inherence Factor): Це біометричні дані користувача, такі як відбиток пальця, розпізнавання обличчя або сканування райдужної оболонки ока.

Приклади використання факторів аутентифікації в 2FA:

– пароль + код з SMS-повідомлення: Користувач вводить пароль та код, який надсилається на його телефон;

– пароль + відбиток пальця: Користувач вводить пароль та прикладає палець до сканера відбитків пальців;

– апаратний ключ + PIN-код: Користувач вставляє апаратний ключ в USB-порт комп'ютера та вводить PIN-код.

Використання 2FA має ряд переваг порівняно з простою парольною авторизацією:

– підвищена безпека: Навіть якщо зловмисник отримає пароль користувача, йому все одно буде потрібен другий фактор аутентифікації для доступу до системи;

– захист від фішингових атак: Фішингові атаки намагаються обманом змусити користувачів ввести свої паролі на підроблених веб-сайтах. 2FA захищає від таких атак, оскільки зловмисник не матиме доступу до другого фактору аутентифікації;

– зменшення ризику витоків даних: Навіть якщо база даних паролів користувачів буде скомпрометована, зловмисники не зможуть отримати доступ до облікових записів без другого фактору аутентифікації.

Хоча 2FA є більш безпечним, ніж просто парольна авторизація, вона має деякі недоліки:

– незручність для користувачів: Введення додаткового фактору аутентифікації може бути незручним для деяких користувачів;

– залежність від додаткових пристроїв: Деякі методи 2FA, наприклад, SMS-повідомлення, потребують використання додаткових пристроїв, таких як мобільний телефон;

– можливість втрати другого фактору аутентифікації: Якщо користувач втратить свій телефон або апаратний ключ безпеки, він може втратити доступ до своїх облікових записів.

2FA є ефективним способом підвищення безпеки авторизації користувачів. Хоча вона має деякі недоліки, її переваги щодо захисту від несанкціонованого доступу роблять її цінним інструментом для захисту особистих даних. Користувачам рекомендується використовувати 2FA для всіх облікових записів, де зберігається важлива інформація.

Біометричні методи аутентифікації - це методи аутентифікації користувачів, які ґрунтуються на їх унікальних фізичних або поведінкових характеристиках. Ці методи стають все більш популярними, адже вони вважаються більш безпечними та зручними, ніж традиційні методи, такі як паролі.

Існує два основних типи біометричних методів:

– фізіологічні: Ці методи ґрунтуються на фізичних характеристиках користувача, таких як:

Відбиток пальця: Один з найпоширеніших методів, який використовує сканер для зчитування відбитків пальців.

Розпізнавання обличчя: Веб-камера використовується для зчитування рис обличчя користувача.

Сканування райдужної оболонки ока: Цей метод використовує спеціальний сканер для зчитування унікального візерунка райдужної оболонки ока.

Геометрія руки: Цей метод використовує сканер для зчитування форми та розміру руки.

– поведінкові: Ці методи ґрунтуються на поведінкових характеристиках користувача, таких як:

Підпис: Спеціальний планшет використовується для зчитування підпису користувача.

Натискання клавіш: Цей метод аналізує манеру натискання клавіш користувачем.

Динаміка голосу: Цей метод аналізує голос користувача.

Біометричні методи аутентифікації мають ряд переваг порівняно з традиційними методами:

– підвищена безпека: біометричні дані користувача є унікальними та складними для підробки;

– зручність: біометричні методи аутентифікації швидкі та прості у використанні;

– захист від крадіжки паролів: біометричні дані не можна вкрасти або вгадати.

Біометричні методи аутентифікації також мають деякі недоліки:

– вартість: біометричні сканери можуть бути дорогими;

– точність: деякі біометричні методи можуть бути неточними, що може призвести до помилок аутентифікації;

– питання конфіденційності: Деякі люди стурбовані тим, що збір та зберігання біометричних даних може призвести до порушення конфіденційності.

Біометричні методи аутентифікації є безпечним і зручним способом захисту доступу до облікових записів користувачів. Незважаючи на деякі недоліки, їхні переваги роблять їх цінним інструментом для забезпечення безпеки даних.

Одноразові паролі (OTP) – це паролі, які генеруються випадковим чином і використовуються лише один раз. Ці паролі зазвичай генеруються програмним забезпеченням або апаратними ключами. OTP використовуються для підвищення безпеки аутентифікації користувачів, адже зловмисник, який отримає OTP, не зможе використовувати його повторно.

OTP можуть генеруватися різними способами:

– програмним забезпеченням: Існує багато програм, які генерують OTP. Ці програми зазвичай генерують OTP на основі алгоритму, який використовує час, хеш-функції та інші фактори для генерації унікального пароля;

– апаратними ключами: Існують апаратні ключі, які генерують OTP. Ці ключі зазвичай мають вбудований генератор випадкових чисел, який використовується для генерації унікального пароля.

OTP мають ряд переваг порівняно з традиційними паролями:

– підвищена безпека: OTP не можна вгадати або вкрасти, адже вони використовуються лише один раз.

– захист від атак грубої сили: Зловмисники не можуть використовувати атаки грубої сили для підбору OTP, адже він використовується лише один раз.

– зручність: OTP прості у використанні, адже користувачеві не потрібно запам'ятовувати їх.

OTP також мають деякі недоліки:

– складність використання: Деякі користувачі можуть вважати OTP складними у використанні, адже їм потрібно отримувати новий пароль для кожного входу.

– необхідність підключення до Інтернету: Деякі методи генерації OTP потребують підключення до Інтернету, що може бути проблемою в деяких ситуаціях.

OTP є безпечним та зручним способом аутентифікації користувачів. Незважаючи на деякі недоліки, їхні переваги роблять їх цінним інструментом для забезпечення безпеки даних [4].

Сучасні методи авторизації пропонують широкий спектр можливостей, від традиційних паролів до нових безпарольних методів. Кожен метод має свої переваги та недоліки, тому важливо вибрати метод, який відповідає потребам у безпеці та зручності.

1.2 Принципи побудови безпечних систем авторизації

Авторизація - це ключовий етап в захисті конфіденційності та цілісності даних, а також обмеженні доступу до них. Детально розглядаються принципи та методи, які лежать в основі створення безпечних систем авторизації, зокрема принцип найменшого доступу, використання криптографічних методів, механізми аутентифікації, моніторинг та аналіз активності користувачів, а також строге керування ідентифікацією і доступом. Розгляд цих аспектів дозволить визначити

ключові принципи та стратегії для побудови ефективних та надійних систем авторизації, які забезпечать високий рівень захисту даних та доступу до них.

Принцип найменшого доступу (Principle of Least Privilege) - це концепція в інформаційній безпеці, яка стверджує, що кожен користувач або системний процес повинні мати тільки необхідний рівень доступу, не більше. Цей принцип забезпечує обмеження прав доступу користувачів до ресурсів і функціональності системи лише на необхідний мінімальний рівень для виконання їхніх обов'язків або завдань. Детальніше розглянемо основні аспекти цього принципу:

– мінімізація ризику: Обмеження прав доступу зменшує можливість зловживання, помилок або недбалості, що може призвести до порушення безпеки.

– обмеження привілеїв: Користувачі повинні мати лише ті привілеї, які необхідні для виконання їхніх конкретних завдань. Наприклад, звичайному користувачеві можуть бути надані обмежені права доступу до файлів і папок, тоді як адміністратор системи матиме повний доступ.

– підвищення безпеки: Застосування принципу найменшого доступу зменшує можливість неправомірного доступу до конфіденційної інформації або виконання небажаних дій.

– зменшення вразливостей: Чим менше прав доступу має користувач, тим менше вразливостей у системі. Обмеження доступу допомагає знизити ризик використання недоліків програмного забезпечення або втручання з боку зловмисників.

– аудит і моніторинг: Застосування принципу найменшого доступу полегшує аудит та моніторинг дій користувачів, оскільки кожен доступ до ресурсів буде обмеженим та легко відстежуваним.

ПНД має ряд переваг:

– підвищення безпеки: обмеження доступу знижує ризик несанкціонованого доступу до даних, зловживання ними або їх витоку;

– зменшення ризику помилок: користувачі, які мають обмежені права, не можуть мимоволі завдати шкоди системі або даним;

– краща керованість: легше контролювати та відстежувати дії користувачів, які мають чітко визначені права доступу;

– спрощення аудиту: простіше проводити аудит систем та даних, коли права доступу чітко визначені та обмежені.

ПНД може бути реалізований на різних рівнях:

– на рівні користувачів: користувачам повинні бути надані лише ті права, які необхідні для виконання їх посадових обов'язків;

– на рівні процесів: процеси повинні мати доступ лише до тих даних, які необхідні для їх функціонування;

– на рівні даних: доступ до конфіденційних даних повинен бути обмежений лише авторизованими користувачами.

Цей принцип є одним з основних стратегій в інформаційній безпеці та широко використовується при проектуванні та впровадженні систем авторизації для забезпечення високого рівня захисту даних [5].

Застосування криптографічних методів в інформаційних системах є ключовим для забезпечення безпеки даних. Розглянемо детально їхнє використання:

– шифрування: Шифрування дозволяє захистити конфіденційні дані шляхом перетворення їх у незрозумілу форму за допомогою спеціального ключа. Тільки особа, яка має відповідний ключ розшифровки, може прочитати зашифровані дані. Два основні типи шифрування - симетричне і асиметричне - забезпечують захист даних у різних сценаріях. Симетричне шифрування використовує один ключ для як шифрування, так і розшифрування даних, тоді як у асиметричному шифруванні використовується пара ключів: публічний та приватний. Такий підхід забезпечує надійний захист інформації під час її передачі по відкритій мережі.

– хешування: Хеш-функції використовуються для створення унікального "відбитка" (хешу) вхідного повідомлення фіксованої довжини. Однак не можливо зворотно отримати вхідне повідомлення з хешу. Це дозволяє використовувати хеш-функції для перевірки цілісності даних: якщо навіть один символ в початковому

повідомленні зміниться, хеш значно зміниться. Хеш-функції також широко використовуються в паролях для збереження їх у вигляді хешів, замість прямого збереження.

– цифрові підписи: Цифрові підписи використовуються для підтвердження автентичності повідомлень та документів. Підписування повідомлення приватним ключем гарантує, що воно було створено автентичним відправником, а перевірка підпису за допомогою відповідного публічного ключа дозволяє перевірити цілісність та автентичність повідомлення.

Загалом, криптографічні методи забезпечують надійний захист конфіденційності, цілісності та автентичності даних у системах передачі та зберігання, що робить їх невід'ємною частиною будь-якої сучасної системи інформаційної безпеки.

Аутентифікація - це процес перевірки особистості користувача. Механізми аутентифікації використовуються для захисту систем та даних від несанкціонованого доступу [6].

Механізми аутентифікації - це способи перевірки ідентичності користувача перед наданням доступу до системи або ресурсів. Розглянемо детально кілька основних механізмів аутентифікації:

– парольна аутентифікація: Це один з найпоширеніших методів, при якому користувач повинен ввести комбінацію ім'я користувача та пароль для входу в систему. Система перевіряє ці дані на відповідність збереженим даним у своїй базі даних. Недоліками цього методу є вразливість до перехоплення паролів та потреба встановлення складних правил для паролів.

– метод аутентифікації на основі біометричних даних: Використовує фізичні характеристики користувача, такі як відбиток пальця, розпізнавання обличчя або структура голосу, для ідентифікації. Цей метод відносно безпечний, оскільки вимагає прямого доступу до фізичних характеристик користувача, але може викликати проблеми з приватністю та точністю.

– аутентифікація за допомогою одноразових паролів (OTP): Кожен раз, коли користувач намагається увійти до системи, йому випускається унікальний одноразовий пароль, який він повинен ввести разом зі своїми звичайними ідентифікаторами. Цей метод забезпечує додатковий рівень безпеки, оскільки пароль може бути використаний лише один раз.

– метод аутентифікації на основі сертифікатів: Користувачі отримують цифрові сертифікати, що підтверджують їхню ідентичність, від авторитетного центру сертифікації. Під час аутентифікації користувач представляє свій сертифікат, а сервер перевіряє його відповідність з відповідними сертифікатами у своїй базі даних.

– аутентифікація за допомогою мультифакторної аутентифікації (MFA): Використовує комбінацію різних методів аутентифікації, таких як пароль, OTP, біометричні дані тощо, для забезпечення більшого рівня безпеки. Цей підхід дозволяє уникнути проблем, пов'язаних з використанням одного методу аутентифікації.

Ці механізми аутентифікації можуть використовуватися окремо або в комбінації, залежно від потреб конкретної системи та рівня безпеки, який вона вимагає [7].

Моніторинг та аналіз активності користувачів - це процес збору, зберігання та аналізу даних про дії користувачів в системі. Цей процес може використовуватися для різних цілей, таких як:

– підвищення безпеки: Виявлення підозрілої активності, наприклад, несанкціонованого доступу або спроб зловживання;

– удосконалення продуктивності: Виявлення вузьких місць та покращення досвіду користувачів;

– відстеження дотримання нормативних вимог: Забезпечення відповідності системі нормативним вимогам, таким як GDPR.

Існує багато методів моніторингу активності користувачів, деякі з яких:

– журнали: запис даних про дії користувачів в журналах;

– аналітика: використання інструментів аналітики для збору та аналізу даних про дії користувачів;

– моніторинг мережі: використання інструментів моніторингу мережі для відстеження активності користувачів в мережі.

Дані про активність користувачів можуть бути проаналізовані для виявлення закономірностей та тенденцій. Цей аналіз може бути використаний для різних цілей, таких як:

– виявлення аномалій: виявлення підозрілої активності, яка може свідчити про атаку або зловживання;

– сегментація користувачів: розбивка користувачів на групи за їх поведінкою;

– персоналізація: надання користувачам персоналізованого досвіду на основі їх поведінки.

Моніторинг та аналіз активності користувачів є важливим інструментом для підвищення безпеки, удосконалення продуктивності та відстеження дотримання нормативних вимог. Використання цього інструменту може допомогти вам захистити свої системи та покращити досвід користувачів [8].

Строге КІД - це комплексний підхід до забезпечення безпеки даних та ресурсів, який ґрунтується на трьох основних принципах:

– мінімальні привілеї: користувачі та процеси повинні мати лише ті права доступу, які необхідні для виконання їх поточних завдань;

– унікальна ідентифікація: користувачі та ресурси повинні мати унікальні ідентифікатори, які стійкі до підробки;

– контроль доступу на основі ролей: доступ до даних та ресурсів повинен надаватися на основі ролей, а не індивідуальних користувачів.

Строге КІД має ряд переваг:

– підвищення безпеки: обмеження доступу знижує ризик несанкціонованого доступу до даних, зловживання ними або їх витоку;

– зменшення ризику помилок: користувачі, які мають обмежені права, не можуть мимоволі завдати шкоди системі або даним;

– краща керованість: легше контролювати та відстежувати дії користувачів, які мають чітко визначені права доступу;

– спрощення аудиту: простіше проводити аудит систем та даних, коли права доступу чітко визначені та обмежені.

Строге КІД може бути реалізовано за допомогою різних заходів:

– сильна аутентифікація: використання складних паролів, біометричних даних або багатофакторної аутентифікації для перевірки особистості користувачів;

– управління доступом на основі ролей (rbac): надання користувачам прав доступу на основі їх ролей в організації;

– шифрування даних: захист даних під час передачі та зберігання;

– моніторинг та аудит: регулярний контроль та аналіз активності користувачів для виявлення підозрілої поведінки.

Строге КІД є важливим компонентом будь-якої програми інформаційної безпеки. Впровадження строгих заходів КІД може значно знизити ризик порушення безпеки та захистити ваші дані від несанкціонованого доступу.

Застосування цих принципів у комплексі може значно підвищити рівень безпеки систем авторизації та захистити дані від несанкціонованого доступу.

Безпечна система авторизації - це багаторівнева система, що ґрунтується на чітко визначених принципах та політиках. Дотримання цих принципів та постійна увага до питань інформаційної безпеки допоможуть вам захистити ваші дані та системи від несанкціонованого доступу.

1.3 Аналіз вразливостей у системах авторизації

Авторизація визначає, як користувачі отримують доступ до ресурсів та функціоналу системи, і відіграє ключову роль у забезпеченні конфіденційності, цілісності та доступності даних. Виявлення вразливостей у процесі авторизації дозволяє ідентифікувати можливі точки атак та ризики безпеки, що можуть бути

використані зловмисниками для незаконного доступу до системи або виконання шкідливих дій.

У цьому контексті, аналіз вразливостей в системах авторизації передбачає дослідження різних аспектів безпеки, таких як типи можливих атак, слабкі місця в механізмах авторизації, стійкість до перехоплення сесій, можливість виконання атак від перебору, обробка помилок та заходи безпеки, використовувані для захисту авторизації. Аналіз цих аспектів дозволяє розробникам та адміністраторам систем безпеки виявляти та виправляти вразливості, забезпечуючи високий рівень захищеності та впевненість в безпеці користувачів та даних [9].

Аналіз вразливостей у системах авторизації можна проводити за наступними критеріями:

– типи атак: Перевіряти, які конкретні види атак можуть бути використані для порушення авторизації, такі як SQL Injection, Cross-Site Scripting (XSS), атаки на перехоплення сесій тощо;

– доступність слабких місць: Виявлення можливих слабких місць у процесі авторизації, таких як слабкі паролі, недостатні перевірки даних або недоліки в механізмах аутентифікації;

– стійкість до перехоплення сесій: Оцінка вразливості системи до атак, які можуть призвести до перехоплення сесій та використання їх зловмисниками;

– можливість виконання атак від перебору: Аналіз можливості зловмисників отримати доступ до системи шляхом перебору паролів або інших методів атаки;

– обробка помилок: Оцінка того, як система обробляє помилки та вразливості, пов'язані з неправильною обробкою помилок в процесі авторизації;

– заходи безпеки: Оцінка того, які конкретні заходи безпеки використовуються для захисту авторизації, такі як шифрування паролів, використання механізмів двофакторної аутентифікації тощо;

– моніторинг та аудит: Перевірка того, чи належним чином налаштований моніторинг та аудит системи для виявлення та реагування на потенційні вразливості в авторизації.

Аналіз вразливостей у системах авторизації є важливою частиною розробки безпечних інформаційних систем. Давайте розглянемо деякі типові вразливості, які можуть бути виявлені при аналізі систем авторизації:

– слабкі паролі;

Слабкі паролі становлять серйозну загрозу безпеці системи, оскільки вони легко піддаються атакам перебору паролів, де зловмисники використовують програми для автоматизованого випробування всіх можливих комбінацій паролів. Такі паролі, як "123456" або "password", або інші легко вгадливі комбінації, зазвичай використовуються через їхню простоту та легкість запам'ятовування, але це робить їх надзвичайно вразливими.

Під час аналізу вразливостей системи авторизації слід звертати особливу увагу на наявність слабких паролів серед користувачів. Такі паролі можуть легко бути відтворені або вгадані зловмисниками, що може призвести до несанкціонованого доступу до системи та порушення конфіденційності та цілісності даних. Для запобігання цим ризикам важливо встановлювати політику використання паролів, яка вимагатиме складних та унікальних паролів, а також регулярно оновлювати їх.

– атаки перехоплення сесій;

Атаки перехоплення сесій - це вид кібератак, коли зловмисник перехоплює або викрадає ідентифікатор сесії або сеансовий токен, що використовується для авторизації користувача в системі. Ця вразливість може виникнути в різних місцях, таких як бездротові мережі, HTTP-запити, куки або параметри URL.

Коли зловмисник отримує доступ до ідентифікатора сесії, він може використовувати його для відновлення сесії авторизації під чужим ім'ям без необхідності знання пароля користувача. Це дозволяє зловмиснику отримати доступ до конфіденційної інформації або виконувати несанкціоновані дії в системі.

Щоб запобігти атакам перехоплення сесій, важливо використовувати надійні механізми авторизації та аутентифікації, такі як HTTPS для захищеного передачі даних, а також механізми перевірки та обміну сеансовими токенами, які надійно

захищені від перехоплення. Також важливо вживати заходів безпеки на рівні програмного забезпечення, такі як захист від XSS атак та використання безпечних куки.

– Cross-Site Scripting (XSS);

Cross-Site Scripting (XSS) - це тип кібератак, коли зловмисник внедрює та виконує злоякісний JavaScript-код у веб-сторінках, які потім відображаються у браузерах інших користувачів. Ці атаки використовуються для викрадення конфіденційної інформації, перехоплення сесій, або виконання інших шкідливих дій на веб-сторінках.

Існує декілька типів атак XSS:

– Stored XSS (збережена атака XSS): Зловмисник вносить злоякісний скрипт безпосередньо до веб-додатка, наприклад, до бази даних або в файл. Коли користувачі отримують доступ до цієї веб-сторінки, скрипт виконується у їхньому браузері;

– Reflected XSS (відбита атака XSS): Зловмисник створює посилання з параметрами, що містять злоякісний скрипт. Коли користувачі переходять за цим посиланням, скрипт виконується у їхньому браузері;

– DOM-based XSS (DOM-атака XSS): У цьому випадку скрипт виконується на клієнтській стороні, шляхом модифікації DOM-структури сторінки. Наприклад, якщо користувач вводить даний, який потім відображається безпосередньо на сторінці, злоякісний скрипт може бути виконаний.

Для запобігання атакам XSS необхідно:

Всі вхідні дані, отримані від користувачів або зовнішніх джерел, слід екранувати та очищати від HTML-тегів та JavaScript-коду.

Використовувати Content Security Policy (CSP), щоб обмежити виконання скриптів на сторінках.

Перевіряти дані, які виводяться на сторінку, та екрануйте або кодуйте їх відповідно до контексту.

Регулярно оновлювати програмне забезпечення, включаючи веб-сервер, фреймворки та бібліотеки, для виправлення вразливостей, які можуть бути використані для атак XSS.

– Injection Attacks;

Injection Attacks - це форма атак, де зловмисники використовують вразливості у програмному забезпеченні для впровадження та виконання злоякісного коду через вхідні дані. Однією з найпоширеніших форм таких атак є SQL Injection (SQLI), коли атакуючий вводить SQL-код через вхідні дані, які потрапляють у систему управління базами даних (СУБД). Однак, існують і інші види ін'єкційних атак, такі як LDAP Injection, XML Injection та Command Injection [10].

Для SQL Injection атакуючий може використовувати різні методи, включаючи вбудовування SQL-коду в поля введення веб-форм, параметри URL або навіть кукі-файли. Після цього, коли система обробляє ці дані, вона виконує вбудований SQL-код, що може призвести до витоку конфіденційної інформації, видалення або модифікації даних, або навіть отримання повного контролю над системою.

Для запобігання ін'єкційним атакам варто вживати наступні заходи:

– використовувати параметризовані запити: Замість вставки змінних безпосередньо у SQL-запити, використовуйте параметризовані запити, де значення передаються як параметри, що дозволяє базі даних правильно інтерпретувати їх як дані, а не як SQL-код;

– використовувати обмежені права доступу до бази даних: Надавати обмежені права доступу до бази даних користувачам та додаткам, щоб у разі вразливості зловмисники могли мінімізувати шкоду;

– перевіряти та очищувати вхідні дані: Перевіряти вхідні дані на наявність потенційно небезпечних символів та очищувати їх від таких символів перед виконанням запиту;

– використовувати бібліотеки ORM та фреймворки: Використання ORM (Object-Relational Mapping) або фреймворків допомагає автоматизувати генерацію безпечних SQL-запитів та уникнути прямої роботи з рядками SQL-коду.

– недостатні права доступу;

Недостатні права доступу є серйозною вразливістю в інформаційних системах, яка може призвести до непередбачуваних наслідків, таких як витік конфіденційної інформації або порушення цілісності даних. Ця вразливість виникає, коли користувачі мають більше прав доступу, ніж необхідно для виконання їхніх обов'язків у системі.

Основні проблеми, пов'язані з недостатніми правами доступу, включають:

– використання неправильних даних: Користувачі з надмірними правами доступу можуть мати можливість переглядати, редагувати або видаляти дані, до яких вони не повинні мати доступу. Це може призвести до використання неправильних даних або впровадження некоректних змін у системі;

– порушення конфіденційності: Якщо користувачі мають доступ до конфіденційної інформації, до якої вони не повинні мати доступу, це може призвести до витоку цієї інформації. Наприклад, зловмисники можуть скористатися недостатніми правами доступу, щоб отримати конфіденційні дані і використовувати їх у шкідливих цілях;

– підвищення ризику безпеки: Надмірні права доступу можуть створити додаткові шляхи для атак на систему. Наприклад, зловмисники можуть використовувати недостатні права доступу, щоб отримати доступ до важливих ресурсів або виконати дії, які можуть підірвати безпеку системи [11].

Для запобігання недостатнім правам доступу необхідно використовувати принцип найменшого доступу, який передбачає, що користувачам надаються тільки ті права, які є необхідними для виконання їхніх обов'язків у системі. Також важливо регулярно аудитувати та оновлювати права доступу, щоб переконатися, що вони відповідають поточним потребам користувачів і безпеці системи.

– недостатня перевірка даних;

Недостатня перевірка даних є серйозною вразливістю в інформаційних системах, яка може бути використана зловмисниками для впровадження шкідливого коду або отримання несанкціонованого доступу до системи. Ця вразливість виникає, коли вхідні дані, що надходять у систему від користувачів або інших джерел, не адекватно перевіряються на валідність і цілісність перед їх обробкою.

Основні проблеми, пов'язані з недостатньою перевіркою даних, включають:

– перехоплення сесій: Зловмисники можуть використовувати недостатню перевірку даних для перехоплення сесій, отримання несанкціонованого доступу до облікових записів користувачів або підробки запитів авторизації;

– атаки на введення: Зловмисники можуть впроваджувати шкідливий код у вхідні дані, що обробляються системою, щоб виконати атаки на введення. Це може призвести до виконання небажаних команд або отримання конфіденційної інформації з системи;

– витік конфіденційної інформації: Якщо система не адекватно перевіряє вхідні дані перед їх збереженням або відображенням, це може призвести до витоку конфіденційної інформації. Наприклад, зловмисники можуть використовувати недостатньо захищені форми вводу на веб-сайті, щоб отримати доступ до конфіденційних даних користувачів.

Для запобігання недостатній перевірці даних необхідно використовувати надійні методи перевірки вхідних даних на валідність, цілісність та безпеку. Це може включати валідацію введених даних на клієнтському та серверному рівнях, фільтрацію вхідних даних та обмеження допустимих символів, а також використання параметризованих запитів для запобігання SQL-ін'єкціям.

– недостатня обробка помилок;

Недостатня обробка помилок є вразливістю, яка може призвести до серйозних проблем у системах авторизації. Ця проблема виникає, коли програмне забезпечення не адекватно реагує на помилки, що виникають під час виконання, або надає зловмисникам надмірну інформацію про внутрішні процеси системи.

Основні аспекти недостатньої обробки помилок включають:

– витік чутливої інформації: Якщо програмне забезпечення відображає детальну інформацію про помилки, це може призвести до витоку чутливих даних або конфіденційної інформації. Наприклад, якщо система відображає повідомлення про помилку з об'єктивними даними, такими як шляхи файлів або запити до бази даних, це може стати джерелом інформації для зловмисників.

– атаки на авторизацію: Зловмисники можуть використовувати недостатню обробку помилок для виявлення вразливостей у системі авторизації та намагатися виконати атаки, такі як перехоплення сесій або введення в систему під чужим ім'ям.

– підказки для зловмисників: Деякі повідомлення про помилки можуть містити інформацію, яка надає зловмисникам підказки щодо структури системи або можливих атак. Наприклад, повідомлення про помилку можуть вказувати на конкретні поля вводу, які є вразливими до атаки.

Для запобігання недостатній обробці помилок необхідно належним чином налаштувати повідомлення про помилки таким чином, щоб вони не розголошували конфіденційну інформацію та не надавали зловмисникам зайвих відомостей про систему. Також важливо забезпечити моніторинг та аналіз журналів помилок для виявлення потенційних вразливостей та швидкої реакції на них [11].

– секретні параметри в URL.

Секретні параметри в URL включають конфіденційну інформацію, таку як ідентифікатор сесії або токен доступу, які передаються безпосередньо у URL-адресі запиту. Ця практика є небезпечною, оскільки URL-адреси можуть бути відкритими та доступними для перегляду, зокрема, у логах сервера, в історії браузера та на мережевих пристроях. Деякі наслідки неправильної обробки секретних параметрів в URL включають:

– перехоплення сесій: Зловмисники можуть перехопити URL-адресу з секретним параметром, що дасть їм можливість отримати доступ до сесії користувача. Це дозволить їм виконувати дії в системі від імені потерпілого користувача;

– підбір та модифікація параметрів: Зловмисники можуть відстежувати URL-адреси і намагатися виконати атаки на підбір та модифікацію параметрів. Вони можуть спробувати відтворити або змінити значення секретних параметрів для незаконного отримання доступу або виконання атак;

– порушення конфіденційності: Використання секретних параметрів у URL може порушити конфіденційність інформації. Це особливо небезпечно, якщо параметри містять особисті дані або іншу конфіденційну інформацію.

Для запобігання цим проблемам рекомендується використовувати більш безпечні методи передачі конфіденційних даних, такі як використання HTTP-запитів з використанням методів POST або передача секретних параметрів у заголовку запиту. Також важливо шифрувати секретні дані під час передачі за допомогою протоколів HTTPS та використовувати механізми безпеки, такі як токени аутентифікації [12].

Для кращого порівняння вразливостей у системах авторизації за наведеними критеріями, можна скласти таблицю, де кожна вразливість буде оцінена залежно від її впливу на ці критерії.

Таблиця 1.1 – Порівняння вразливостей у системах авторизації за критеріями аналізу.

Вразливість	Тип атаки	Доступність	Стійкість до перехоплення сесій	Заходи безпеки
Слабкі паролі	Атаки грубої сили, словникової атаки	Висока	Низька	Політика паролів, шифрування, 2FA
Перехоплення сесій	"Людина посередині", CSRF	Висока	Низька	HTTPS, CSRF-токені, HSTS
XSS	Введення JavaScript	Висока	Не застосовується	Валідація, екранування, CSP
Injection Attacks	SQL, NoSQL, LDAP	Висока	Не застосовується	Параметризована передача, валідація, брандмауери

Продовження таблиці 1.1

Недостатні права	Вертикальне/горизонтальне розширення	Висока	Не застосовується	Принцип мінімальних привілеїв, RBAC
Недостатня перевірка даних	Введення несанкціонованих даних	Висока	Не застосовується	Валідація, екранування, санітизація
Недостатня обробка помилок	Розкриття інформації	Висока	Не застосовується	Детальне логування, інформативні повідомлення
Секретні параметри в URL	Витік конфіденційних даних	Висока	Не застосовується	HTTPS, cookies/HTTP-заголовки

Аналіз вразливостей у системах авторизації за різними критеріями показує, що існують різні загрози для безпеки даних та доступу. Слабкі паролі, атаки перехоплення сесій, Cross-Site Scripting (XSS), Injection Attacks та інші вразливості можуть спричинити серйозні проблеми, якщо не вживаються відповідні заходи безпеки.

Для ефективного захисту системи авторизації необхідно впроваджувати комплексні заходи безпеки, такі як використання сильних паролів, захист від атак перехоплення сесій, фільтрація вхідних даних, обробка помилок, використання шифрування та механізми аудиту і моніторингу. Такий підхід дозволить зменшити ризики порушення безпеки та забезпечити надійний захист авторизаційної системи.

1.4 Роль метаданих користувача в авторизації

У контексті авторизації в інформаційних системах, роль метаданих користувача набуває важливого значення. Метадані, що описують різні аспекти користувача в системі, допомагають ефективно керувати доступом до ресурсів та забезпечують безпеку та персоналізацію.

Роль метаданих користувача в авторизації полягає в тому, що ці дані допомагають ідентифікувати користувача та встановлювати його права доступу до різних ресурсів в інформаційній системі. Метадані - це дані, що описують характеристики або властивості інших даних. У випадку авторизації, метадані користувача можуть включати такі дані, як ім'я, електронна пошта, роль в системі, права доступу, інформація про попередні дії користувача, тощо.

Розглянемо детальніше, як метадані користувача впливають на процес авторизації:

– ідентифікація користувача;

Ідентифікація користувача - це процес визначення та підтвердження особистої ідентичності особи, яка намагається отримати доступ до системи. Метадані, такі як ім'я, електронна пошта або унікальний ідентифікатор, відіграють ключову роль у цьому процесі. Давайте розглянемо детальніше, як ці метадані допомагають у ідентифікації користувача:

– ім'я: Ім'я користувача може бути використане як один зі засобів ідентифікації. Воно може бути унікальним для кожного користувача або використовувати в поєднанні з іншими метаданими для точнішої ідентифікації;

– електронна пошта: Кожен користувач може мати унікальну електронну адресу, яка служить унікальним ідентифікатором у системі. Підтвердження доступу за допомогою електронної пошти може бути ефективним методом ідентифікації;

– унікальний ідентифікатор: Це унікальний код або номер, який призначений кожному користувачу. Він може бути використаний для точної ідентифікації користувача незалежно від його ім'я або інших особистих даних.

Метадані ідентифікації користувача використовуються для перевірки, чи користувач дійсно той, за кого себе видає, та надання доступу до відповідних ресурсів або функціональностей. Збір і використання цих метаданих в авторизаційному процесі допомагає забезпечити безпеку та впевненість у тому, що лише правомірні користувачі мають доступ до системи.

– визначення ролі користувача;

Визначення ролі користувача є важливим аспектом авторизації та управління доступом у системі. Роль користувача визначає, які дії він може виконувати в системі та до яких ресурсів він має доступ [15]. Давайте розглянемо детальніше, як цей процес відбувається:

– встановлення прав доступу: Кожна роль користувача має свій набір дозволів і обмежень. При визначенні ролі користувача система присвоює йому певний набір прав доступу до функціональностей та ресурсів системи;

– надання дозволів: Ролі користувача можуть бути пов'язані з певними діями або завданнями в системі. Наприклад, адміністратор може мати повний доступ до всіх функцій та ресурсів, тоді як звичайний користувач може мати обмежений доступ лише до певних функцій;

– обмеження доступу: Ролі користувача можуть бути використані для обмеження доступу до конфіденційної інформації або критичних функцій системи. Це допомагає уникнути несанкціонованого доступу та забезпечує безпеку даних;

– управління ролями: Система повинна мати механізми для надання, зміни та видалення ролей користувача відповідно до потреб бізнесу та змін в організаційній структурі.

Визначення ролі користувача допомагає ефективно управляти доступом до ресурсів та забезпечує безпеку системи шляхом обмеження прав доступу лише до необхідних функцій та даних. Це важлива складова частина авторизаційного процесу, яка сприяє забезпеченню конфіденційності, цілісності та доступності даних у системі.

– права доступу;

Права доступу - це параметри, які визначають, які дії користувач може виконувати та до яких ресурсів він має доступ в системі. Вони грають важливу роль у забезпеченні безпеки та управління доступом до інформації та функціональності. Розглянемо детальніше, як це працює:

– визначення прав доступу: Права доступу вказують, які дії може виконувати користувач в системі. Наприклад, це може бути право на перегляд, редагування, видалення або створення даних;

– контроль доступу до ресурсів: Права доступу визначають, до яких конкретних ресурсів у системі має доступ користувач. Це можуть бути файли, папки, бази даних, веб-сторінки тощо;

– призначення прав за ролями: Часто права доступу призначаються на основі ролей користувачів. Наприклад, адміністратор системи може мати повний доступ до всіх ресурсів, тоді як звичайні користувачі можуть мати обмежений доступ;

– дотримання принципу найменшого доступу: Важливо, щоб користувачі мали лише необхідні права доступу для виконання своїх обов'язків. Це допомагає зменшити ризик неправомірного доступу до конфіденційної інформації;

– забезпечення безпеки даних: Права доступу грають ключову роль у забезпеченні конфіденційності, цілісності та доступності даних. Вони дозволяють контролювати, хто має доступ до якої інформації та як ця інформація може бути використана;

– управління правами: Системи часто мають механізми управління правами, які дозволяють адміністраторам призначати, змінювати та видаляти права доступу користувачів відповідно до їхніх потреб та ролей.

Права доступу є важливою складовою частиною систем авторизації, яка допомагає забезпечити безпеку даних та ефективне управління доступом користувачів до ресурсів системи [16].

– аудит та моніторинг;

Аудит та моніторинг відіграють критичну роль у забезпеченні безпеки та виявленні аномальних подій у системі. Розглянемо це детальніше:

– збір метаданих про дії користувачів: Метадані, які стосуються авторизації та дій користувачів у системі, збираються та записуються для подальшого аналізу. Це може включати історію входів користувача, дії, виконані ним у системі, час та місце входу тощо;

– виявлення аномальних дій: Шляхом аналізу зібраних метаданих можна виявити незвичайні або підозрілі дії користувачів, які можуть свідчити про можливі атаки або порушення безпеки. Наприклад, якщо виявлено спроби входу до системи з незвичайних місць або з незвичайних IP-адрес, це може бути підозріле;

– реагування на інциденти: Моніторинг та аудит допомагають швидко виявляти та реагувати на інциденти безпеки, такі як спроби несанкціонованого доступу або атаки. Це дозволяє оперативно вжити заходів для мінімізації збитків та запобігти подальшим атакам;

– удосконалення системи безпеки: Аналіз даних аудиту та моніторингу може розкрити слабкі місця у системі та допомогти удосконалити заходи безпеки. На основі отриманих даних можуть бути прийняті рішення щодо покращення архітектури системи або вдосконалення політик безпеки;

– відповідність стандартам безпеки: Моніторинг та аудит дій користувачів може бути важливим елементом відповідності з вимогами стандартів безпеки та регулювання. Збір та збереження даних про дії користувачів може бути необхідним для проведення аудитів безпеки та відповідності;

– підвищення свідомості користувачів: Результати аудиту та моніторингу можуть бути використані для навчання та підвищення свідомості користувачів про безпеку. Аналіз подій може допомогти ідентифікувати патерни небезпечної поведінки та надавати рекомендації з її уникнення.

Отже, аудит та моніторинг метаданих користувачів є важливими складовими частинами систем безпеки, які дозволяють виявляти, аналізувати та реагувати на можливі загрози та порушення безпеки.

– персоналізація та зручність.

Персоналізація та зручність грають ключову роль у забезпеченні задоволення користувачів та покращенні їхнього досвіду використання системи. Давайте розглянемо це детальніше:

– персоналізація інтерфейсу: Метадані про налаштування користувача, такі як мова, тема, розмір шрифту тощо, можуть використовуватися для налаштування

інтерфейсу системи під конкретні потреби користувача. Це дозволяє створювати індивідуалізовані інтерфейси, які краще відповідають вподобанням користувачів;

– рекомендації та персоналізований контент: За допомогою метаданих про вподобання користувача, система може надавати персоналізовані рекомендації щодо контенту, товарів або послуг. Це допомагає користувачам знаходити більш цікавий та корисний контент і поліпшує їхній досвід використання системи;

– автоматична настройка параметрів: Метадані про користувача можуть використовуватися для автоматичної настройки параметрів системи відповідно до його вподобань. Наприклад, автоматична настройка мови інтерфейсу або рекомендацій на основі геолокації;

– зручність використання: Збирання та використання метаданих дозволяє створювати більш зручні та інтуїтивно зрозумілі інтерфейси, що полегшує навігацію користувачів та знижує час, потрібний для виконання різних завдань;

– адаптація до потреб користувача: Аналіз метаданих дозволяє системі адаптуватися до змінних потреб користувачів та надавати їм більш індивідуалізований досвід використання, що сприяє збереженню лояльності користувачів та підвищенню задоволення від взаємодії з системою.

Отже, персоналізація та зручність, забезпечені за допомогою метаданих користувача, є важливими складовими частинами досвіду користувача, які сприяють підвищенню задоволення від використання системи та покращенню її ефективності [17].

У можна відзначити, що ці дані є ключовим елементом для ефективної та безпечної авторизації користувачів у інформаційних системах. Метадані, такі як ідентифікаційна інформація, ролі, права доступу, а також історія авторизацій та інші параметри, допомагають системі впізнавати користувача, визначати його доступ до ресурсів та керувати безпекою даних.

Використання метаданих у системах авторизації дозволяє забезпечити ідентифікацію користувача, належну визначення його ролі та прав доступу, а також забезпечити зручність використання та персоналізований досвід. Важливою

складовою є також аудит та моніторинг, які дозволяють виявляти та реагувати на можливі загрози безпеки.

Отже, розуміння та використання метаданих користувача є необхідним для побудови надійних та ефективних систем авторизації, які забезпечують безпеку даних та задоволення від використання системи для всіх користувачів.

1.5 Переваги двоетапної генерації та перевірки токенів

Переваги двоетапної генерації та перевірки токенів у системі авторизації визначаються їхнім важливим внеском у підвищення рівня безпеки та забезпечення надійного доступу до інформації та ресурсів. Цей підхід включає в себе використання двох або більше факторів для підтвердження ідентичності користувача, що дозволяє зменшити ризик несанкціонованого доступу та захистити дані від різних видів атак. Враховуючи поширення кіберзлочинності та розвиток методів атак, перехід до двоетапної генерації та перевірки токенів стає важливим кроком для підвищення безпеки та захисту інформації [18].

Переваги двоетапної генерації та перевірки токенів у системі авторизації полягають у збільшенні рівня безпеки та зниженні ризику несанкціонованого доступу. Розглянемо детальніше ці переваги:

– покращена безпека;

Покращена безпека, яка випливає з використання двоетапної аутентифікації, полягає у використанні двох різних факторів для підтвердження ідентичності користувача. Цей підхід значно підвищує рівень безпеки авторизації та захисту конфіденційної інформації.

Першим фактором є те, що користувач знає, наприклад, його пароль. Це стандартний метод авторизації, який багато систем використовують для перевірки ідентичності користувача.

Другим фактором є те, що користувач має при собі, такий як спеціальний код, отриманий на мобільний телефон через смс або додаток для аутентифікації. Цей

фактор може бути різноманітним, включаючи фізичні пристрої, біометричні дані, або одноразові паролі.

Використання двох факторів аутентифікації робить процес входу більш безпечним. Навіть якщо хтось отримає доступ до одного з факторів (наприклад, зламавши пароль), це не надасть їм повного доступу до системи без наявності другого фактора (наприклад, фізичного пристрою або коду з мобільного телефону). Таким чином, двоетапна аутентифікація забезпечує додатковий шар захисту, що робить доступ до системи значно більш складним для несанкціонованих користувачів [19].

– захист від атак на перехоплення паролів;

Захист від атак на перехоплення паролів є однією з ключових переваг двоетапної аутентифікації. У звичайній парольній авторизації пароль може бути вкрадений або вгаданий, що створює ризик несанкціонованого доступу до системи. Однак, в двоетапній аутентифікації, навіть якщо зловмисник отримає доступ до пароля, він все одно не зможе отримати доступ до системи без додаткового токена.

Додатковий токен може приймати різні форми, такі як одноразовий код, отриманий на мобільний телефон через смс або спеціальний додаток для аутентифікації. Цей токен є необхідним для завершення процесу аутентифікації після введення пароля. Такий підхід ускладнює завдання зловмисників, оскільки навіть якщо вони отримають доступ до пароля, їм все одно буде потрібно мати доступ і до другого фактора, який є недоступним без фізичного доступу до користувача або його мобільного пристрою.

Отже, двоетапна аутентифікація надає значно вищий рівень захисту від атак на перехоплення паролів, збільшуючи складність процесу несанкціонованого доступу до системи.

– зменшення ризику фішингу;

Зменшення ризику фішингу є ще однією важливою перевагою двоетапної аутентифікації. Фішери зазвичай намагаються виманити паролі та інші конфіденційні дані, використовуючи хитрі підступи, такі як підроблені веб-сайти

або електронні листи. Однак, навіть якщо зловмисник здобуде пароль внаслідок атаки фішингу, він не матиме можливості отримати доступ до системи без другого фактора аутентифікації.

Другий фактор аутентифікації, який може бути, наприклад, одноразовий код, отриманий на мобільний телефон або інший пристрій, є недоступним для зловмисника. Це ускладнює завдання фішерів, оскільки навіть якщо вони отримають доступ до пароля, вони не зможуть використовувати його для несанкціонованого доступу до системи без додаткового токена.

Таким чином, двоетапна аутентифікація допомагає зменшити ризик успішних атак фішингу, забезпечуючи додатковий шар захисту навіть у випадку компрометації пароля.

– додатковий шар безпеки для критичних систем.

Додатковий шар безпеки, який надає двоетапна аутентифікація, особливо важливий для критичних систем або тих, що містять конфіденційну інформацію. Цей підхід є ефективним способом підвищення рівня безпеки та забезпечення додаткового шару захисту для важливих даних та ресурсів.

Критичні системи, такі як банківські системи, системи управління медичними даними або державні системи, містять чутливу та важливу інформацію, доступ до якої потрібно надавати тільки авторизованим користувачам. В таких випадках двоетапна аутентифікація стає обов'язковим елементом безпеки [21].

Перше, автентифікація за паролем або іншим фактором встановлює основний рівень ідентифікації користувача. Другий фактор, як правило, є щось, що користувач має фізично, наприклад, смартфон або спеціальний пристрій аутентифікації. Цей додатковий шар захисту робить процес аутентифікації більш надійним, оскільки навіть якщо зловмисник отримає доступ до пароля або основного фактора, він все одно не матиме можливості проникнути в систему без другого фактора аутентифікації.

Такий підхід допомагає запобігти несанкціонованому доступу до критичних даних та ресурсів, зменшуючи ризик порушення безпеки та негативних наслідків

для організації. В результаті двоетапна аутентифікація стає необхідним інструментом для забезпечення безпеки в критичних системах.

Двоетапна генерація та перевірка токенів є потужним інструментом для підвищення безпеки авторизації в інформаційних системах. Вона забезпечує покращену безпеку, захист від атак на перехоплення паролів, зменшує ризик фішингу та надає додатковий шар безпеки для критичних систем і конфіденційної інформації. Використання двоетапної аутентифікації є ключовим елементом у вирішенні проблем безпеки в інформаційних технологіях.

1.6 Висновки та постановка задачі

У першому розділі магістерської кваліфікаційної роботи було проведено детальний аналіз сучасних методів авторизації користувачів в інформаційних системах. Розглянуто різноманітні підходи до авторизації, виявлено їх переваги та недоліки. Досліджено принципи побудови безпечних систем авторизації та аналізовано існуючі вразливості в таких системах. Зосереджено увагу на ролі метаданих користувача у процесі авторизації та їх впливі на забезпечення безпеки даних.

Крім того, розглянуті переваги використання двоетапної генерації та перевірки токенів у системах авторизації, що дозволяє підвищити рівень безпеки та запобігти вразливостям. Висновки розділу визначили основні напрями подальших досліджень та завдання, які необхідно вирішити для підвищення ефективності та безпеки процесу авторизації користувачів.

Отже, на основі проведеного дослідження та аналізу було сформульовано ряд завдань для подальшої розробки:

- проаналізувати особливості механізмів збору та аналізу метаданих
- здійснити розробка архітектури та алгоритмів системи
- здійснити програмну реалізацію системи

2 ПРОЕКТУВАННЯ СИСТЕМИ ПІДВИЩЕННЯ ЗАХИЩЕНОСТІ АВТОРИЗАЦІЇ КОРИСТУВАЧІВ ВДОСКОНАЛЕНОЮ СИСТЕМОЮ З ДВОЕТАПНОЮ ГЕНЕРАЦІЄЮ ТА ПЕРЕВІРКОЮ ТОКЕНІВ ДОСТУПУ НА ОСНОВІ МЕТАДАНИХ КОРИСТУВАЧА

В даному розділі буде розглянуто особливості механізмів збору та аналізу метаданих, також буде розглянута розробка архітектури системи, розробка алгоритму двоетапної генерації та перевірки токенів та розробка алгоритму збору та аналізу метаданих

2.1 Особливості механізмів збору та аналізу метаданих

Механізми збору та аналізу метаданих відіграють вирішальну роль у підвищенні ефективності та безпеки сучасних інформаційних систем, особливо у контексті авторизації користувачів. Метадані, які можуть включати інформацію про час, місцеположення, тип пристрою, мережеві параметри тощо, надають глибший контекст для процесів ідентифікації та авторизації, тим самим збільшуючи здатність системи розпізнавати легітимні спроби доступу від неавторизованих атак [22].

Процес збору метаданих передбачає вилучення відомостей, що описують характеристики та поведінку користувачів під час використання системи. Для ефективної роботи цей процес повинен бути максимально автоматизованим та непомітним для користувачів, щоб не порушувати їхнє право на конфіденційність та не знижувати продуктивність системи. Важливим аспектом є також забезпечення достовірності та актуальності збираних даних, що вимагає регулярного оновлення механізмів збору метаданих у відповідь на зміни у технологіях та поведінці користувачів [23].

Збір метаданих є критично важливим процесом у структурі сучасних інформаційних систем, що забезпечує збирання відомостей, необхідних для аналізу, моніторингу, та підвищення безпеки. У контексті авторизації користувачів, збір метаданих дозволяє формувати детальне уявлення про контекст спроби

доступу, сприяючи тим самим підвищенню точності системи ідентифікації та авторизації. Нижче представлений детальний опис ключових аспектів процесу збору метаданих [25].

Метадані, які збираються в процесі авторизації, можуть бути класифіковані на декілька типів:

– технічні метадані: включають інформацію про пристрій користувача, операційну систему, тип браузера, ip-адресу, mac-адресу, конфігурацію пристрою тощо.

– поведінкові метадані: охоплюють дані про поведінку користувача, наприклад, час входу в систему, тривалість сесій, частоту запитів, послідовність дій на платформі.

– геолокаційні метадані: інформація про місцеположення користувача в момент спроби доступу, що може бути отримана через gps, wi-fi мережі, мобільні вишки тощо.

– мережеві метадані: відомості про мережеве середовище користувача, включаючи інформацію про інтернет-провайдера, тип підключення, використані мережеві порти, шифрування тощо.

Збір метаданих може здійснюватись за допомогою різноманітних методів і технологій:

– веб-трекери та кукі: використовуються для збору інформації про поведінку користувача на веб-сайтах.

– системні журнали: автоматичне логування системних подій та дій користувача.

– сенсорні дані: збір інформації з фізичних сенсорів пристрою, таких як gps, акселерометр.

– api для збору метаданих: інтерфейси програмування додатків, які дозволяють збирати технічну інформацію про пристрої та програмне забезпечення.

Під час збору метаданих необхідно строго дотримуватися етичних норм та правових вимог, зокрема щодо захисту персональних даних користувачів. Важливо забезпечити:

– прозорість: користувачі повинні бути поінформовані про збір та використання метаданих.

– дозвіл: збір деяких типів метаданих може вимагати попередньої згоди користувача.

– конфіденційність: збереження метаданих повинно відбуватись у зашифрованому вигляді для запобігання несанкціонованому доступу.

Ефективний збір та обробка метаданих є фундаментом для розробки безпечних та надійних систем авторизації. Проте, успішна реалізація цього процесу вимагає не лише технічного досвіду, а й глибокого розуміння правових та етичних аспектів, пов'язаних з обробкою даних користувачів.

Аналіз метаданих полягає у виявленні закономірностей, що можуть вказувати на легітимні або нелегітимні спроби доступу до системи. Сучасні методи аналізу часто базуються на алгоритмах машинного навчання та штучного інтелекту, здатних адаптуватися до постійно змінюваних моделей поведінки користувачів та загроз. Це дозволяє системі не лише реагувати на відомі типи атак, а й прогнозувати та запобігати новим загрозам. Ключовим фактором є здатність системи до самонавчання та адаптації, що вимагає постійного оновлення наборів даних для навчання та оптимізації алгоритмів [26].

Аналіз метаданих є важливим етапом у процесі забезпечення безпеки інформаційних систем, особливо в контексті авторизації користувачів. Цей процес включає в себе обробку та інтерпретацію зібраних метаданих з метою ідентифікації закономірностей, що можуть вказувати на нормальну або аномальну поведінку. Далі представлено детальний огляд ключових аспектів аналізу метаданих.

Аналіз метаданих спрямований на досягнення кількох ключових цілей:

– виявлення аномалій: ідентифікація незвичайної поведінки, яка може вказувати на спроби несанкціонованого доступу або інші безпекові інциденти.

– підвищення точності авторизації: використання додаткового контексту для покращення систем ідентифікації та авторизації користувачів.

– оптимізація безпекових політик: адаптація стратегій безпеки на основі отриманих даних про поведінку користувачів та загроз.

Для аналізу метаданих застосовуються різноманітні методики та технології:

– статистичний аналіз: використання статистичних методів для виявлення відхилень від звичайних патернів поведінки.

– машинне навчання: застосування алгоритмів машинного навчання для ідентифікації складних закономірностей та аномалій на основі великих обсягів даних.

– глибоке навчання: використання нейронних мереж для обробки особливо складних даних, що можуть включати зображення, аудіо та відео метадані.

– поведінковий аналіз: аналіз поведінкових метаданих для виявлення шаблонів, характерних для легітимної або маліційної діяльності.

При аналізі метаданих існують певні виклики, які потребують уваги:

– обробка великих обсягів даних: збільшення обсягів метаданих вимагає від систем високої продуктивності та ефективного управління даними.

– забезпечення конфіденційності: необхідність захисту персональних даних користувачів при аналізі метаданих.

– адаптація до нових загроз: швидке змінення технологій та методів атак вимагає постійної адаптації алгоритмів аналізу.

2.2 Розробка архітектури системи

Розробка архітектури системи є ключовим етапом у створенні ефективних та безпечних інформаційних систем, зокрема у контексті підвищення захищеності авторизації користувачів через вдосконалену систему з двоетапною генерацією та перевіркою токенів доступу на основі метаданих користувача. Архітектура системи слугує фундаментом, на якому базуються всі ключові компоненти та процеси,

забезпечуючи необхідну гнучкість, масштабованість, безпеку, та високу доступність системи. Далі представлений огляд основних аспектів, які необхідно врахувати при розробці архітектури системи.

Компоненти архітектури системи забезпечують основу для її функціонування, взаємодію між різними частинами та реалізацію ключових процесів. В контексті системи для підвищення захищеності авторизації користувачів через вдосконалену систему з двоетапною генерацією та перевіркою токенів доступу на основі метаданих користувача, наступні компоненти відіграють вирішальну роль:

1. Сервер авторизації

Цей компонент є центральним у системі авторизації. Він відповідає за обробку запитів на авторизацію, верифікацію користувачів, генерацію, видачу, та перевірку токенів доступу. Сервер авторизації повинен забезпечувати високий рівень безпеки, маючи засоби для захисту від різноманітних атак і спроб несанкціонованого доступу.

2. База даних метаданих користувачів

Цей компонент зберігає в собі важливу інформацію про користувачів, яка використовується для процесів авторизації, включаючи метадані, необхідні для генерації та перевірки токенів доступу. База даних повинна забезпечувати швидкий доступ до даних при високих рівнях безпеки та конфіденційності.

3. Модуль збору метаданих

Цей компонент відповідає за збір метаданих з різноманітних джерел, включаючи дані про пристрої користувачів, їхнє місцеположення, мережеві параметри тощо. Зібрані метадані використовуються для додаткової верифікації користувачів і підвищення рівня безпеки системи.

Модуль збору метаданих в архітектурі системи авторизації відіграє ключову роль у забезпеченні додаткового рівня безпеки, аналізуючи поведінку користувача та контекст його дій. Цей компонент відповідає за автоматичний збір, нормалізацію, та передачу метаданих від користувачів та їхніх пристроїв до

системи. Далі наведено детальний опис основних аспектів модуля збору метаданих [28].

Модуль автоматично збирає метадані з різноманітних джерел, включаючи:

- дані пристрою: модель, операційна система, версія пз, унікальні ідентифікатори (наприклад, imei на мобільних телефонах, mac-адреса мережевих карт).
- мережеві дані: ip-адреса, інформація про wi-fi мережі або мобільні мережі, які використовуються для доступу.
- геолокаційні дані: місцеположення користувача, отримане через gps, ip-геолокацію, або інші методи.
- поведінкові дані: час доступу до системи, тривалість сесії, типи виконуваних дій.

Після збору даних модуль виконує їх нормалізацію, перетворюючи сиру інформацію в стандартизований формат, що спрощує подальший аналіз. Це може включати конвертацію часових зон, уніфікацію форматів даних, видалення або анонімізацію персональної інформації.

Зібрані та оброблені метадані передаються в інші компоненти системи, зокрема в модуль аналізу метаданих та базу даних, для подальшого використання в процесі авторизації.

Цей компонент аналізує зібрані метадані для виявлення закономірностей, які можуть свідчити про легітимність чи підозрілість спроб доступу. Використовуючи технології машинного навчання та штучного інтелекту, модуль аналізу дозволяє системі адаптуватися до нових загроз і підвищити ефективність процесів авторизації.

Модуль аналізує поведінкові патерни користувачів, засновані на метаданих, таких як час доступу до системи, геолокація, тип пристрою, та інші. Це дозволяє ідентифікувати відхилення від звичної поведінки, які можуть свідчити про спроби взлому або фішингу.

Застосування алгоритмів машинного навчання та статистичного аналізу для виявлення аномальних дій або змін у поведінці користувачів, які можуть вказувати на безпекові загрози.

Використання технік адаптивного навчання для постійного оновлення моделей поведінки користувачів на основі нових даних, що дозволяє системі ефективно адаптуватися до змін у поведінкових патернах та нових методів атак.

Здатність обробляти великі обсяги даних у реальному часі є критичною для ефективного функціонування модуля аналізу метаданих, вимагаючи використання технологій великих даних та оптимізованої обчислювальної інфраструктури.

Забезпечення високого рівня безпеки при зберіганні та обробці метаданих, включаючи застосування шифрування, контролю доступу, та інших механізмів захисту конфіденційності користувачів [29].

Інтерфейс користувача (UI) дозволяє користувачам взаємодіяти з системою. Це може включати веб-інтерфейси, мобільні додатки або інші клієнтські додатки. UI повинен бути інтуїтивно зрозумілим, забезпечувати легкий доступ до необхідних функцій та високий рівень захисту при передачі даних.

Мережева інфраструктура забезпечує зв'язок між компонентами системи та її користувачами. Це включає сервери, мережеві протоколи, засоби шифрування та захисту даних при передачі.

Кожен з цих компонентів відіграє важливу роль у загальній архітектурі системи та її здатності ефективно виконувати задачі, пов'язані з авторизацією користувачів. Правильне проектування та взаємодія між цими компонентами є ключем до створення безпечної, надійної та масштабованої системи.

2.3 Розробка алгоритму збору та аналізу метаданих

Розробка алгоритму збору та аналізу метаданих для системи авторизації вимагає інтеграції передових технік обробки даних, машинного навчання та кібербезпеки. Цей процес включає збір релевантних метаданих користувачів,

обробку цих даних для виявлення закономірностей та аномалій, та використання отриманих висновків для покращення системи авторизації (рис. 2.1).

Розберемо алгоритм даний алгоритм по крокам.

Крок 1: Початок

Процес розпочинається. Це стандартний перший крок будь-якого алгоритму або процедури.

Крок 2: Збір інформації про браузер користувача

Збір даних, які браузер відправляє до сервера, включно з версією браузера, типом і версією операційної системи, встановленими плагінами, розширеннями тощо.

Крок 3: Збір інформації про місцезнаходження користувача

Визначення фізичного місцезнаходження користувача за допомогою GPS, Wi-Fi або інших методів геолокації.

Крок 4: Збір інформації про IP адресу користувача

Фіксація IP адреси, з якої користувач здійснює підключення до системи, що може бути використано для визначення місцезнаходження та провайдера інтернет-послуг.

Крок 5: Збір інформації про часовий пояс користувача

Реєстрація часового поясу, в якому перебуває користувач, що може допомогти у верифікації його ідентичності.

Крок 6: Збір інформації про системні дані пристрою користувача

Збір даних про апаратне забезпечення пристрою користувача, включно з типом пристрою, наявністю певних сенсорів, роздільною здатністю екрану тощо.

Крок 7: Конкатенація даних

Об'єднання всіх зібраних метаданих в єдину структуру даних для подальшого аналізу.

Крок 8: Перевірка умови

Перевіряється, чи зібрано більше ніж 50% запланованих даних. Якщо ні, то процес переходить до виведення помилки; якщо так, то відбувається перехід до аналізу метаданих.

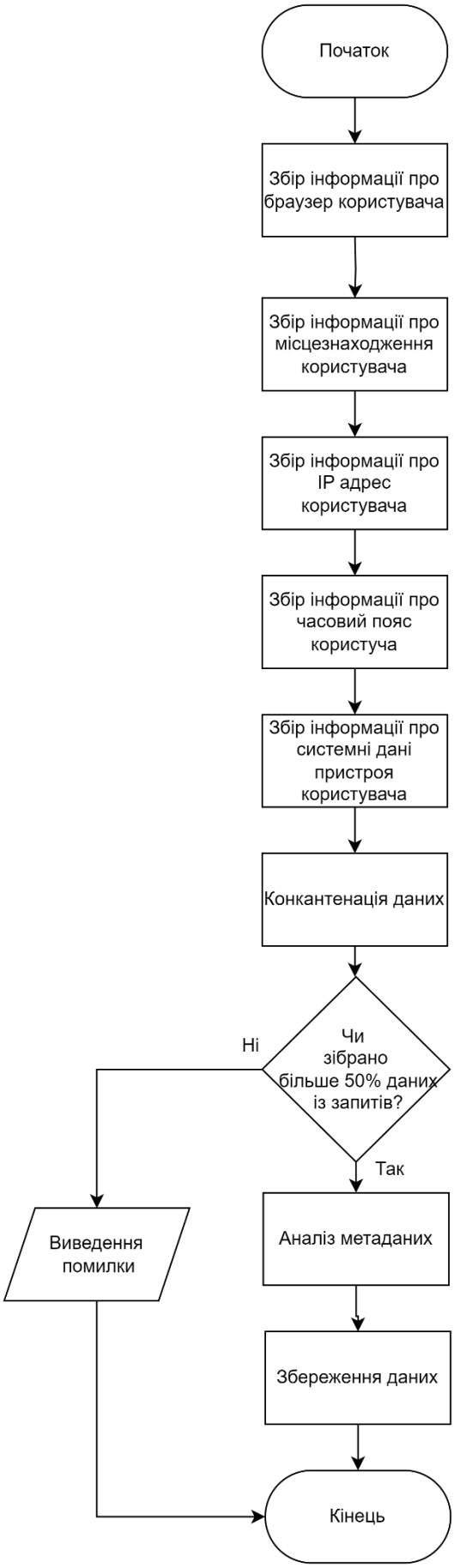


Рисунок 2.1 - Алгоритм збору та аналізу метаданих

Крок 9: Виведення помилки

У разі, якщо умова попереднього кроку не виконується, система виводить помилку. Це може вказувати на неповноту даних або помилку збору даних.

Крок 10: Аналіз метаданих

Виконується обробка зібраних метаданих алгоритмами для виявлення патернів поведінки, аномалій, тощо.

Крок 11: Збереження даних

Зібрані та проаналізовані метадані зберігаються для подальшого використання або архівування.

Крок 12: Кінець

Процес аналізу метаданих завершується.

Цей алгоритм може бути частиною більшого процесу авторизації або системи виявлення вторгнень, де метадані використовуються для верифікації ідентичності користувача та виявлення потенційних загроз.

2.4 Розробка алгоритму двоетапної генерації та перевірки токенів

Алгоритми генерації та перевірки токенів є важливою складовою систем авторизації та контролю доступу. Вони забезпечують засіб для безпечного представлення та перевірки прав доступу користувача.

Розробимо даний алгоритм (рис. 2.2) та покроково розберемо його.

Крок 1: Початок

Початкова точка алгоритму, де запускається процес генерації та перевірки токенів.

Крок 2: Запит користувача

Користувач ініціює процес, зробивши запит на доступ до ресурсу.

Крок 3: Збір метаданих

Система збирає метадані користувача, які можуть включати інформацію про його пристрій, локацію, час запиту тощо.

Крок 4: Аналіз метаданих

Зібрані метадані аналізуються на предмет відповідності встановленим критеріям безпеки.

Крок 5: Генерація токену на основі метаданих

На основі результатів аналізу метаданих генерується токен, який кодує відомості про запит користувача та його дозволи.

Крок 6: Генерація токену доступу на основі токену з метаданими

Створюється токен доступу, що містить або посилається на токен з метаданими.

Крок 7: Відправлення токену доступу користувачу

Сгенерований токен доступу відправляється користувачу.

Крок 8: Авторизований запит з токеном доступу від користувача

Користувач пред'являє токен доступу як частину свого авторизованого запиту на доступ до ресурсу.

Крок 9: Відправлення повідомлення про помилку

Якщо в процесі перевірки виникає помилка, користувачу відправляється повідомлення про неї.

Крок 10: Перевірка валідності токену

Система перевіряє токен на валідність, що може включати перевірку цілісності, терміну дії та інші параметри безпеки.

Крок 11: Дешифрування метаданих користувача

Якщо токен валідний, система дешифрує метадані користувача, закодовані в токені.

Крок 12: Метадані збігаються?

Система перевіряє, чи збігаються метадані, отримані з токену, з метаданими запиту користувача.

Крок 13: Обробка запиту користувача

Якщо метадані збігаються, система обробляє запит користувача, надаючи йому доступ до запитаних ресурсів.

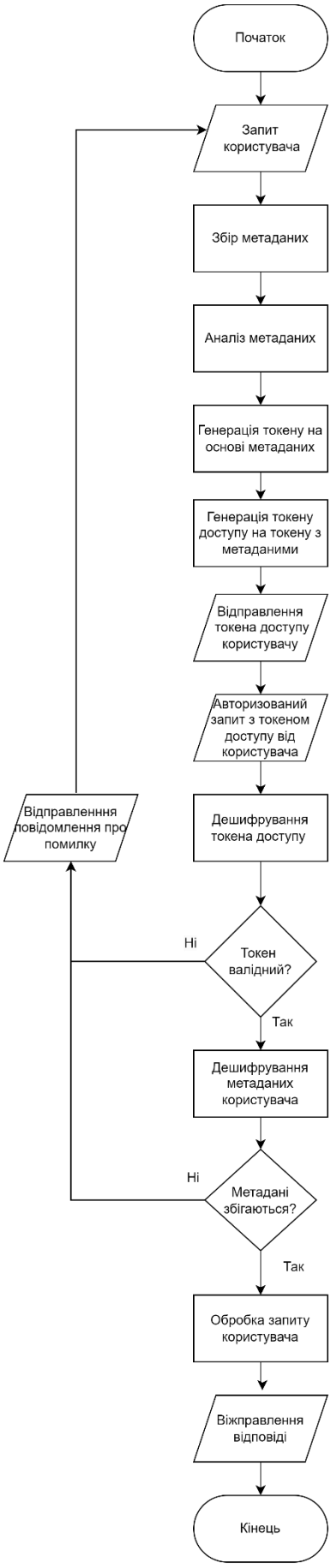


Рисунок 2.1 - Алгоритм двоетапної генерації та перевірки токенів

Крок 14: Відправлення відповіді

Система відправляє відповідь користувачу, яка може містити результати запиту або інформацію про статус обробки.

Крок 15: Кінець

Завершення алгоритму.

Цей алгоритм реалізує двоетапний процес, в якому спочатку створюється токен на основі метаданих, а потім цей токен використовується для генерації токену доступу, що надається користувачу. Вся ця процедура забезпечує додатковий рівень безпеки для системи, оскільки вона використовує детальну інформацію про користувача для створення валідного токену доступу.

2.5 Висновки до розділу.

У цій роботі було успішно розроблено алгоритм двоетапної генерації та перевірки токенів, який покликаний зміцнити безпеку процесів авторизації в системах управління доступом. Цей алгоритм був заснований на використанні метаданих для створення динамічних токенів, які не тільки здійснюють аутентифікацію особи, але й забезпечують високий рівень захисту від несанкціонованого доступу.

Протягом роботи були визначені ключові етапи розробки, від визначення вимог до системи до тестування та верифікації. Особлива увага була приділена забезпеченню безпеки на кожному кроці, від збору метаданих до генерації та передачі токенів.

Завдяки вдосконаленню алгоритму збору метаданих та їх аналізу, було досягнуто високої точності у виявленні аномалій, що є суттєвим покращенням у контексті превентивних заходів безпеки. Водночас, увага до деталей та впровадження найкращих практик кібербезпеки забезпечили високий рівень захисту персональних даних користувачів, відповідаючи сучасним нормативним стандартам.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ

В даному розділі буде здійснено обгрунтування вибору програмної мови реалізації та середовища розробки, після чого буде здійснена програмна реалізація додатку та програмна реалізація інтерфейсу додатку, після чого буде здійснено тестування та аналіз роботи реалізованого додатку.

3.1 Обгрунтування вибору програмної мови реалізації та середовища розробки

Вибір програмної мови реалізації та середовища розробки для розробки вдосконаленої системи авторизації користувачів з двоетапною генерацією та перевіркою токенів доступу на основі метаданих користувача обумовлений рядом технічних і практичних переваг. У контексті даного проекту було обрано Node.js як основну мову програмування і середовище розробки, що забезпечує оптимальне поєднання продуктивності, масштабованості та зручності використання.

Node.js є середовищем виконання JavaScript, яке дозволяє виконувати код на стороні сервера. Однією з ключових переваг Node.js є його асинхронна, неблокуюча модель вводу-виводу, що дозволяє ефективно обробляти велику кількість одночасних запитів. Це особливо важливо для систем авторизації, які повинні мати високу продуктивність та здатність масштабуватися під час обробки запитів від великої кількості користувачів [32].

Вибір Node.js також обумовлений його широкою підтримкою спільноти та великою кількістю доступних модулів і пакетів, зокрема через npm (Node Package Manager). Це значно полегшує процес розробки та інтеграції додаткових функцій, необхідних для реалізації двоетапної генерації та перевірки токенів доступу. Використання готових рішень та бібліотек дозволяє зосередитися на розробці унікальних аспектів системи, що підвищують її безпеку.

Сучасна екосистема розробки, яку підтримує Node.js, є ще одним вагомим аргументом на користь його вибору. Node.js активно підтримує новітні стандарти JavaScript, включаючи ECMAScript 6 і новіші версії. Це дозволяє використовувати

сучасні можливості мови програмування, такі як стрілочні функції, деструктурування, класи та інші. Це не тільки покращує якість коду, але й сприяє підвищенню його читабельності та підтримуваності.

Бібліотека Passport.js є провідним інструментом для реалізації автентифікації в Node.js-додатках. Вона забезпечує модульну, гнучку та зручну платформу для інтеграції різноманітних методів автентифікації, таких як локальні стратегії, OAuth, OpenID Connect та багато інших. Це робить Passport.js надзвичайно корисною для проектів, які вимагають надійного та безпечного управління автентифікацією користувачів.

Однією з основних переваг Passport.js є його легкість та модульність. В основі Passport.js лежить принцип, згідно з яким кожна стратегія автентифікації є окремим модулем, що може бути доданий або видалений з проекту без значного впливу на інші частини системи. Це дозволяє розробникам легко інтегрувати нові методи автентифікації або замінювати існуючі. Наприклад, можна використовувати локальну стратегію для автентифікації через ім'я користувача та пароль, а також додати підтримку OAuth для автентифікації через соціальні мережі, такі як Facebook або Google.

Passport.js підтримує понад 500 стратегій автентифікації, що робить його надзвичайно гнучким. Незалежно від специфічних вимог до автентифікації, швидше за все, існує стратегія, яка задовольнить ці вимоги. Для особливих випадків можна створювати власні стратегії, що забезпечує ще більшу гнучкість та контроль над процесом автентифікації [33].

Інтеграція Passport.js з існуючими додатками на Node.js є досить простою та прямолінійною. Passport.js добре поєднується з популярними фреймворками для веб-розробки, такими як Express.js. Це дозволяє легко додати автентифікацію до додатку, використовуючи зрозумілий та лаконічний API. Наприклад, для додавання локальної автентифікації достатньо визначити стратегію, налаштувати маршрути для входу та виходу, а також забезпечити збереження стану автентифікації через сесії або токени.

Безпека є ще однією важливою перевагою Passport.js. Бібліотека надає розробникам всі необхідні інструменти для створення безпечних систем автентифікації. Зокрема, вона підтримує використання HTTPS для шифрування даних, які передаються між клієнтом і сервером, що захищає конфіденційну інформацію від перехоплення. Крім того, Passport.js дозволяє легко реалізувати захист від атак, таких як CSRF (Cross-Site Request Forgery) і XSS (Cross-Site Scripting), шляхом налаштування відповідних middleware.

Passport.js також забезпечує зручний механізм управління сесіями користувачів. Використовуючи middleware, розробники можуть зберігати стан автентифікації користувачів між різними запитами, що дозволяє реалізувати функціональність "запам’ятати мене" або інші подібні функції. Це значно покращує користувацький досвід, забезпечуючи зручний та безперервний доступ до захищених ресурсів додатку [35].

Крім того, Passport.js добре масштабується та легко інтегрується з іншими інструментами та бібліотеками для побудови більш комплексних систем. Наприклад, можна використовувати Passport.js разом з бібліотеками для управління токенами доступу, такими як JSON Web Tokens (JWT), або для реалізації двофакторної автентифікації.

Однією з важливих переваг Node.js є його зручність для full-stack розробки. Оскільки JavaScript використовується як на стороні сервера, так і на стороні клієнта, це дозволяє розробникам використовувати одну мову для створення всього додатку. Такий підхід спрощує процес розробки, знижує ймовірність помилок, пов’язаних з переходом між різними мовами програмування, та полегшує підтримку та розвиток проекту в майбутньому.

Обґрунтування вибору середовища розробки WebStorm для реалізації вдосконаленої системи авторизації користувачів з двоетапною генерацією та перевіркою токенів доступу на основі метаданих користувача спирається на декілька ключових факторів, які забезпечують високу ефективність, продуктивність і зручність для розробників.

WebStorm, розроблений компанією JetBrains, є одним з провідних інтегрованих середовищ розробки (IDE) для JavaScript та його екосистеми. Основною причиною вибору WebStorm є його потужний функціонал, орієнтований на розробку сучасних веб-додатків. Однією з найбільших переваг WebStorm є його інтелектуальний редактор коду, який забезпечує контекстуальну підказку, автозавершення, навігацію по коду, а також перевірку та виправлення помилок у режимі реального часу. Це дозволяє розробникам значно підвищити продуктивність і зосередитися на написанні якісного та оптимізованого коду.

WebStorm також підтримує всі сучасні технології та фреймворки для веб-розробки, включаючи Node.js, React, Angular, Vue.js, і багато інших. Це забезпечує розробникам гнучкість у виборі інструментів і технологій, які найкраще відповідають вимогам проекту. Зокрема, для проекту, що включає двоетапну генерацію та перевірку токенів доступу, підтримка Node.js є критично важливою. WebStorm пропонує повну інтеграцію з Node.js, включаючи підтримку npm, налагодження, тестування та управління проектом.

Ще однією важливою перевагою WebStorm є його інструменти для роботи з версіями систем контролю, такими як Git, Mercurial та інші. WebStorm забезпечує зручний інтерфейс для керування репозиторіями, що дозволяє легко виконувати операції комітів, злиття, ревертів та інших дій, пов’язаних з контролем версій. Це є важливим аспектом для командної розробки та підтримки цілісності коду під час роботи над складними проектами.

Інтеграція з системами тестування є ще однією вагомою причиною вибору WebStorm. Середовище розробки підтримує такі фреймворки для тестування, як Mocha, Jest, Karma та інші. Це дозволяє розробникам ефективно проводити юніт-тестування, інтеграційне тестування та інші види тестування, що є критично важливим для забезпечення якості та надійності програмного забезпечення [36].

WebStorm також відомий своїми потужними інструментами для налагодження коду. Завдяки інтеграції з браузерами та можливості віддаленого налагодження, розробники можуть легко відстежувати та виправляти помилки в

коді, що значно скорочує час на вирішення проблем і покращує загальну ефективність розробки.

Важливим аспектом є також підтримка WebStormом інструментів для аналізу та рефакторингу коду. Середовище надає потужні можливості для оптимізації коду, що включає автоматизовані рефакторинги, перевірку коду на відповідність стандартам та пошук потенційних проблем у коді. Це сприяє підтримці високої якості коду та полегшує його подальше обслуговування [38].

Загалом, вибір WebStorm як середовища розробки обумовлений його потужним функціоналом, зручністю для розробників, підтримкою сучасних технологій та інструментів для тестування, налагодження та рефакторингу коду. Ці особливості роблять WebStorm ідеальним вибором для реалізації складних і надійних систем авторизації, що вимагають високої продуктивності та якості розробки.

3.2 Програмна реалізація додатку

Програмна реалізація додатку для підвищення захищеності авторизації користувачів із використанням вдосконаленої системи з двоетапною генерацією та перевіркою токенів доступу на основі метаданих користувача включає кілька основних етапів: проектування архітектури, налаштування середовища розробки, реалізація серверної та клієнтської частин, інтеграція системи автентифікації та забезпечення безпеки.

Архітектура додатку базується на модель-клієнт-серверній моделі. Серверна частина реалізована з використанням Node.js, що забезпечує асинхронне оброблення запитів і високу продуктивність. Клієнтська частина додатку може бути реалізована за допомогою сучасних фреймворків, таких як React або Angular, що забезпечують динамічний і інтерактивний користувацький інтерфейс.

Серверна частина реалізована на основі Express.js, що дозволяє швидко створювати і налаштовувати маршрути та обробники запитів.

Реалізація серверної частини додатку для підвищення захищеності авторизації користувачів з використанням вдосконаленої системи з двоетапною генерацією та перевіркою токенів доступу на основі метаданих користувача починається з налаштування середовища розробки та створення основних компонентів додатку. Основні етапи включають налаштування серверу, маршрутизацію, інтеграцію Passport.js для автентифікації, реалізацію двоетапної генерації токенів та забезпечення безпеки.

Далі створюємо файл `server.js`, який буде основним файлом нашого серверу, і налаштовуємо базову конфігурацію Express.js

```
const express = require('express');
const bodyParser = require('body-parser');
const session = require('express-session');
const passport = require('passport');
const app = express();
// Middleware для обробки JSON-запитів
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({ extended: true }));
// Налаштування сесій
app.use(session({
  secret: 'your_secret_key',
  resave: false,
  saveUninitialized: false,
}));
// Ініціалізація Passport.js
app.use(passport.initialize());
app.use(passport.session());
// Налаштування маршрутизації
const authRoutes = require('./routes/auth');
app.use('/auth', authRoutes);
// Запуск серверу
const PORT = process.env.PORT || 3000;
app.listen(PORT, () => {
  console.log(`Server is running on port ${PORT}`);
});
```

Далі створюємо директорію routes і файл auth.js всередині неї для маршрутизації запитів, пов'язаних з автентифікацією.

```
const express = require('express');
const passport = require('passport');
const LocalStrategy = require('passport-local').Strategy;
const router = express.Router();
passport.use(new LocalStrategy(
  function(username, password, done) {
    const user = users.find(u => u.username === username);
    if (!user) {
      return done(null, false, { message: 'Incorrect username.' });
    }
    if (user.password !== password) {
      return done(null, false, { message: 'Incorrect password.' });
```

```
    }
    return done(null, user);
  }
));
// Серіалізація та десеріалізація користувачів
passport.serializeUser(function(user, done) {
  done(null, user.id);
});
passport.deserializeUser(function(id, done) {
  const user = users.find(u => u.id === id);
  done(null, user);
});
// Маршрут для входу
router.post('/login', passport.authenticate('local', {
  successRedirect: '/auth/success',
  failureRedirect: '/auth/failure'
}));
// Маршрут для успішного входу
router.get('/success', (req, res) => {
  res.send('Login successful!');
})
// Маршрут для невдалого входу
router.get('/failure', (req, res) => {
  res.send('Login failed!');
});

module.exports = router;
```

У вищезазначеному коді вже інтегровано базову локальну стратегію автентифікації за допомогою Passport.js. Важливі кроки включають налаштування стратегії, серіалізацію та десеріалізацію користувачів.

Для реалізації двоетапної генерації та перевірки токенів доступу, використовуємо JSON Web Tokens (JWT).

Додаємо генерацію і перевірку токенів у файл auth.js.

```
const jwt = require('jsonwebtoken');
const secretKey = 'your_jwt_secret_key';
// Маршрут для генерації токену
router.post('/token', (req, res) => {
  const user = req.user;
  const payload = {
    id: user.id,
    username: user.username,
  };
  const token = jwt.sign(payload, secretKey, { expiresIn: '1h' });
  res.json({ token });
});
// Middleware для перевірки токену
const verifyToken = (req, res, next) => {
  const token = req.headers['authorization'];
  if (!token) {
    return res.status(403).send('Token is required');
  }
  jwt.verify(token, secretKey, (err, decoded) => {
```

```
    if (err) {
      return res.status(401).send('Invalid token');
    }
    req.user = decoded;
    next();
  });
};router.get('/protected', verifyToken, (req, res) => {
  res.send('This is a protected route');
});
```

Для забезпечення безпеки, додаємо SSL/TLS підтримку, налаштовуємо захист від CSRF і XSS атак. Використовуємо модуль https для налаштування HTTPS сервера.

```
const fs = require('fs');
const https = require('https');
const privateKey = fs.readFileSync('path/to/your/private.key', 'utf8');
const certificate = fs.readFileSync('path/to/your/certificate.crt', 'utf8');
const credentials = { key: privateKey, cert: certificate };
const httpsServer = https.createServer(credentials, app);
httpsServer.listen(PORT, () => {
  console.log(`HTTPS Server is running on port ${PORT}`);
});
```

Додаємо middleware для захисту від CSRF

```
const csurf = require('csurf');

app.use(csurf());

app.use((req, res, next) => {
  res.locals.csrfToken = req.csrfToken();
  next();
});
```

Це дозволить додати CSRF токени у форми та забезпечити захист від CSRF атак.

Реалізація серверної частини додатку на основі Express.js включає налаштування серверу, маршрутизацію, інтеграцію Passport.js для автентифікації, реалізацію двоетапної генерації та перевірки токенів, а також забезпечення безпеки додатку. Використання сучасних інструментів та бібліотек, таких як Express.js, Passport.js та JSON Web Tokens, дозволяє створити надійний і безпечний додаток для авторизації користувачів.

3.3 Програмна реалізація інтерфейсу додатку

Програмна реалізація інтерфейсу додатку включає створення динамічного та інтерактивного користувацького інтерфейсу, що дозволяє користувачам зручно взаємодіяти з системою. Для цього використовуються сучасні фреймворки, такі як React, які забезпечують високу продуктивність і зручність розробки.

Створюємо новий проект за допомогою Create React App:

```
npx create-react-app secure-auth-client
cd secure-auth-client
```

Проект матиме наступну структуру:

```
secure-auth-client/
  ├── public/
  ├── src/
  |   ├── components/
  |   |   ├── LoginForm.js
  |   |   ├── RegisterForm.js
  |   |   ├── ProtectedComponent.js
  |   |   └── Notification.js
  |   ├── App.js
  |   ├── index.js
  |   └── api.js
  ├── package.json
  └── README.md
```

Створимо файл App.js, цей файл буде основним компонентом додатку, що містить маршрути та загальну структуру додатку:

```
import React from 'react';
import { BrowserRouter as Router, Route, Switch } from 'react-router-dom';
import LoginForm from './components/LoginForm';
import RegisterForm from './components/RegisterForm';
import ProtectedComponent from './components/ProtectedComponent';
import Notification from './components/Notification';

function App() {
  return (
    <Router>
      <div className="App">
        <Notification />
        <Switch>
          <Route path="/login" component={LoginForm} />
          <Route path="/register" component={RegisterForm} />
          <Route path="/protected" component={ProtectedComponent} />
        </Switch>
      </div>
    </Router>
  );
}

export default App;
```

Створимо компонент який відповідає за форму входу користувача.

```
import React, { useState } from 'react';
import { useHistory } from 'react-router-dom';
import axios from 'axios';
function LoginForm() {
  const [username, setUsername] = useState('');
  const [password, setPassword] = useState('');
  const history = useHistory();

  const handleLogin = async (event) => {
    event.preventDefault();
    try {
      const response = await axios.post('/auth/login', { username, password });
      if (response.data.success) {
        history.push('/protected');
      } else {
        alert('Login failed');
      }
    } catch (error) {
      alert('An error occurred');
    }
  };
  return (
    <form onSubmit={handleLogin}>
      <div>
        <label>Username:</label>
        <input type="text" value={username} onChange={(e) => setUsername(e.target.value)} />
      </div>
      <div>
        <label>Password:</label>
        <input type="password" value={password} onChange={(e) => setPassword(e.target.value)} /
      </div>
      <button type="submit">Login</button>
    </form>
  );
}
export default LoginForm;
```

Створимо компонент який відповідає за форму реєстрації користувача.

```
import React, { useState } from 'react';
import axios from 'axios';
function RegisterForm() {
  const [username, setUsername] = useState('');
  const [password, setPassword] = useState('');
  const handleRegister = async (event) => {
    event.preventDefault();
    try {
      const response = await axios.post('/auth/register', { username, password });
      if (response.data.success) {
        alert('Registration successful');
      } else {
        alert('Registration failed');
      }
    } catch (error) {
      alert('An error occurred');
    }
  };

  return (
```

```
      <form onSubmit={handleRegister}>
       <div>
        <label>Username:</label>
        <input type="text" value={username} onChange={(e) => setUsername(e.target.value)} />
       </div>
       <div>
        <label>Password:</label>
        <input type="password" value={password} onChange={(e) => setPassword(e.target.value)}
/>
       </div>
       <button type="submit">Register</button>
      </form>
     );
    }
    export default RegisterForm;
```

ProtectedComponent.js, цей компонент відповідає за захищений контент, доступний тільки після автентифікації:

```
    import React, { useEffect, useState } from 'react';
    import axios from 'axios';
    function ProtectedComponent() {
     const [message, setMessage] = useState('');
     useEffect(() => {
      const fetchData = async () => {
       try {
        const response = await axios.get('/auth/protected', {
         headers: {
          'Authorization': `Bearer ${localStorage.getItem('token')}`
         }
        });
        setMessage(response.data.message);
       } catch (error) {
        setMessage('You are not authorized to view this content');
       }
      };

      fetchData();
     }, []);
     return (
      <div>
       <h1>Protected Content</h1>
       <p>{message}</p>
      </div>
     );
    }
    export default ProtectedComponent;
```

Файл api.js містить функції для взаємодії з сервером, такі як реєстрація, вхід, отримання токену та перевірка токену:

```
    import axios from 'axios';
    export const register = async (username, password) => {
     return await axios.post('/auth/register', { username, password });
    };

    export const login = async (username, password) => {
     return await axios.post('/auth/login', { username, password });
```

```
};

export const getProtectedData = async (token) => {
  return await axios.get('/auth/protected', {
    headers: {
      'Authorization': `Bearer ${token}`
    }
  });
};
```

Для забезпечення безпеки на клієнтській стороні використовуємо наступні заходи:

Валідація даних на стороні клієнта: Перевірка введених даних перед відправкою на сервер для зменшення навантаження та підвищення безпеки.

Використання HTTPS: Забезпечення шифрування даних, що передаються між клієнтом і сервером. Захист від CSRF: Використання токенів CSRF для захисту від атак CSRF. Захист від XSS: Санітизація введених даних для захисту від атак XSS.

Реалізація інтерфейсу додатку включає створення компонентів для входу, реєстрації, захищеного контенту та сповіщень, а також інтеграцію з сервером для взаємодії з API. Використання сучасного фреймворку React забезпечує динамічний та інтерактивний користувацький інтерфейс, що дозволяє зручно працювати з системою авторизації. Забезпечення безпеки на клієнтській стороні є важливим аспектом, який допомагає захистити дані користувачів і забезпечити надійну роботу додатку.

3.4 Тестування та аналіз роботи реалізованого додатку

Тестування та аналіз роботи реалізованого додатку є важливими етапами в забезпеченні його функціональності, надійності та безпеки.

Початок роботи з розробленим додатком починається з сторінки реєстрації користувача (рис. 3.1).

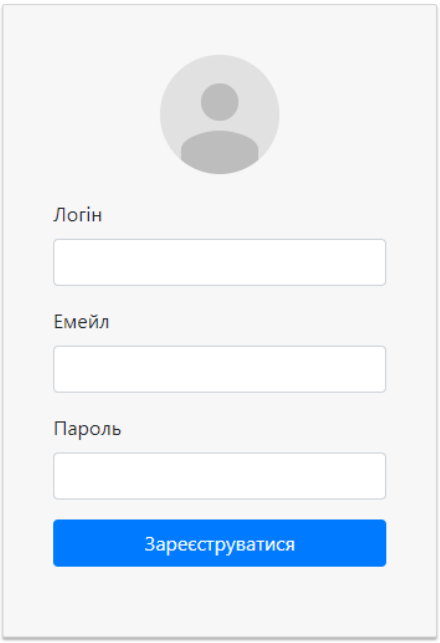


Рисунок 3.1 - Сторінка реєстрації користувача

Для реєстрації користувача, користуваачу потрібно ввести відповідіні дані в вілповідіні поля та натиснути кнопку зареєструватися (рис. 3.2).

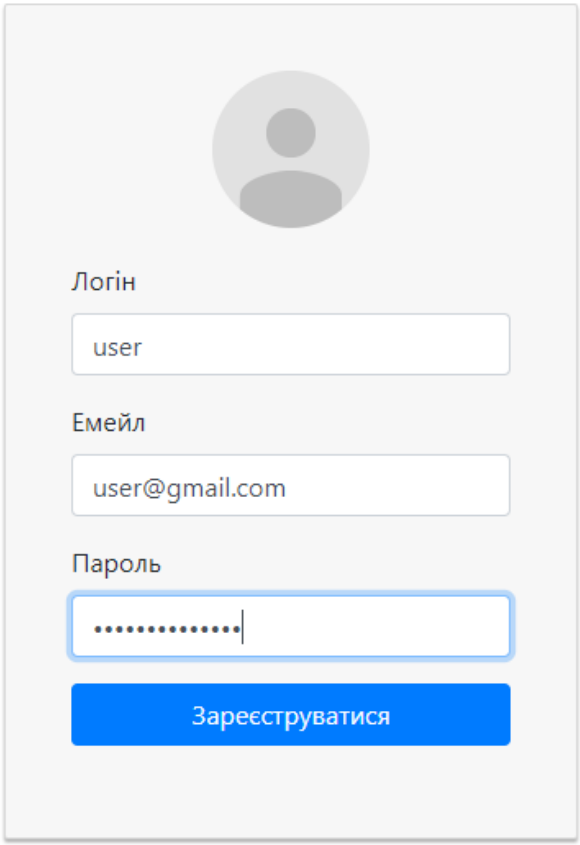


Рисунок 3.2 – Введення даних для реєстрації користувачем

Після натискання кнопки зареєструватися користувачу виводиться повідомлення про успішну реєстрацію (рис. 3.3).

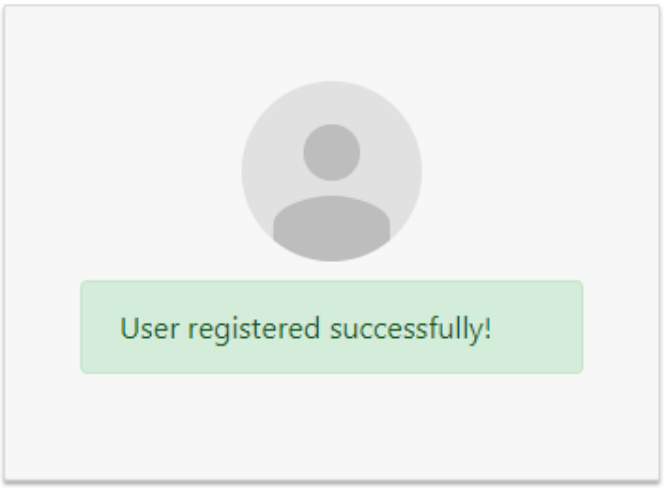


Рисунок 3.3 – Повідомлення про успішну реєстрацію користувача

Після того як користувач успішно зареєструвався, йому потрібно ввійти в свій створений аккаунт на сторінці входу (рис. 3.4).

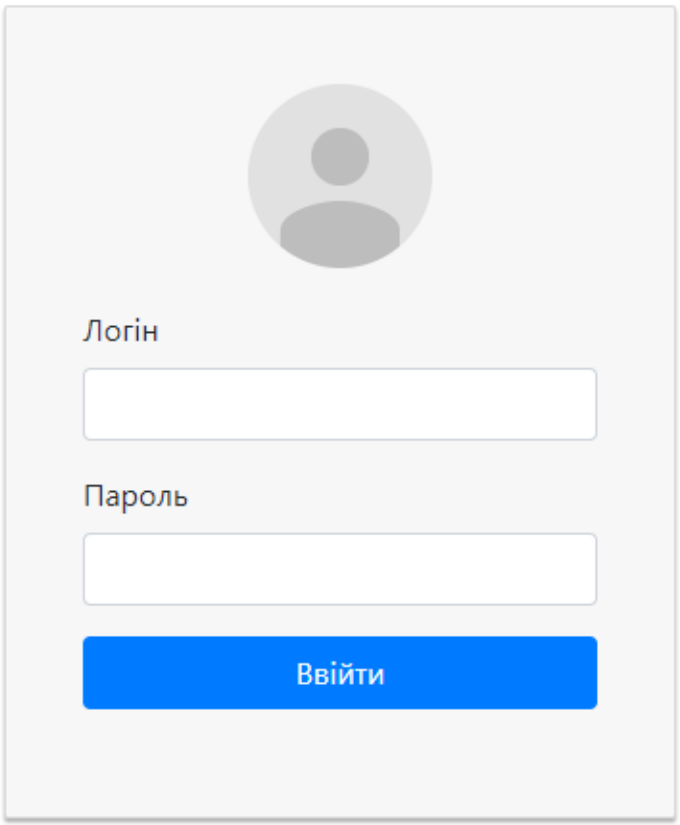


Рисунок 3.4 – Сторінка входу

Після того як користувач проходить процедуру входу, його перенаправляє на захищену сторінку профіля користувача, де він може побачити інформацію про себе (рис. 3.5).

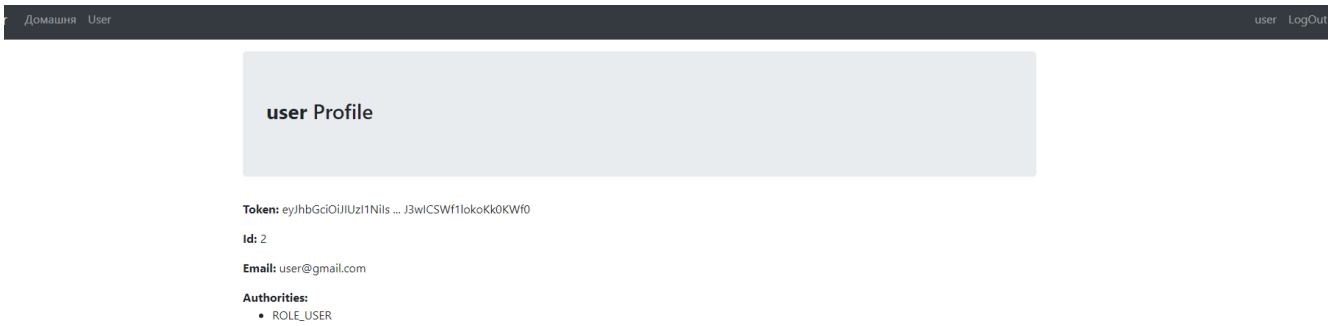


Рисунок 3.5 – Сторінка профіля користувача

Також на сторінці профіля користувача, можна побачити токен доступу користувача (рис. 3.6).

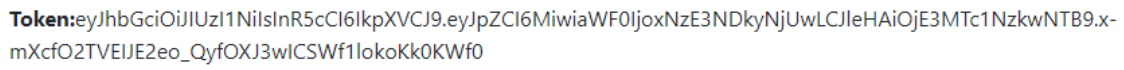


Рисунок 3.6 – Токен доступу користувача

Проаналізуємо токен доступу користувача за допомогою сервісу JWTIO (рис. 3.7).

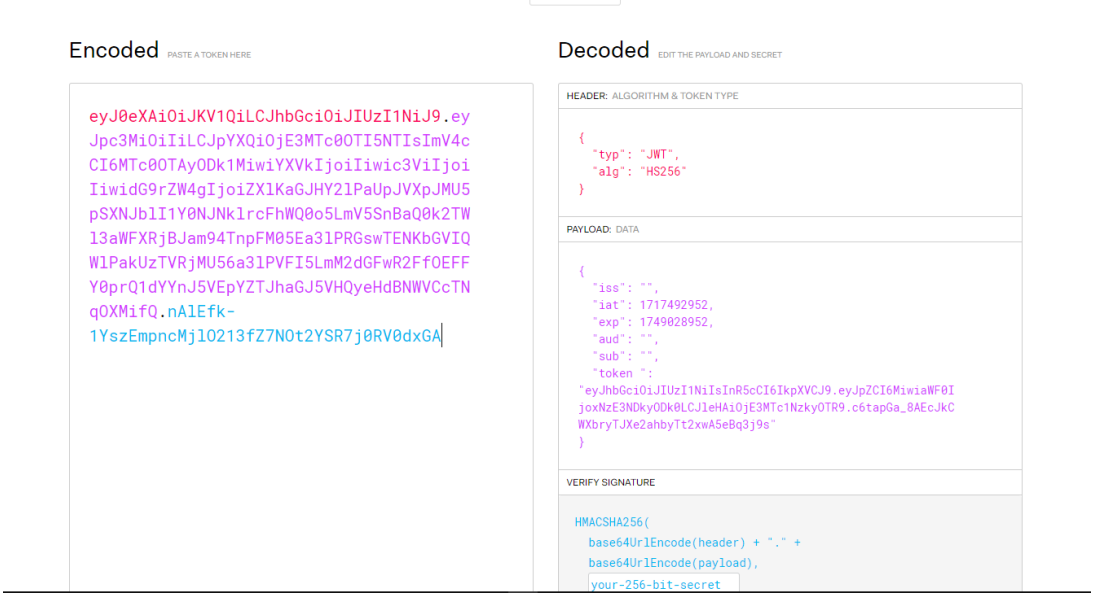


Рисунок 3.7 – Дешифрований токен доступу

Як можна побачити на рисунку 3.7 в середині токена користувача знаходиться, дані про його час життя та ще один токен, що означає, що даний токен є токеном другої генерації. Проаналізуємо токен першої генерації в всередині першого токена (рис. 3.8).

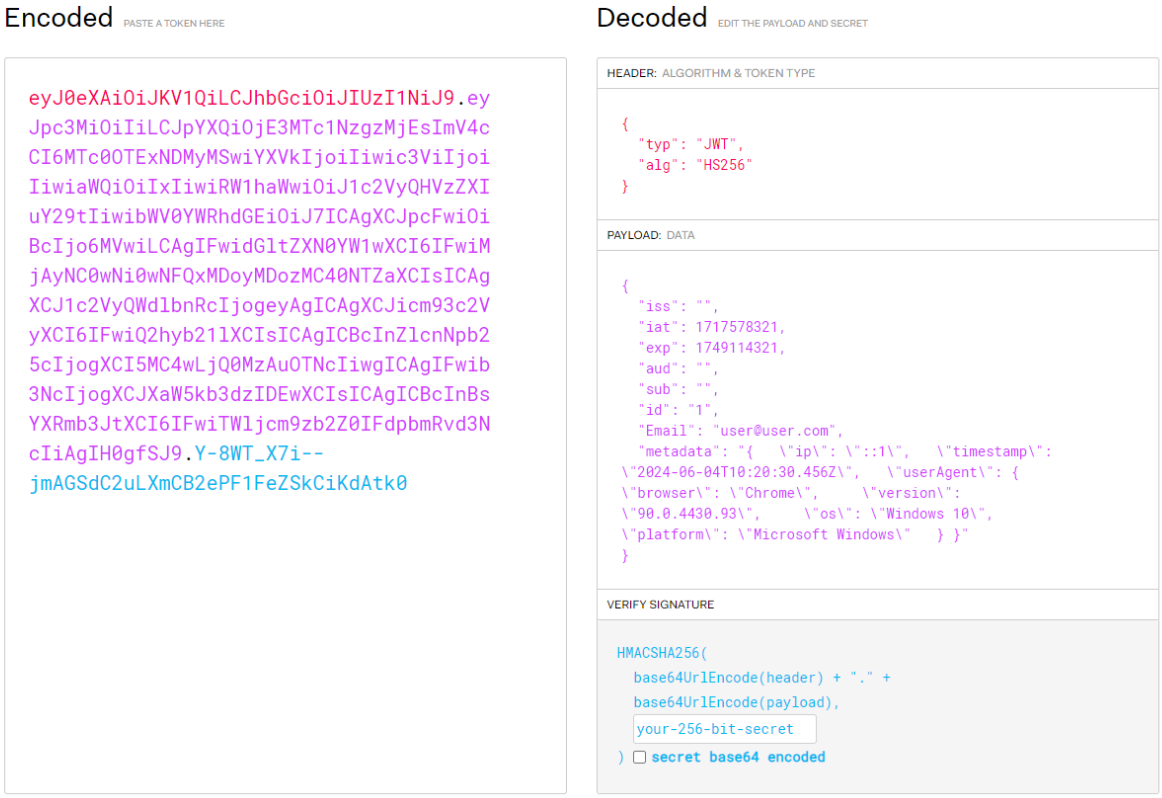


Рисунок 3.8 – Вміст токена першої генерації

Як можна побачити на рисунку 3.8 в середині токена першої генерації знаходяться дані для ідентифікації користувача та його метадані, що свідчить про правильність роботи всієї системи двоетапної генерації токенів на основі метаданих користувача.

3.5 Висновки до розділу

У цьому розділі було детально розглянуто процес створення та реалізації серверної та клієнтської частини додатку для підвищення захищеності авторизації користувачів, що використовує вдосконалену систему з двоетапною генерацією та перевіркою токенів доступу на основі метаданих користувача. Вибір програмної мови та середовища розробки був обґрунтований з урахуванням вимог до безпеки, продуктивності та зручності розробки.

Зокрема, було продемонстровано реалізацію серверної частини на основі фреймворку Express.js, який забезпечує легкість і гнучкість у розробці веб-додатків. Було використано бібліотеку Passport.js для автентифікації користувачів, що дозволяє легко інтегрувати різні стратегії автентифікації, включаючи локальну та токенову (JWT). Крім того, було розглянуто методи захисту, такі як використання SSL/TLS для забезпечення безпеки передачі даних та захист від CSRF-атак.

На клієнтській стороні, інтерфейс було реалізовано за допомогою фреймворку React, що забезпечує створення динамічних та інтуїтивно зрозумілих компонентів. Було розроблено компоненти для входу, реєстрації, роботи із захищеним контентом, а також для відображення сповіщень користувачу. Використання Axios для взаємодії з API серверу дозволило легко інтегрувати клієнтську і серверну частини додатку.

Тестування та аналіз роботи реалізованого додатку включали модульне та інтеграційне тестування, тестування безпеки та аналіз продуктивності. Було продемонстровано, як ці методи допомагають забезпечити правильну роботу окремих компонентів та їх взаємодію, виявити можливі вразливості та оптимізувати продуктивність додатку.

Таким чином, реалізація даного додатку продемонструвала ефективність використання сучасних технологій та методів для створення безпечних і надійних систем авторизації користувачів. Отримані результати можуть бути корисними для подальших досліджень та розробки систем з підвищеним рівнем безпеки в інших областях.

4 ЕКОНОМІЧНА ЧАСТИНА

Науково-технічна розробка повинна відповідати вимогам часу в контексті науково-технічного прогресу та економіки, що обґрунтовує необхідність оцінки економічної ефективності її результатів.

Магістерська кваліфікаційна робота на тему «Підвищення захищеності авторизації користувачів вдосконаленою системою з двоетапною генерацією та перевіркою токенів доступу на основі метаданих користувача» належить до науково-технічних робіт, орієнтованих на комерціалізацію. Це пріоритетний напрямок, оскільки інші споживачі можуть отримати економічний ефект від використання результатів розробки. Для цього необхідно знайти інвестора та переконати його в економічній доцільності проекту.

Для цього випадку потрібно виконати такі етапи:

– провести комерційний аудит науково-технічної розробки, визначивши її науково-технічний рівень та комерційний потенціал;

– розрахувати витрати на здійснення науково-технічної розробки;

– оцінити економічну ефективність розробки при її впровадженні та комерціалізації, обґрунтувавши економічну доцільність для потенційного інвестора.

4.1 Оцінювання комерційного потенціалу розробки програмного забезпечення

Мета проведення технологічного аудиту полягає у визначенні комерційного потенціалу розробки, що виникла в результаті науково-технічної діяльності.

В рамках магістерської кваліфікаційної роботи було розроблено систему для підвищення захищеності авторизації користувачів за допомогою вдосконаленої технології двоетапної генерації та перевірки токенів доступу на основі метаданих користувача. Ця розробка була фактично реалізована у вигляді веб-сервісу, який може знайти широке застосування для забезпечення безпеки в різних веб-додатках.

Технологічний аудит допоможе оцінити потенціал комерціалізації цієї розробки, визначити можливості для її впровадження на ринку, а також залучити потенційних інвесторів для подальшого розвитку та використання.

Оцінювання комерційного потенціалу здійснимо за критеріями, що наведені в таблиці 4.1 [40]

Таблиця 4.1 – Рекомендовані критерії оцінювання науково-технічного рівня і комерційного потенціалу розробки та бальна оцінка

Бали (за 5-ти бальною шкалою)					
	0	1	2	3	4
Технічна здійсненність концепції					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено працездатність продукту в реальних
Ринкові переваги (недоліки)					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів
4	Технічні та споживчі властивості продукту значно гірші,	Технічні та споживчі влас-тивості продукту трохи гірші, ніж в	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в	Технічні та споживчі властивості продукту значно кращі,
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
Ринкові перспективи					
6	Ринок малий і не має позитивної	Ринок малий, але має позитивну	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає

Продовження таблиці 4.1

Практична здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві
11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь-які регламентні обмеження на виробництво та реалізацію продукту

Результати оцінювання науково-технічного рівня та комерційного потенціалу науково-технічної розробки потрібно звести до таблиці.

Таблиця 4.2 – Результати оцінювання науково-технічного рівня і комерційного потенціалу розробки експертами

Критерії	Експерт (ПІБ, посада)		
	1	2	3
	Бали:		
1. Технічна здійсненність концепції	4	5	4
2. Ринкові переваги (наявність аналогів)	5	4	4
3. Ринкові переваги (ціна продукту)	3	4	4
4. Ринкові переваги (технічні властивості)	4	5	4
5. Ринкові переваги (експлуатаційні витрати)	3	4	5
6. Ринкові перспективи (розмір ринку)	5	3	3
7. Ринкові перспективи (конкуренція)	5	3	4
8. Практична здійсненність (наявність фахівців)	5	4	4
9. Практична здійсненність (наявність фінансів)	3	4	3
10. Практична здійсненність (необхідність нових матеріалів)	4	5	5
11. Практична здійсненність (термін реалізації)	4	4	3
12. Практична здійсненність (розробка документів)	4	3	3
Сума балів	*49*	*48*	*46*
Середньоарифметична сума балів $СБ_c$	47,6		

На основі даних, представлених у таблиці 4.2, можна здійснити аналіз комерційного потенціалу розробки. Порівняємо ці результати з вказаними рівнями комерційного потенціалу в таблиці 4.3. [41]

Середньоарифметична сума балів СБ, розрахована на основі висновків експертів	Рівень комерційного потенціалу розробки
0 – 10	Низький
11 – 20	Нижче середнього
21 – 30	Середній
31 – 40	Вище середнього
41 – 48	Високий

В результаті досліджень було встановлено, що рівень комерційного потенціалу розробки, що присвячена підвищенню захищеності авторизації користувачів через удосконалену систему з двоетапною генерацією та перевіркою токенів доступу на основі метаданих користувача, становить 47,6 балів. Згідно з таблицею 4.3, це свідчить про високу комерційну важливість проведення цих досліджень.

4.2 Прогнозування витрат на виконання наукової роботи та впровадження її результатів

Прогнозування витрат на науково-дослідну, дослідно-конструкторську та конструкторсько-технічну роботу складається з трьох ключових етапів, які детально розглядають різні аспекти витрат та впливають на усі аспекти проекту.

На першому етапі визначаються витрати, які безпосередньо пов'язані з ресурсами та зусиллями, витраченими на виконання цієї роботи. Це включає оплату праці, навчання та інші витрати, що виникають у зв'язку з виконанням цієї конкретної роботи.

Другий етап включає розрахунок загальних витрат на виконання всієї роботи, що охоплює витрати на матеріали, обладнання, послуги та інші загальні витрати, пов'язані з проектом в цілому.

Третій етап охоплює прогнозування загальних витрат на впровадження результатів роботи, включаючи витрати на впровадження розробок, рекламу, підготовку персоналу та інші витрати, пов'язані з реалізацією отриманих результатів.

Важливо зазначити, що на цьому етапі використовується конкретна структура витрат, з огляду на те, що для розробки інформаційної технології використовується лише один розробник програмного забезпечення.

Витрати на основну заробітну плату дослідників ($З_о$) розраховуємо у відповідності до посадових окладів працівників, за формулою:

$$З_o = \sum_{i=1}^{k} \frac{M_{ni} \cdot t_i}{T_p}, \quad (4.1)$$

де k – кількість посад дослідників залучених до процесу досліджень;

M_{ni} – місячний посадовий оклад конкретного дослідника, грн;

t_i – число днів роботи конкретного дослідника, дні;

T_p – середнє число робочих днів в місяці, T_p = 22 день. [40]

$$З_о = \frac{36500 * 50}{22} = 82954{,}5 \text{ грн}$$

Таблиця 4.4 – Витрати на заробітну плату дослідників

Найменування посади	Місячний посадовий оклад, грн.	Оплата за робочий день, грн	Число днів роботи	Витрати на заробітну плату
Менеджер	36500	1659,0	50	82954,5
Інженер-розробник	28600	1300	45	58500,0
Всього				141454,5

Погодинну тарифну ставку робітника відповідного розряду C_i можна визначити за формулою:

$$C_i = \frac{M_M \cdot K_i \cdot K_c}{T_p \cdot t_{зм}}, \quad (4.2)$$

де M_M – розмір прожиткового мінімуму працездатної особи, або мінімальної місячної заробітної плати (в залежності від діючого законодавства).

Приймемо *Мм* = 8000 грн;

K_i – коефіцієнт міжкваліфікаційного співвідношення для встановлення тарифної ставки робітнику відповідного розряду.

K_c – мінімальний коефіцієнт співвідношень місячних тарифних ставок робітників першого розряду з нормальними умовами праці виробничих об’єднань і підприємств до законодавчо встановленого розміру мінімальної заробітної плати.

$T_р$ – середнє число робочих днів в місяці, приблизно $T_р$ = 22 день;

$t_{зм}$ – тривалість зміни, год.

$$\mathrm{C_1} = \frac{8000 \times 1.10 \times 1{,}65}{22 \times 8} = 82.5 \text{ грн.}$$

Витрати на основну заробітну плату робітників ($З_р$) за відповідними найменуваннями робіт НДР розраховуємо за формулою:

$$З_р = \sum_{i=1}^{n} C_i \cdot t_i \qquad (4.3)$$

де C_i – погодинна тарифна ставка робітника відповідного розряду, за виконану відповідну роботу, грн/год;

t_i – час роботи робітника при виконанні визначеної роботи, год. [41]

$$\mathrm{Зр_1} = 82{,}5 \times 4 = 330 \text{ грн.}$$

Таблиця 4.5 – Величина витрат на основну заробітну плату робітників

Найменування робіт	Тривалість роботи, год	Розряд роботи	Тарифний коефіцієнт	Погодинна тарифна ставка, грн	Величина оплати на робітника грн
Встановлення обчислювального обладнання	4	2	1,1	82,5	330
Конфігурація системи	3	2	1,1	82,5	247,5
Встановлення програмного забезпечення	3	5	1,7	127,5	385,5
				Всього	963

Додаткову заробітну плату розраховуємо як 10 - 12% від суми основної заробітної плати дослідників та робітників за формулою:

$$З_{дод} = (З_о + З_р) \cdot \frac{Н_{дод}}{100\%} \quad (4.4)$$

де $Н_{дод}$ – норма нарахування додаткової заробітної плати.

$Н_{дод}$ - приймемо, як 12%.

$$З_{дод} = (141454{,}5 + 963) \times \frac{12}{100\%} = 17090{,}1 \text{ грн.}$$

Нарахування на заробітну плату дослідників та працівників становить 22% від суми їх основної та додаткової заробітної плати і розраховується за наступною формулою:

$$З_н = (З_о + З_р + З_{дод}) \cdot \frac{Н_{зп}}{100\%} \quad (4.5)$$

де $Н_{зп}$ – норма нарахування на заробітну плату.

$$З_н = (141454{,}5 + 963 + 17090{,}1) \times \frac{22}{100\%} = 35091{,}7 \text{ грн.}$$

Вартість матеріалів (М) розраховується окремо для кожного виду матеріалів за наступною формулою:

$$М = \sum_{j=1}^{n} Н_j \cdot Ц_j \cdot К_j - \sum_{j=1}^{n} В_j \cdot Ц_{вj}, \quad (4.6)$$

де $Н_j$ – норма витрат матеріалу j-го найменування, кг;

n – кількість видів матеріалів;

$Ц_j$ – вартість матеріалу j-го найменування, грн/кг;

$К_j$ – коефіцієнт транспортних витрат, ($К_j$ *= 1,1 ... 1,15*);

$К_j$ *= 1,1;*

$В_j$ – маса відходів j-го найменування, кг;

$Ц_{вj}$ – вартість відходів j-го найменування, грн/кг. [40]

$$М_1 = 150 * 2 * 1{,}1 - 0{,}0 - 0{,}0 = 330 \text{ грн.}$$

Таблиця 4.6 – Витрати на матеріали

Найменування матеріалу, марка, тип, сорт	Ціна за од, грн	Норма витрат, од	Величина відходів, кг	Ціна відходів, грн/кг	Вартість витраченого матеріалу, грн
Набір офісний	590,0	2	0	0	1298,0
Папір для принтера	174,0	1	0	0	191,4
Картридж для Canon PIXMA	1099,0	1	0	0	1208,9
Блокнот для нотаток	164,0	2	0	0	360,8
Флеш USB Goodram UTS2 Twister 64GB	175,0	1	0	0	192,5
				Всього	3251,6

Витрати на матеріали ($К_в$), використовувані під час проведення науково-дослідної роботи на тему "Підвищення захищеності авторизації користувачів вдосконаленою системою з двоетапною генерацією та перевіркою токенів доступу на основі метаданих користувача", не враховані.

В рамках статті "Спецустаткування для наукових (експериментальних) робіт" враховуються витрати на придбання та виготовлення спеціалізованого обладнання, необхідного для проведення досліджень, а також на його проектування, транспортування, монтаж і встановлення. Проте, у даній науковій роботі витрати на спецустаткування не передбачені.

В рамках статті "Програмне забезпечення для наукових (експериментальних) робіт" враховуються витрати на розробку та придбання спеціалізованих програмних засобів і програмного забезпечення, таких як програми, алгоритми, бази даних, необхідних для проведення досліджень, а також на їх проектування, створення і встановлення.

Балансова вартість програмного забезпечення розраховується за формулою:

$$B_{прг} = \sum_{i=1}^{k} Ц_{iпрг} \cdot C_{прг.i} \cdot K_i \ , \qquad (4.7)$$

де $Ц_{iпрг}$ – ціна придбання одиниці програмного засобу даного виду, грн;

$C_{прг.i}$ – кількість одиниць програмного забезпечення відповідного найменування, які придбані для проведення досліджень, шт.;

K_i – коефіцієнт, що враховує інсталяцію, налагодження програмного засобу тощо, ($K_i = 1{,}10…1{,}12$); [41]

$K_i = 1{,}10$;

k – кількість найменувань програмних засобів.

$$\text{В}_{\text{прг}} = 16340{,}0 \times 2 \times 1{,}1 = 35948{,}0 \text{ грн.}$$

Таблиця 4.7 – Витрати на придбання програмних засобів по кожному виду

Найменування програмного засобу	Кількість, шт	Ціна за одиницю, грн	Вартість, грн
IntelliJ IDEA	2	16340,0	35948,0
Microsoft 365 Персональний	2	1899,0	4177,8
Операційна система Windows 11 Професійна	2	7950,0	17490,0
		Всього	57615,8

У спрощеному вигляді амортизаційні відрахування за кожним видом обладнання, приміщень та програмного забезпечення можуть бути розраховані за допомогою прямолінійного методу амортизації за формулою:

$$A_{обл} = \frac{Ц_б}{T_в} \cdot \frac{t_{вик}}{12} \ , \qquad (4.8)$$

де $Ц_б$ – балансова вартість обладнання, програмних засобів, приміщень тощо, які використовувались для проведення досліджень, грн;

$t_{вик}$ – термін використання обладнання, програмних засобів, приміщень під час досліджень, місяців;

$T_в$ – строк корисного використання обладнання, програмних засобів, приміщень тощо, років. [40]

$$А_{обл} = \frac{35\ 999 \times 2}{3 \times 12} = 1083{,}3$$

Таблиця 4.8 – Амортизаційні відрахування по кожному виду обладнання

Найменування обладнання	Балансова вартість, грн	Строк корисного використання, років	Термін використання обладнання, місяців	Амортизаційні відрахування, грн
Ноутбук Acer Aspire 3 A315-24P	19500,0	3	2	1083,3
Ноутбук Apple MacBook Air 13.6	48500,0	3	2	2694,4
МФУ Canon PIXMA	3600,0	4	2	150,0
Приміщення (офіс)	30 000,0	5	2	250,0
			Всього	4177,7

Витрати на силову електроенергію (Ве) розраховуються за формулою:

$$B_e = \sum_{i=1}^{n} \frac{W_{yi} \cdot t_i \cdot Ц_e \cdot K_{вni}}{\eta_i}, \qquad (4.9)$$

де W_{yi} – встановлена потужність обладнання на визначеному етапі розробки, кВт;

t_i – тривалість роботи обладнання на етапі дослідження, год;

$Ц_e$ – вартість 1 кВт-години електроенергії, грн; (вартість електроенергії визначається за даними енергопостачальної компанії), приймемо $Ц_е$ = 7,50 грн;

$K_{вni}$ – коефіцієнт, що враховує використання потужності, $K_{вni} < 1$;

η_i – коефіцієнт корисної дії обладнання, $\eta_i < 1$.

$$В_е = \frac{0{,}45 \times 400 \times 7{,}50 \times 0{,}95}{0{,}97} = 1322{,}16 \text{ грн.}$$

Таблиця 4.9 – Витрати на електроенергію

Найменування обладнання	Встановлена потужність, кВт	Тривалість роботи, год	Сума, грн
Ноутбук Acer Aspire 3 A315-24P	0,45	400	1322,16
Ноутбук Apple MacBook Air 13.6	0,3	360	793,29
МФУ Canon PIXMA	0,2	25	36,7
Приміщення (офіс)	0,25	400	734,5
		Всього	2886,6

Витрати за цією статтею розраховуються у розмірі 20–25% від суми основної заробітної плати дослідників та робітників за допомогою формули:

$$B_{св} = (З_о + З_р) \cdot \frac{H_{св}}{100\%}, \quad (4.10)$$

де $H_{св}$ – норма нарахування за статтею «Службові відрядження», приймемо $Н_{св}$ = 20%.

$$В_{св} = (141454{,}5 + 963) \times \frac{20}{100\%} = 28\,483{,}5 \text{ грн.}$$

Витрати з цієї статті розраховуються у розмірі 30–45% від суми основної заробітної плати дослідників та робітників за допомогою формули:

$$B_{сп} = (З_о + З_р) \cdot \frac{H_{сп}}{100\%}, \quad (4.11)$$

де $Н_{сп}$ – норма нарахування за статтею «Витрати на роботи, які виконують сторонні підприємства, установи і організації», приймемо $Н_{сп}$= 35%.

$$В_{сп} = (141454{,}5 + 963) \times \frac{35}{100\%} = 49846{,}1 \text{ грн.}$$

Витрати за цією статтею обчислюються у розмірі 50–100% від суми основної заробітної плати дослідників та робітників за допомогою такої формули:

$$I_в = (З_о + З_р) \cdot \frac{Н_{ів}}{100\%}, \quad (4.12)$$

де $Н_{ів}$ – норма нарахування за статтею «Інші витрати», приймемо $Н_{ів}$ = 50%.

$$\text{І}_\text{в} = (141454{,}5 + 963) \times \frac{50}{100\%} = 71208{,}8 \text{ грн.}$$

Витрати за цією статтею розраховуються у розмірі 100–150% від суми основної заробітної плати дослідників та працівників з використанням такої формули:

$$В_{нзв} = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%}, \quad (4.13)$$

де $Н_{нзв}$ – норма нарахування за статтею «Накладні (загальновиробничі) витрати», приймемо $Н_{нзв}$ = 100%.

$$\text{В}_\text{нзв} = (141454{,}5 + 963) \times \frac{100}{100\%} = 142417{,}5 \text{ грн.}$$

Витрати на проведення науково-дослідної роботи розраховуються як сума всіх попередніх статей витрат за формулою:

$$В_{заг} = З_о + З_р + З_{дод} + З_н + М + К_в + В_{спец} + В_{прг} + А_{обл} + В_е + В_{св} + В_{сп} + І_в + В_{нзв} \quad (4.14)$$

$$\begin{aligned}\text{В}_\text{заг} = {} & 141454{,}5 + 963 + 17090{,}1 + 35091{,}7 + 3251{,}6 + 57615{,}8 + 4177{,}7 \\ & + 2886{,}6 + 49846{,}1 + 28\,483{,}5 + 71208{,}8 + 142417{,}5 \\ & = 554\,486{,}9 \text{ грн.}\end{aligned}$$

Вартість завершення науково-дослідної (науково-технічної) роботи та оформлення її результатів обчислюється відповідно до наступної формули:

$$ЗВ = \frac{В_{заг}}{\eta}, \quad (4.15)$$

де η - коефіцієнт, який характеризує етап (стадію) виконання науково-дослідної роботи, приймемо η=0,7.

$$\text{ЗВ} = \frac{554\,486{,}9}{0{,}7} = 792124{,}1 \text{ грн.}$$

Отже, прогноз загальних витрат ЗВ на виконання та впровадження результатів виконаної роботи складає 792124,1 грн.

4.3 Прогнозування комерційних ефектів від реалізації результатів розробки

У ринкових умовах основним позитивним результатом для потенційного інвестора від впровадження науково-технічної розробки є зростання чистого прибутку. Дослідження, спрямовані на підвищення захищеності авторизації користувачів вдосконаленою системою з двоетапною генерацією та перевіркою токенів доступу на основі метаданих користувача, передбачають комерціалізацію протягом трьох років.

У цьому випадку майбутній економічний ефект ґрунтується на збільшенні кількості користувачів продукту протягом аналізованого періоду:

- перший рік – 50 користувачів;
- другий рік – 120 користувачів;
- третій рік – 180 користувачів.

N – кількість споживачів які використовували аналогічний продукт у році до впровадження результатів нової науково-технічної розробки. Приймемо 70 користувачів;

$Ц_б$ – вартість програмного продукту у році до впровадження результатів розробки, приймемо 111000,00 грн;

$\pm\Delta Ц_о$ – зміна вартості програмного продукту від впровадження результатів науково-технічної розробки, приймемо 11 000,00 грн.

Для кожного з випадків потенційне збільшення чистого прибутку у потенційного інвестора ΔПі в роки очікуваного позитивного результату від можливого впровадження та комерціалізації науково-технічної розробки розраховується за відповідною формулою:

$$\Delta П_i = (\pm\Delta Ц_о \cdot N + Ц_о \cdot \Delta N)_i \cdot \lambda \cdot \rho \cdot (1 - \frac{\vartheta}{100}), \quad (4.16)$$

де λ – коефіцієнт, який враховує сплату потенційним інвестором податку на додану вартість. У 2024 році ставка податку на додану вартість складає 20%, а коефіцієнт $\lambda = 0{,}8333$;

ρ – коефіцієнт, який враховує рентабельність інноваційного продукту. Приймемо ρ=30%;

ϑ – ставка податку на прибуток, який має сплачувати потенційний інвестор, у 2023 році ϑ=18%; [40]

$$\Delta П_1 = (11\,000{,}00 \times 70 + 111000{,}00 \times 50) \times 0{,}83 \times 0{,}3 \times \left(1 - \frac{0{,}18}{100}\right) = 1\,570\,847{,}38 \text{ грн.}$$

$$\Delta П_2 = \left(11\,000{,}00 \times 70 + 111000{,}00 \times (50 + 120)\right) \times 0{,}83 \times 0{,}3 \times \left(1 - \frac{0{,}18}{100}\right) = 4\,881\,557{,}35 \text{ грн.}$$

$$\Delta П_3 = \left(11\,000{,}00 \times 70 + 111000{,}00 \times (50 + 120 + 180)\right) \times 0{,}83 \times 0{,}3 \times \left(1 - \frac{0{,}18}{100}\right) = 9\,847\,622{,}32 \text{ грн.}$$

Для кожного з випадків потенційне збільшення чистого прибутку у потенційного інвестора $\Delta П_i$ в роки очікуваного позитивного результату від можливого впровадження та комерціалізації науково-технічної розробки розраховується за відповідною формулою:

$$ПП = \sum_{i=1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, \quad (4.17)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному з років, протягом яких виявляються результати впровадження науково-технічної розробки, грн;

T – період часу, протягом якого очікується отримання позитивних результатів від впровадження та комерціалізації науково-технічної розробки, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні, τ=0,2;

t – період часу (в роках) від моменту початку впровадження науково-технічної розробки до моменту отримання потенційним інвестором додаткових чистих прибутків у цьому році. [41]

$$ПП = \frac{1570847,38}{(1+0,2)^1} + \frac{4881557,35}{(1+0,2)^2} + \frac{9847622,32}{(1+0,2)^3} = 10397865,37 \text{ грн.}$$

Отже, згідно з розрахунками, комерційна користь від упровадження розробки буде значною, що підтверджує прогнози, і проявиться у зростанні чистого прибутку підприємства.

4.4 Розрахунок ефективності вкладених інвестицій та періоду їх окупності

Ключовими факторами, що визначають обґрунтованість інвестування певним інвестором у наукову розробку, є абсолютна та відносна ефективність інвестицій, а також термін їх повернення. Першим кроком на цьому шляху є розрахунок сучасної вартості інвестицій (PV), вкладених у наукову розробку.

Для цього можна використати формулу:

$$PV = k_{інв} \cdot ЗВ, \tag{4.18}$$

де $k_{інв}$ – коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію, приймаємо $k_{інв}$=3;

$ЗВ$ – загальні витрати на проведення науково-технічної розробки та оформлення її результатів, приймаємо 792124,1 грн.

$$\text{PV} = 3 * 792124,1 = 2\,376\,372,3 \text{ грн.}$$

Таким чином, чистий приведений дохід (NPV) або абсолютний економічний ефект ($Е_{абс}$) для потенційного інвестора від можливого впровадження та комерціалізації науково-технічної розробки буде таким:

$$E_{абс} = ПП - PV \tag{4.19}$$

де $ПП$ – приведена вартість зростання всіх чистих прибутків від можливого впровадження та комерціалізації науково-технічної розробки, 10397865,37 грн.

PV – теперішня вартість початкових інвестицій, 2 376 372,3 грн.

$$\text{Е}_{\text{абс}} = 10397865,37 - 2\,376\,372,3 = 8\,021\,493,07 \text{ грн.}$$

Внутрішня економічна дохідність ($Е_в$) інвестицій, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки, обчислюється за допомогою такої формули:

$$E_в = \sqrt[T_ж]{1 + \frac{E_{абс}}{PV}} - 1, \quad (4.20)$$

де $E_{абс}$ – абсолютний економічний ефект вкладених інвестицій, 8 021 493,07 грн.

PV – теперішня вартість початкових інвестицій, 2 376 372,3 грн;

$T_ж$ – життєвий цикл науково-технічної розробки, тобто час від початку її розробки до закінчення отримування позитивних результатів від її впровадження, 3 роки. [40]

$$\text{Е}_\text{в} = \sqrt[3]{1 + \frac{8\,021\,493{,}07}{2\,376\,372{,}3}} - 1 = 0{,}6$$

Мінімальна внутрішня економічна дохідність вкладених інвестицій (мін τ) визначається згідно такою формулою:

$$\tau_{мін} = d + f, \quad (4.21)$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2024 році в Україні d = 0,2;

f – показник, що характеризує ризикованість вкладення інвестицій, приймемо 0,2.

$$\tau_{min} = 0{,}2 + 0{,}2 = 0{,}4$$

Оскільки $E_в$ = 60% > τ_{min} = 40%, це свідчить про те, що внутрішня економічна дохідність інвестицій, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки, перевищує мінімальну внутрішню дохідність.

Далі обчислюємо період окупності інвестицій ($Т_{ок}$ або DPP, Discounted Payback Period), які потенційний інвестор може вкласти у впровадження та комерціалізацію науково-технічної розробки:

$$T_{ок} = \frac{1}{E_в}, \quad (4.22)$$

$$\mathrm{T_{ок}} = \frac{1}{0{,}8} = 1{,}6 \text{ року.}$$

З огляду на те, що період окупності інвестицій у реалізацію наукового проєкту становить менше трьох років, можна дійти висновку, що фінансування цієї нової розробки є виправданим.

4.5 Висновки до розділу

Згідно з проведеними дослідженнями, рівень комерційного потенціалу розробки за темою «Підвищення захищеності авторизації користувачів вдосконаленою системою з двоетапною генерацією та перевіркою токенів доступу на основі метаданих користувача» становить 47,6 балів, що свідчить про високу комерційну значущість цих досліджень.

Термін окупності проекту складає 1,6 року, що є значно меншим за стандартний трирічний період. Це підтверджує комерційну привабливість розробки і може мотивувати потенційних інвесторів до фінансування її впровадження та виходу на ринок.

Таким чином, можна зробити висновок про доцільність проведення науково-дослідних робіт за цією темою, оскільки вони мають високий потенціал для комерційної успішності.

ВИСНОВКИ

У роботі проведено дослідження та розробку вдосконаленої системи підвищення захищеності авторизації користувачів, яка базується на двоетапній генерації та перевірці токенів доступу з використанням метаданих користувача. Здійснений аналіз сучасних методів авторизації показав, що традиційні підходи, такі як паролі та одноразові коди, вже не забезпечують необхідного рівня захисту в умовах сучасних кіберзагроз, що зумовлює необхідність розробки нових методів, здатних ефективно протистояти різноманітним видам атак.

Розробка алгоритмів двоетапної генерації та перевірки токенів доступу продемонструвала ефективність використання метаданих користувача для забезпечення високого рівня захисту. Алгоритми, розроблені у межах цієї роботи, забезпечують надійну перевірку автентичності користувачів та знижують ризик несанкціонованого доступу. Проектування архітектури системи дозволило створити комплексну та ефективну систему підвищення захищеності авторизації користувачів. Запропонована архітектура включає механізми збору та аналізу метаданих, а також алгоритми генерації та перевірки токенів доступу.

Програмна реалізація розробленої системи продемонструвала можливість її успішного впровадження в різноманітні інформаційні системи. Результати тестування показали високу ефективність та надійність запропонованого рішення. Економічний аналіз підтвердив комерційну доцільність впровадження розробленої системи. Прогнозовані витрати на виконання наукової роботи та впровадження результатів є обґрунтованими, а очікувані комерційні ефекти від реалізації розробки перевищують витрати, забезпечуючи високу ефективність вкладених інвестицій та швидку окупність.

Загалом, результати роботи підтверджують доцільність та ефективність впровадження вдосконаленої системи авторизації з двоетапною генерацією та перевіркою токенів доступу на основі метаданих користувача. Запропоноване рішення забезпечує високий рівень захисту інформаційних систем, знижуючи ризик несанкціонованого доступу та підвищуючи безпеку конфіденційної інформації.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Authentication - Django REST framework. *Home - Django REST framework*. URL: https://www.django-rest-framework.org/api-guide/authentication/.

2. Jena B. K. AES Encryption: Secure Data with Advanced Encryption Standard. *Simplilearn.com*. URL: https://www.simplilearn.com/tutorials/cryptography-tutorial/aes-encryption.

3. Magnusson A. Token-based Authentication: Everything You Need to Know. *StrongDM: Your Partner in Zero Trust Privileged Access*. URL: https://www.strongdm.com/blog/token-based-authentication.

4. What Is an Authentication Token? | Fortinet. *Fortinet*. URL: https://www.fortinet.com/resources/cyberglossary/authentication-token.

5. What Is Token-Based Authentication? | Okta. *Employee and Customer Identity Solutions | Okta*. URL: https://www.okta.com/identity-101/what-is-token-based-authentication/.

6. What is token-based authentication?. *Stack Overflow*. URL: https://stackoverflow.com/questions/1592534/what-is-token-based-authentication.

7. Cloudflare Application Services | Security and Performance. Cloudflare. URL: https://www.cloudflare.com/application-services/?utm_source=google&utm_medium=cpc&utm_campaign=DG_EMEig7BqZ7VtZrN6wUzk1Nhp6nkaAgwEEALw_wcB&gclsrc=aw.ds).

8. Кібербезпека. URL: https://www.span.eu/ua/рішення-та-послуги/сервіси-з-безпеки/кібербезпека/?gad_source=1&gclid=Cj0KCQiA67CrBhC1ARIsACKAa8REXQS8JOs2ZaaOW6g5OyaeVWM8ksz3M6viwjX_pclBefnGsQgibfMaAgYHEALw_wcB.

9. Contributors to Wikimedia projects. Information security - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/Information_security

10. Imperva. URL: https://www.imperva.com/learn/data-security/information-security-infosec/.

11. What is information security (infosec)?. Cisco. URL: https://www.cisco.com/c/en/us/products/security/what-is-information-security-infosec.html.

12. What is information security (infosec)? Goals, types and applications. Exabeam. URL: https://www.exabeam.com/explainers/information-security/information-security-goals-types-and-applications/.

13. Yasar K., Wright G., Teravainen T. What is information security (infosec)? – techtarget definition. Security. URL: https://www.techtarget.com/searchsecurity/definition/information-security-infosec.

14. What is information security? | IBM. IBM in Deutschland, Österreich und der Schweiz | IBM. URL: https://www.ibm.com/topics/information-security.

15. What is information security? - geeksforgeeks. GeeksforGeeks. URL: https://www.geeksforgeeks.org/what-is-information-security/.

16. INFOSEC - Glossary | CSRC. NIST Computer Security Resource Center | CSRC. URL: https://csrc.nist.gov/glossary/term/INFOSEC.

17. Методичні вказівки до виконання економічної частини магістерських кваліфікаційних робіт / Уклад. : В. О. Козловський, О. Й. Лесько, В. В. Кавецький. Вінниця : ВНТУ, 2021. 42 с.

18. Кавецький В. В. Економічне обґрунтування інноваційних рішень: практикум / В. В. Кавецький, В. О. Козловський, І. В. Причепа. Вінниця : ВНТУ, 2016. 113 с.

19. What is authorization? - examples and definition - auth0. *Auth0*. URL: https://auth0.com/intro-to-iam/what-is-authorization .

20. Academy B. Seed Phrase | Binance Academy. Binance Academy. URL: https://academy.binance.com/en/glossary/seed-phrase.

21. What is authorization? - examples and definition - auth0. *Auth0*. URL: https://auth0.com/intro-to-iam/what-is-authorization .

22. Aholi ICT limited - cyber security services company in bangladesh. Aholi ICT Limited - Cyber Security Services Company in Bangladesh. URL: https://www.aholiict.com/usbsecurity.php

23. What Is a USB Security Key?. USB Memory Direct. URL: https://www.usbmemorydirect.com/blog/what-is-a-usb-security-key/

24. Eban Escott. Adopting a AAA approach to software security. URL: https://workingmouse.com.au/software-development/adopting-a-aaa-approach-to-software-security/

25. Suzanne Humphries. What is a USB security key, and should you use one?. URL: https://www.reviewgeek.com/63448/what-is-a-usb-security-key-and-should-you-use-one/

26. What is authentication, authorization, and accounting (AAA)?. URL: https://www.fortinet.com/resources/cyberglossary/aaa-security.

27. Visual studio code. Visual Studio Code. URL: https://code.visualstudio.com/

28. Редактор коду visual studio code. Habr. URL: https://habr.com/ua/articles/490754 (дата звернення: 25.05.2023).

29. JavaScript. URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript

30. What is javascript? A basic introduction to JS for beginners. Hostinger Tutorials. URL: https://www.hostinger.com/tutorials/what-is-javascript).

31. Visual studio code. Visual Studio Code. URL: https://code.visualstudio.com/).

32. Редактор коду visual studio code. Habr. URL: https://habr.com/ua/articles/490754

33. JavaScript. URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript.

34. Electron. Electron. URL: https://www.electronjs.org/ .

35. Node.js. Node.js. URL: https://nodejs.org/uk .

36. The Modern JavaScript. URL: https://javascript.info/ .

37. Sufiyan T. What is node.js: a comprehensive guide. *Simplilearn.com*. URL: https://www.simplilearn.com/tutorials/nodejs-tutorial/what-is-nodejs).

38. Megida D. What is javascript? A definition of the JS programming language. freeCodeCamp.org. URL: https://www.freecodecamp.org/news/what-is-javascript-definition-of-js/

39. What is javascript? A basic introduction to JS for beginners. Hostinger Tutorials. URL: https://www.hostinger.com/tutorials/what-is-javascript).

40. Методичні вказівки до виконання економічної частини магістерських кваліфікаційних робіт / Уклад. : В. О. Козловський, О. Й. Лесько, В. В. Кавецький. Вінниця : ВНТУ, 2021. 42 с.

41. Кавецький В. В. Економічне обґрунтування інноваційних рішень: практикум / В. В. Кавецький, В. О. Козловський, І. В. Причепа. Вінниця : ВНТУ, 2016. 113 с

ДОДАТКИ

Додаток А. Технічне завдання

Вінницький національний технічний університет
Факультет менеджменту та інформаційної безпеки
Кафедра менеджменту та безпеки інформаційних систем

ЗАТВЕРДЖУЮ
Голова секції "Управління інформаційною безпекою" кафедри МБІС
д.т.н., професор
_____ Юрій ЯРЕМЧУК
"____" _____ 2024 р.

ТЕХНІЧНЕ ЗАВДАННЯ

до магістерської кваліфікаційної роботи на тему:
Підвищення захищеності авторизації користувачів вдосконаленою системою з двоетапною генерацією та перевіркою токенів доступу на основі метаданих користувача
08-72.МКР.000.00.000.ТЗ

Керівник магістерської кваліфікаційної роботи
д.т.н., професор каф. МБІС
_____ Яремчук Ю.Є.

Вінниця – 2024 р.

1. Найменування та область застосування

Підвищення захищеності авторизації користувачів вдосконаленою системою з двоетапною генерацією та перевіркою токенів доступу на основі метаданих користувача **2.Підстава для розробки**

Розробка виконується на основі наказу ректора ВНТУ № 81 від 11. 03. 2024 р.

3. Мета та призначення розробки

3.1 Мета розробки: підвищення захищеності авторизації користувачів вдосконаленою системою з двоетапною генерацією та перевіркою токенів доступу на основі метаданих користувача

3.2 Призначення: підвищення захищеності авторизації користувачів

4. Джерела розробки

4.1. Ахрамович В. М. Ідентифікація й аутентифікація, керування доступом // Сучасний захист інформації. – 2016. №4.– С. 47-51.

4.2. Бурячок В.Л. Політика інформаційної безпеки: підручник. / В.Л.Бурячок, Р.В.Грищук, В.О.Хорошко / За заг. ред. докт. техн. наук, проф. В.О. Хорошка. – К.: ПВП «Задруга», 2014. – 222 с.

4.3. Єсін В.І. Безпека інформаційних систем і технологій / В.І.Єсін, О.О. Кузнецов, Л.С. Сорока. – Харків: ХНУ імені В.Н. Каразіна, 2013. – 632 с.

4.4. ZakariaOmar, ZangooeiToomaj, MohdAfiziMohdShukran. Enhancing Mixing Recognition-Based and Recall-Based Approach in Graphical Password Scheme. IJACT, Vol. 4, No. 15, pp. 189-197, 2012.

5. Вимоги до програми

5.1 Вимоги до функціональних характеристик:

5.1.1 Програмний засіб повинен мати зручний, легкий у використанні інтерфейс користувача;

5.1.2 Реалізація не повинна вимагати спеціальних ліцензійних програмних додатків;

5.1.3 Програмний засіб повинен виконувати процес автентифікації користувачів у системі.

5.2 Вимоги до надійності:

5.2.1 Програмний засіб повинен працювати без помилок, у випадку виникнення критичних ситуацій необхідно передбачити виведення відповідних повідомлень;

5.2.2 Бази даних повинні бути налаштовані на автоматичне створення резервних копій;

5.2.3 Програмний засіб повинен виконувати свої функції.

5.3 Вимоги до складу і параметрів технічних засобів:

– процесор – Pentium 1500 МГц і подібні до них;

– оперативна пам’ять – не менше 512 Mb;

– середовище функціонування – операційна система сімейство Windows;

– вимоги до техніки безпеки при роботі з програмою повинні відповідати існуючим вимогам та стандартам з техніки безпеки при користуванні комп’ютерною технікою.

6. Вимоги до програмної документації

6. Вимоги до програмної документації

6.1 Обов'язкова поетапна інструкція для майбутніх користувачів, наведена у пункті 3.4

7. Вимоги до технічного захисту інформації

7.1 Необхідно забезпечити захист розроблюваного програмного засобу від несанкціонованого використання.

7.2 Неможливість отримання доступу незареєстрованих користувачів до інформаційних ресурсів.

8. Техніко-економічні показники

8.1 Цінність результатів використання даного проекту повинна перевищувати витрати на його реалізацію.

8.2 Має бути реалізований таким чином, щоб підходити для використання широкого загалу.

9. Стадії та етапи розробки

№	Назва етапів магістерської кваліфікаційної роботи	Строк виконання етапів роботи		Примітка
1.	Визначення напрямку магістерської роботи, формулювання теми	1.03.2024	15.03.2024	
2.	Аналіз предметної області обраної теми	15.03.2024	1.04.2024	
3.	Розробка роботи	1.04.2024	10.04.2024	
4.	Написання магістерської роботи на основі розробленої теми	10.04.2024	20.04.2024	
5.	Передзахист магістерської кваліфікаційної роботи	20.04.2024	10.05.2024	
6.	Виправлення, уточнення, корегування магістерської кваліфікаційної роботи	10.05.2024	4.06.2024	
7.	Захист магістерської кваліфікаційної роботи	10.06.2024	10.06.2024	

10. Порядок контролю та прийому

10.1 До приймання магістерської кваліфікаційної роботи надається:

– ПЗ до магістерської кваліфікаційної роботи;
– програмний додаток;
– презентація;
– відзив керівника роботи;
– відзив опонента

Технічне завдання до виконання прийняв _____ Грабар Р.І.

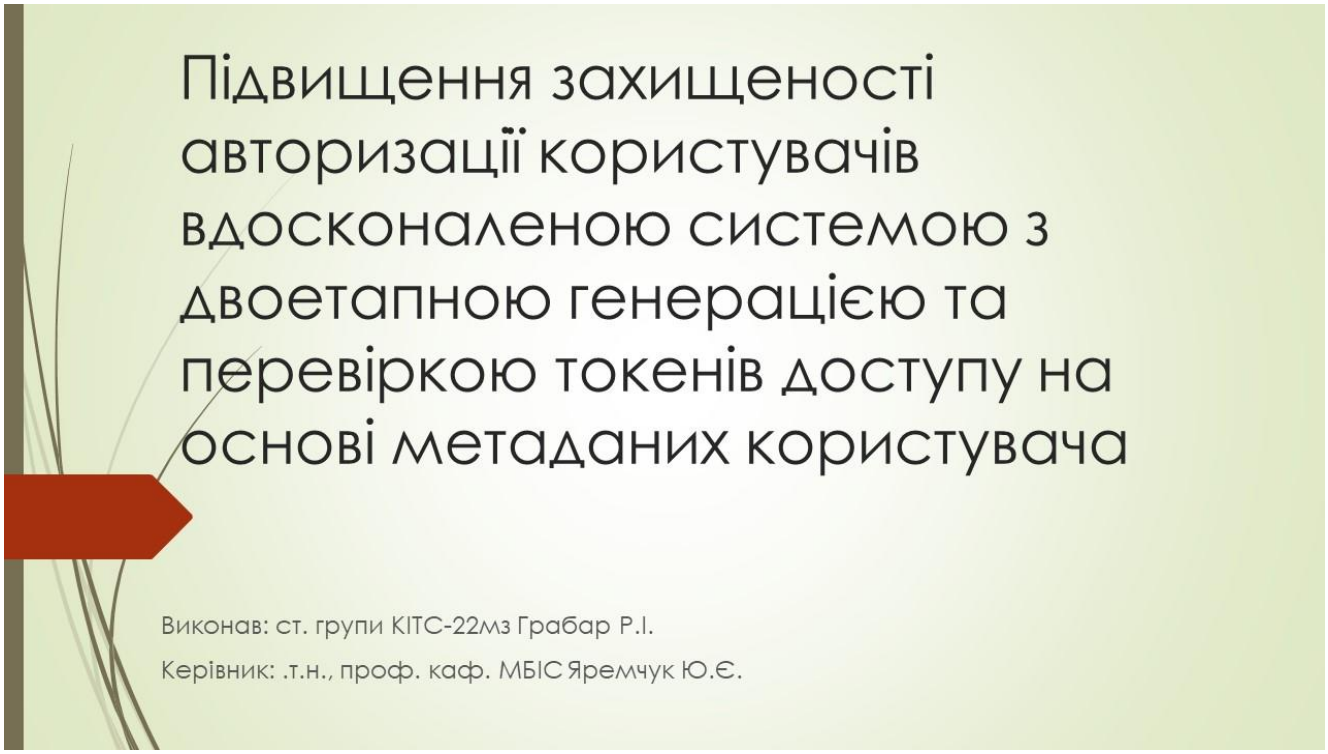
Додаток Б. Лістинг програми

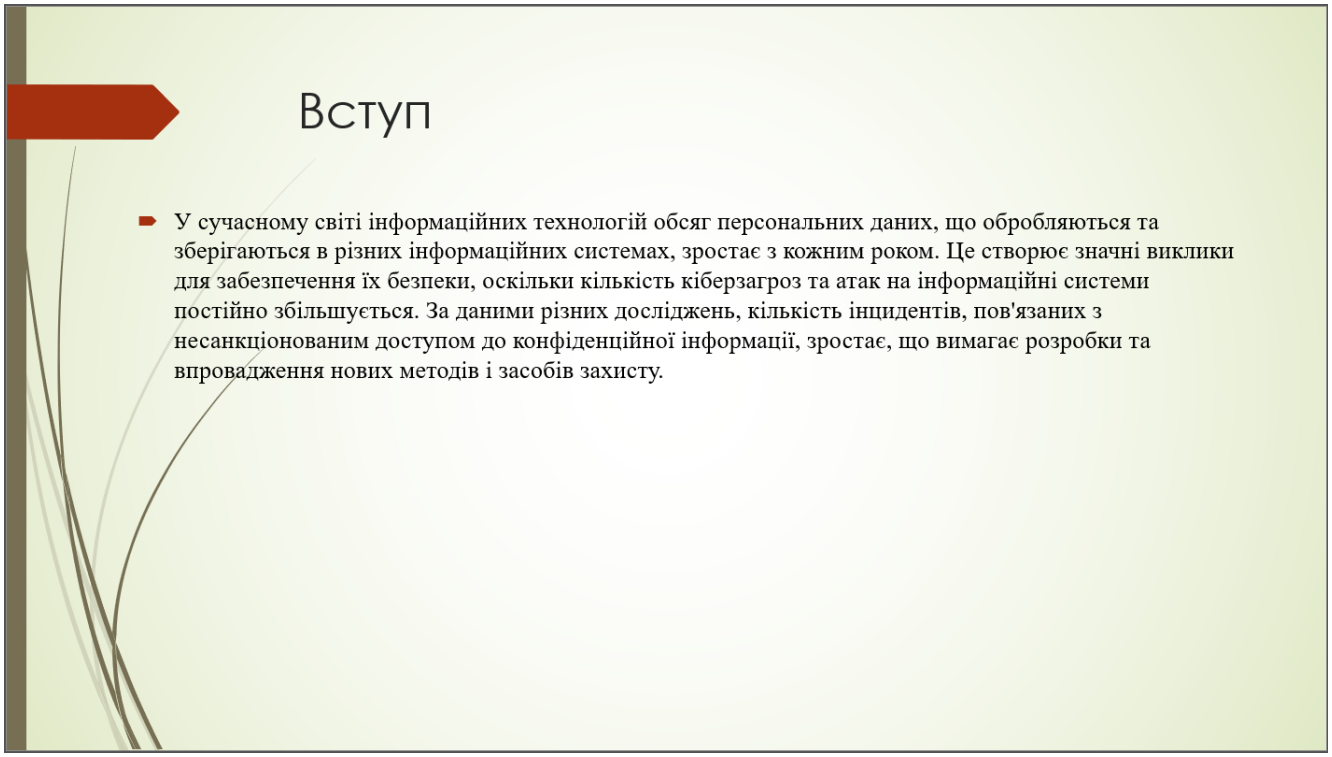
```
import React from 'react';
import { BrowserRouter as Router, Route, Switch } from 'react-router-dom';
import LoginForm from './components/LoginForm';
import RegisterForm from './components/RegisterForm';
import ProtectedComponent from './components/ProtectedComponent';
import Notification from './components/Notification';

function App() {
 return (
  <Router>
   <div className="App">
    <Notification />
    <Switch>
     <Route path="/login" component={LoginForm} />
     <Route path="/register" component={RegisterForm} />
     <Route path="/protected" component={ProtectedComponent} />
    </Switch>
   </div>
  </Router>
 );
}

export default App
import React, { useState } from 'react';
import { useHistory } from 'react-router-dom';
import axios from 'axios';
function LoginForm() {
 const [username, setUsername] = useState('');
 const [password, setPassword] = useState('');
 const history = useHistory();

 const handleLogin = async (event) => {
  event.preventDefault();
  try {
   const response = await axios.post('/auth/login', { username, password });
   if (response.data.success) {
    history.push('/protected');
   } else {
    alert('Login failed');
   }
  } catch (error) {
   alert('An error occurred');
  }
 };
 return (
  <form onSubmit={handleLogin}>
   <div>
    <label>Username:</label>
    <input type="text" value={username} onChange={(e) => setUsername(e.target.value)} />
   </div>
   <div>
    <label>Password:</label>
    <input type="password" value={password} onChange={(e) => setPassword(e.target.value)} /
   </div>
   <button type="submit">Login</button>
  </form>
```

```
  );
}
import React, { useState } from 'react';
import axios from 'axios';
function RegisterForm() {
  const [username, setUsername] = useState('');
  const [password, setPassword] = useState('');
  const handleRegister = async (event) => {
    event.preventDefault();
    try {
      const response = await axios.post('/auth/register', { username, password });
      if (response.data.success) {
        alert('Registration successful');
      } else {
        alert('Registration failed');
      }
    } catch (error) {
      alert('An error occurred');
    }
  };

  return (
    <form onSubmit={handleRegister}>
      <div>
        <label>Username:</label>
        <input type="text" value={username} onChange={(e) => setUsername(e.target.value)} />
      </div>
      <div>
        <label>Password:</label>
        <input type="password" value={password} onChange={(e) => setPassword(e.target.value)}
/>
      </div>
      <button type="submit">Register</button>
    </form>
  );
}
export default RegisterForm;
import React, { useEffect, useState } from 'react';
import axios from 'axios';
function ProtectedComponent() {
  const [message, setMessage] = useState('');
  useEffect(() => {
    const fetchData = async () => {
      try {
        const response = await axios.get('/auth/protected', {
          headers: {
            'Authorization': `Bearer ${localStorage.getItem('token')}`
          }
        });
        setMessage(response.data.message);
      } catch (error) {
        setMessage('You are not authorized to view this content');
      }
    };

    fetchData();
  }, []);
  return (
```

```
    <div>
      <h1>Protected Content</h1>
      <p>{message}</p>
    </div>
  );
}
export default ProtectedComponent;
import axios from 'axios';
export const register = async (username, password) => {
  return await axios.post('/auth/register', { username, password });
};

export const login = async (username, password) => {
  return await axios.post('/auth/login', { username, password });
};

export const getProtectedData = async (token) => {
  return await axios.get('/auth/protected', {
    headers: {
      'Authorization': `Bearer ${token}`
    }
  });
};
```

Додаток В. Ілюстративний матеріал





Актуальність та Мета

- Метою даної роботи є підвищення захищеності авторизації користувачів шляхом розробки та впровадження вдосконаленої системи з двоетапною генерацією та перевіркою токенів доступу на основі метаданих користувача.
- Актуальність. Традиційні методи авторизації, такі як паролі або одноразові коди, все частіше виявляються недостатніми для забезпечення необхідного рівня безпеки. Атаки типу фішинг, підміна сесій, атаки на посередника та інші стають більш витонченими, що підвищує ризик компрометації облікових даних користувачів. У цьому контексті виникає потреба у впровадженні більш ефективних механізмів авторизації, які забезпечують підвищений рівень захисту.Одним із перспективних напрямків у цій сфері є використання двоетапної генерації та перевірки токенів доступу на основі метаданих користувача. Метадані, такі як геолокація, час доступу, інформація про пристрій тощо, дозволяють значно підвищити точність ідентифікації користувачів та забезпечити додатковий рівень захисту. Впровадження такої системи дозволяє зменшити ризики несанкціонованого доступу та забезпечити більш надійний захист конфіденційної інформації.

Сучасні методи авторизації

- У сучасному цифровому світі, де доступ до важливих даних та ресурсів є ключем до успіху, забезпечення безпеки ідентифікації користувачів стає пріоритетним завданням. Методи авторизації відіграють визначну роль у забезпеченні захищеного доступу до систем, програм та інформаційних ресурсів. Однак із зростанням кількості і складності цифрових загроз, необхідно постійно оновлювати та удосконалювати методи забезпечення ідентифікації.
- Сучасні методи авторизації пропонують широкий спектр можливостей, від традиційних паролів до нових безпарольних методів. Кожен метод має свої переваги та недоліки, тому важливо вибрати метод, який відповідає потребам у безпеці та зручності.

Порівняння вразливостей у системах авторизації

Вразливість	Тип атаки	Доступність	Стійкість до перехоплення сесій	Заходи безпеки
Слабкі паролі	Атаки грубої сили, словникової атаки	Висока	Низька	Політика паролів, шифрування, 2FA
Перехоплення сесій	"Людина посередині", CSRF	Висока	Низька	HTTPS, CSRF-токені, HSTS
XSS	Введення JavaScript	Висока	Не застосовується	Валідація, екранування, CSP
Injection Attacks	SQL, NoSQL, LDAP	Висока	Не застосовується	Параметризована передача, валідація, брандмауери
Недостатні права	Вертикальне/горизонтальне розширення	Висока	Не застосовується	Принцип мінімальних привілеїв, RBAC
Недостатні права	Вертикальне/горизонтальне розширення	Висока	Не застосовується	Принцип мінімальних привілеїв, RBAC
Недостатня перевірка даних	Введення несанкціонованих даних	Висока	Не застосовується	Валідація, екранування, санітизація
Недостатня обробка помилок	Розкриття інформації	Висока	Не застосовується	Детальне логування, інформативні повідомлення

Роль метаданих користувача в авторизації

- Роль метаданих користувача в авторизації полягає в тому, що ці дані допомагають ідентифікувати користувача та встановлювати його права доступу до різних ресурсів в інформаційній системі. Метадані - це дані, що описують характеристики або властивості інших даних. У випадку авторизації, метадані користувача можуть включати такі дані, як ім'я, електронна пошта, роль в системі, права доступу, інформація про попередні дії користувача, тощо.
- Метадані ідентифікації користувача використовуються для перевірки, чи користувач дійсно той, за кого себе видає, та надання доступу до відповідних ресурсів або функціональностей. Збір і використання цих метаданих в авторизаційному процесі допомагає забезпечити безпеку та впевненість у тому, що лише правомірні користувачі мають доступ до системи.

Переваги двоетапної генерації та перевірки токенів

- Переваги двоетапної генерації та перевірки токенів у системі авторизації полягають у збільшенні рівня безпеки та зниженні ризику несанкціонованого доступу, а саме:

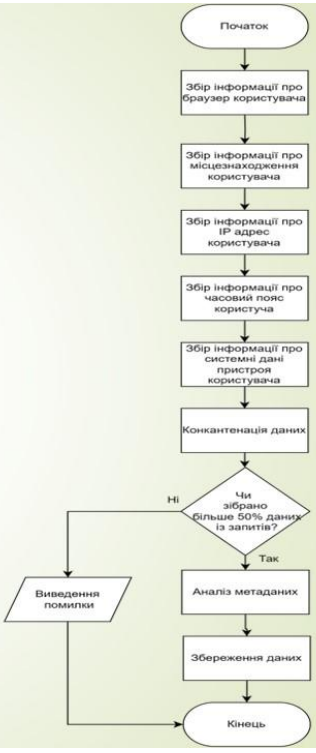
– покращена безпека;

– захист від атак на перехоплення паролів;

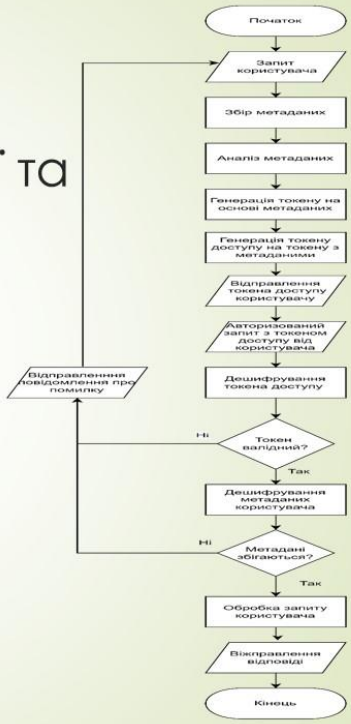
– зменшення ризику фішингу;

– додатковий шар безпеки для критичних систем.

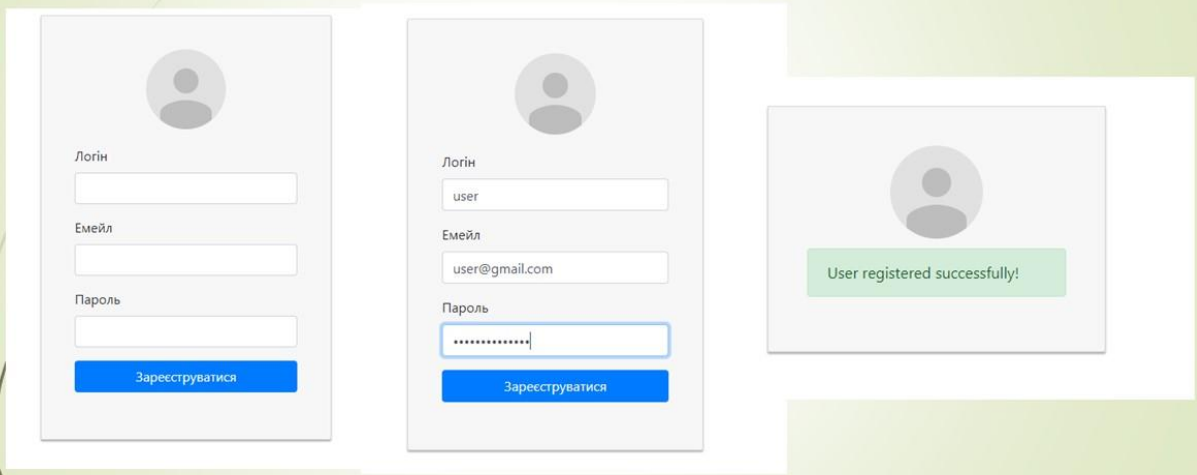
Алгоритм збору та аналізу метаданих



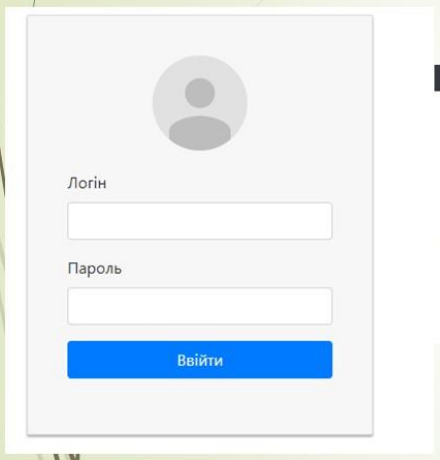
Алгоритм двоетапної генерації та перевірки токенів

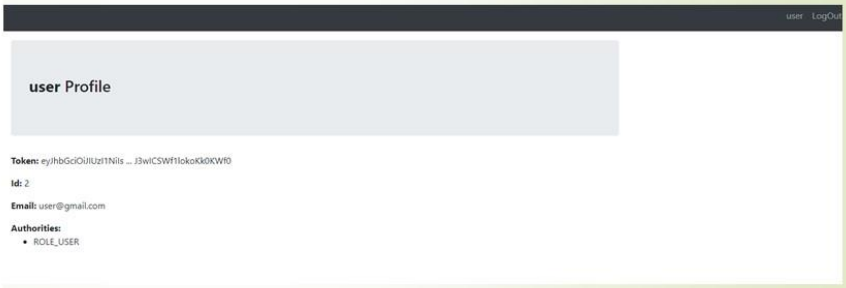


Інтерфес додатку

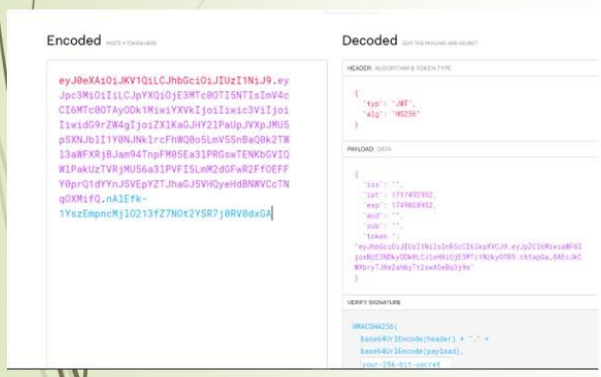


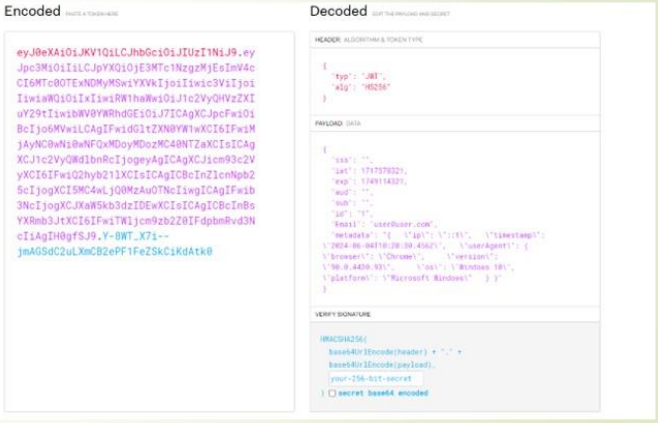
Інтерфес додатку





Перевірка токенів двоетапної генерації





Висновки

- У роботі проведено дослідження та розробку вдосконаленої системи підвищення захищеності авторизації користувачів, яка базується на двоетапній генерації та перевірці токенів доступу з використанням метаданих користувача. Здійснений аналіз сучасних методів авторизації показав, що традиційні підходи, такі як паролі та одноразові коди, вже не забезпечують необхідного рівня захисту в умовах сучасних кіберзагроз, що зумовлює необхідність розробки нових методів, здатних ефективно протистояти різноманітним видам атак.
- Проектування архітектури системи дозволило створити комплексну та ефективну систему підвищення захищеності авторизації користувачів. Запропонована архітектура включає механізми збору та аналізу метаданих, а також алгоритми генерації та перевірки токенів доступу.
- Програмна реалізація розробленої системи продемонструвала можливість її успішного впровадження в різноманітні інформаційні системи. Результати тестування показали високу ефективність та надійність запропонованого рішення.

Дякую за увагу!

ПРОТОКОЛ
ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ
НА НАЯВНІСТЬ ТЕКСТОВИХ
ЗАПОЗИЧЕНЬ

Назва роботи: Підвищення захищеності авторизації користувачів вдосконаленою системою з двоетапною генерацією та перевіркою токенів доступу на основі метаданих користувача

Тип роботи: магістерська кваліфікаційна робота
(БДР, МКР)

Підрозділ: Кафедра менеджменту та безпеки інформаційних систем
Факультет менеджменту та інформаційної безпеки
(кафедра, факультет)

Показники звіту подібності Unicheck

Оригінальність 94 % Схожість 6 %

Аналіз звіту подібності (відмітити потрібне):

1. Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.

2. Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її виконання автором. Роботу направити на розгляд експертної комісії кафедри.

3. Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Особа, відповідальна за перевірку _____ Коваль Н.П.
(підпис) (прізвище, ініціали)

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck щодо роботи.

Автор роботи _____ Грабар Р.І.
(підпис) (прізвище, ініціали)

Керівник роботи _____ Яремчук Ю.Є.
(підпис) (прізвище, ініціали)