Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії Кафедра захисту інформації

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Система протидії кіберзлочинності в ігровій індустрії»

Виконав: студент 2 курсу, групи 1БС-23м спеціальності 125 Кібербезпека та захист інформації
_____ Іван ЛАЗУРЕНКО
Керівник: д. т. н., проф., завідувач каф. ЗІ
_____Володимир ЛУЖЕЦЬКИЙ
«13» 12 2024 р.
Опонент: д. т. н., проф., завідувач каф. ПЗ
_____ Олександр РОМАНЮК
«13» 12 2024 р.

Допущено до захисту
Завідувач кафедри ЗІ
д. т. н., проф.
_____Володимир ЛУЖЕЦЬКИЙ
«13» 12 2024 р

Вінниця ВНТУ - 2024 року

Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра захисту інформації
Рівень вищої освіти II (магістерський)
Галузь знань – 12 Інформаційні технології
Спеціальність – 125 Кібербезпека та захист інформації
Освітньо-професійна програма – Безпека інформаційних і комунікаційних систем

ЗАТВЕРДЖУЮ
Завідувач кафедри ЗІ, д. т. н., проф.
_____ Володимир ЛУЖЕЦЬКИЙ
«18» 09 2024 року

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

Лазуренку Івану Дмитровичу

1. Тема роботи: «Система протидії кіберзлочинності в ігровій індустрії» Керівник роботи: Лужецький Володимир Андрійович, завідувач кафедри ЗІ, д. т. н., проф.
затверджені наказом ректора ВНТУ від 17 вересня 2024 року №310.
2. Строк подання студентом роботи 13 грудня 2024 р.
3. Вихідні дані до роботи:
   — клієнт-серверна гра;
   — засіб для здійснення кіберзлочину – чіт-система;
   — сукупність ігрових даних;
   — мова програмування – C#;
   — середовище розробки – Visual Studio.
4. Зміст текстової частини: Вступ. 1. Аналіз кіберзлочинів в ігровій індустрії та методів протидії їм. 2. Розробка методів протидії кіберзлочинності в ігровій індустрії 3. Розробка та тестування програмного засобу для протидії кіберзлочинності 4. Економічна частина. Висновки. Список використаних джерел.
5. Перелік ілюстративного матеріалу: Метод 1 обмеження доступу до ігрових даних. Метод 2 обмеження доступу до ігрових даних. Метод автентифікації користувача на основі характеристик апаратної складової. Алгоритм формування HMAC. Метод автентифікації ігрових даних. Результати автентифікації користувача. Результати автентифікації ігрових даних.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Завідувач кафедри ЗІ, д. т. н., проф. Володимир ЛУЖЕЦЬКИЙ	21.09.24	26.09.24
2	Завідувач кафедри ЗІ, д. т. н., проф. Володимир ЛУЖЕЦЬКИЙ	01.10.24	15.10.24
3	Завідувач кафедри ЗІ, д. т. н., проф. Володимир ЛУЖЕЦЬКИЙ	10.24	01.11.24
4	к.т.н., доцент, доцент каф. ЕПВМ Ольга РАТУШНЯК	06.11.24	14.11.24

7. Дата видачі завдання 17 вересня 2024 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз завдання. Вступ	18.09.2024–20.09.2024	
2	Аналіз кіберзлочинів в ігровій індустрії та методів протидії їм	21.09.2024–30.09.2024	
3	Розробка методів протидії кіберзлочинності в ігровій індустрії	1.10.2024– 15.10.2024	
4	Розробка програмного засобу для протидії кіберзлочинності	16.10.2024– 27.10.2024	
5	Тестування розробленого засобу	28.10.2024– 04.11.2024	
6	Розробка економічної частини	06.11.2024– 14.11.2024	
9	Оформлення пояснювальної записки	25.11.2024– 4.12.2024	
10	Попередній захист та доопрацювання МКР	28.11.2024– 01.12.2024	
11	Перевірка МКР на наявність плагіату	02.12.2024– 14.12.2024	
12	Представлення МКР до захисту	11.12.2024– 14.12.2024	
13	Захист МКР	17.12.2024– 20.12.2024	

Студент _____ Іван ЛАЗУРЕНКО

Керівник роботи _____ Володимир ЛУЖЕЦЬКИЙ

АНОТАЦІЯ

УДК 004.056.794

Лазуренко І. Система протидії кіберзлочинності в ігровій індустрії. Магістерська кваліфікаційна робота зі спеціальності 125 – Кібербезпека та захист інформації, освітня програма – Безпека інформаційних і комунікаційних систем. Вінниця: ВНТУ, 2024. 92 с.

Укр. мовою. Бібліогр.: 28 назв; рис.: 11; табл.: 8.

Магістерська кваліфікаційна робота присвячена розробці методів протидії кіберзлочинності в ігровій індустрії. Підготовлено науково-дослідне та техніко-економічне обґрунтування доцільності досліджень. У роботі проведено аналіз існуючих методів протидії кіберзлочинності в ігровій індустрії та обґрунтовано вибір реалізації методу для підвищення рівня захисту ігрових даних. Розроблено програмний засіб, що емулює процес формування пакету даних для передачі між користувачем та сервером.

Ілюстративна частина складається з 7 плакатів з демонстрацією методів та алгоритмів.

В економічному розділі оцінено витрати на розробку.

Ключові слова: ігрова індустрія, кіберзлочинність, чіт-системи, автентифікація, HMAC, шифрування, AES, SHA-256, геш-значення.

ABSTRACT

Lazurenko I. System for preventing cyber malware in the gaming industry. Master's qualification in specialties 125 – Cybersecurity and information protection, software – Security of information and communication systems. Vinnytsia: VNTU, 2024. 92 p.In Ukrainian. Bibliography: 28 titles; Fig.: 11; Table: 8.

The master's qualification work is devoted to the development of methods of combating cybercrime in the gaming industry. A scientific research and feasibility study of the research has been prepared. The work analyzes existing methods of combating cybercrime in the gaming industry and justifies the choice of implementing a method to increase the level of protection of gaming data. A software tool has been developed that emulates the process of forming a data packet for transmission between the user and the server.

The illustrative part consists of 7 posters demonstrating methods and algorithms.

The economic section estimates the development costs.

Keywords: gaming industry, cybercrime, cheat systems, authentication, NMAS, encryption, AES, SHA-256, hash value.

ЗМІСТ

ВСТУП

Кіберзлочинність у сучасному світі набуває все більшої актуальності через стрімкий розвиток цифрових технологій. З кожним роком масштаби діяльності зловмисників збільшуються, охоплюючи нові сфери, що робить боротьбу з кіберзлочинами складнішою та багатоаспектною. Серед численних галузей, що зазнають впливу кіберзагроз, особливу увагу привертає ігрова індустрія. Це пов’язано з унікальними характеристиками цієї сфери: активною взаємодією користувачів, значними фінансовими операціями, великою кількістю персональних даних, що обробляються, та популярністю цифрових активів.

Мільйони людей щоденно проводять час на ігрових платформах, інвестуючи кошти у внутрішньоігрові товари, отримуючи емоційне задоволення та формуючи онлайн-спільноти. Проте це також робить їх об'єктами уваги кіберзлочинців, які використовують вразливості платформ для здійснення шахрайських дій, викрадення конфіденційної інформації чи навіть для незаконного отримання доступу до платіжних даних користувачів. Унаслідок цього зростає загроза для економічної безпеки індустрії, репутації компаній-розробників і, найголовніше, для приватного життя гравців.

Ключовими викликами кіберзлочинності в ігровій сфері є:

Шахрайство. Кіберзлочинці розробляють складні схеми, спрямовані на маніпуляцію ігровими механізмами для отримання нечесної переваги. Наприклад, це можуть бути випадки продажу підроблених ігрових предметів, використання шкідливого програмного забезпечення для змін у грі або поширення фальшивих пропозицій на платформах для мікротранзакцій.

Злом акаунтів. Часто зловмисники зламують акаунти гравців, отримуючи доступ до цифрових активів, придбаних за реальні гроші, та використовують ці дані для фінансових шахрайств або подальшого продажу викрадених облікових записів.

Крадіжка персональних даних. Оскільки ігрові компанії збирають значну

кількість персональної інформації про своїх користувачів, ця інформація стає привабливою ціллю для хакерів. Викрадені дані можуть використовуватись для створення фальшивих профілів, шантажу чи інших нелегальних дій.

У відповідь на ці загрози ігрова індустрія впроваджує сучасні технології та інструменти для забезпечення кібербезпеки. Основні напрями роботи включають:

Аутентифікація та ідентифікація. Використання багатофакторної аутентифікації (MFA) стає одним із базових способів захисту акаунтів користувачів. Крім того, активно впроваджуються біометричні методи, які значно підвищують рівень безпеки.

Моніторинг активності та виявлення аномалій. Використання штучного інтелекту (ШІ) для аналізу активності користувачів допомагає виявляти підозрілу поведінку, таку як нетипові транзакції чи аномально висока кількість запитів із одного акаунта.

Захист мережевої інфраструктури. Установка сучасних систем виявлення та запобігання вторгненням (IDS/IPS), а також регулярні оновлення програмного забезпечення дозволяють мінімізувати ризики експлуатації вразливостей.

Криптографія та шифрування. Для забезпечення конфіденційності даних використовуються передові алгоритми шифрування, які ускладнюють доступ до інформації для зловмисників навіть у разі компрометації даних.

Освітні програми для користувачів. Регулярне інформування гравців про потенційні загрози, а також навчання їх правилам безпечної поведінки в онлайн-просторі зменшують ризики шахрайства через людський фактор.

У межах цього дослідження буде детально проаналізовано кожен із зазначених аспектів, розглянуто практичні кейси, пов’язані з боротьбою з кіберзагрозами, а також оцінено ефективність різних підходів до захисту ігрових платформ.

Метою даної роботи є аналіз поточних кіберзагроз, які мають безпосередній вплив на розвиток і функціонування ігрової індустрії, та

визначення інноваційних методів протидії. Зокрема, дослідження буде зосереджено на вивченні новітніх технологій і стратегій, що використовуються для мінімізації ризиків як для компаній, так і для користувачів.

Завдання дослідження

1. Виявлення ключових типів кіберзлочинів, що загрожують ігровій індустрії, їх характеристик та особливостей.

2. Аналіз поточного стану захисту ігрових платформ, зокрема інструментів та технологій, що застосовуються.

3. Визначення основних вразливостей, які залишаються відкритими для атак, та способів їх усунення.

4. Розробка рекомендацій для ігрових компаній щодо покращення кібербезпеки та формування комплексного підходу до захисту даних.

Результати цього дослідження можуть стати базою для подальшої оптимізації механізмів захисту ігрових платформ, а також для підвищення обізнаності розробників і користувачів щодо важливості дотримання стандартів кібербезпеки.

Наукова новизна даної магістерської роботи полягає у використанні методів автентифікації даних та залежності від характеристик апаратного забезпечення, що надає додаткову унікальність при створенні секретних ключів та можливість блокування не тільки ігрового акаунту, а й весь комп’ютер.

1. АНАЛІЗ КІБЕРЗЛОЧИНІВ В ІГРОВІЙ ІНДУСТРІЇ ТА МЕТОДІВ ПРОТИДІЇ ЇМ

1.1 Огляд ігрової індустрії: масштаби, тенденції та розвиток

Ігрова індустрія на сьогодні є однією з найбільш швидкозростаючих і впливових галузей світової економіки. За даними аналітичних компаній, ринок відеоігор за останні кілька років перевищив 150 мільярдів доларів і продовжує демонструвати динамічне зростання. Відеоігри більше не є виключно розвагою для молоді — сучасна аудиторія охоплює людей різного віку, соціального статусу та національності. Це, своєю чергою, зробило ігрову індустрію важливою не лише в економічному, а й у соціальному аспекті.

Розвиток цифрових технологій призвів до кардинальних змін у геймінгу: з'явилися нові жанри ігор, такі як VR (віртуальна реальність), AR (доповнена реальність), кіберспорт, мобільні ігри. З цим зростає і кількість користувачів, які взаємодіють з цифровими продуктами щодня. За оцінками експертів, кількість активних гравців у світі досягає майже 3 мільярдів осіб, що робить ігрову індустрію однією з найбільших за кількістю споживачів.

Масштаби ринку та його учасники.

Основними учасниками ринку є великі ігрові корпорації (наприклад, Sony, Microsoft, Nintendo), розробники програмного забезпечення (Ubisoft, Electronic Arts, Blizzard Entertainment), платформи цифрової дистрибуції (Steam, Epic Games Store) та безліч незалежних студій розробки. Кожен з цих сегментів має свою роль і внесок у розвиток галузі [1].

Консольні ігри. Традиційний сегмент ігрової індустрії, представлений гігантами, такими як Sony PlayStation та Microsoft Xbox. Консольні ігри забезпечують багатий ігровий досвід завдяки потужному апаратному забезпеченню [2].

Мобільні ігри. З популяризацією смартфонів та планшетів цей ринок виріс у кілька разів за останнє десятиліття. Основними гравцями тут є компанії, що розробляють ігри для мобільних платформ, такі як King (Candy Crush), Supercell (Clash of Clans) та Tencent [3].

PC-геймінг. Персональні комп'ютери залишаються важливим інструментом для ігор завдяки своїй універсальності та доступу до потужних ігрових платформ, таких як Steam, що пропонує мільйони різних ігор для широкої аудиторії [4].

Кіберспорт. Один з найбільш швидкозростаючих сегментів індустрії. Кіберспорт з кожним роком набирає популярність, створюючи багатомільйонні призові фонди для турнірів з таких ігор, як League of Legends, Dota 2 та Counter-Strike.

Технологічний прогрес в індустрії.

Збільшення потужності процесорів, вдосконалення графічних систем та розвиток інтернет-технологій дозволили зробити геймінг більш інтерактивним та реалістичним. Окрім цього, зростання інтересу до технологій доповненої та віртуальної реальності стимулює інвестиції у розвиток нових способів занурення у гру. Ігри, що використовують VR та AR, надають гравцям можливість пережити унікальні враження, які раніше були недосяжними.

Також великий вплив на індустрію мали хмарні технології, які дозволяють запускати ігри через інтернет без необхідності мати потужне апаратне забезпечення на кінцевому пристрої. Такі сервіси, як Google Stadia, GeForce Now та Xbox Cloud Gaming, відкрили нові горизонти для користувачів, надаючи доступ до преміум-класу ігор з будь-якого пристрою [5].

Тенденції розвитку.

Ігрова індустрія постійно змінюється, адаптуючись до нових викликів. Ось кілька ключових тенденцій, що визначають її розвиток:

Глобалізація ринку. Сьогодні ігрова індустрія не має меж — користувачі можуть грати в багатокористувацькі ігри в реальному часі з людьми з усього світу. Це створює нові можливості для розробників та платформ.

Кіберспорт та стрімінгові платформи. Кіберспорт стає професійним видом спорту, що приваблює мільйони глядачів та сотні мільйонів доларів інвестицій. Стрімінгові платформи, такі як Twitch і YouTube Gaming, є важливими каналами для трансляції ігрового контенту та залучення нових користувачів.

Мікротранзакції та моделі монетизації. Багато сучасних ігор базуються на фріміум-моделі, де гра є безкоштовною, але гравці можуть витрачати реальні гроші на додатковий контент, що створює нові економічні моделі та ризики, пов'язані з безпекою [6].

Інтеграція з соціальними мережами. Ігри все більше стають соціальними платформами, де гравці взаємодіють один з одним, діляться досягненнями та співпрацюють. Це підвищує їхню привабливість, але одночасно робить їх уразливими до кіберзагроз.

Вплив ігрової індустрії на інші галузі.

Ігрова індустрія також впливає на розвиток інших секторів економіки. Технології, що використовуються в геймінгу, знаходять застосування в навчальних програмах, медичних симуляторах, тренінгах для професіоналів та у сферах розваг, таких як кіноіндустрія та віртуальні тури.

Ігри стали важливим інструментом для навчання, розвитку навичок та соціальної інтеграції. Багато навчальних програм використовують гейміфікацію для покращення навчального процесу, оскільки це підвищує мотивацію студентів та залученість до навчання.

Ігрова індустрія не тільки розвивається шаленими темпами, але й створює нові виклики, зокрема у сфері кібербезпеки. З кожним роком вона стає більш складною, інтегруючи новітні технології та залучаючи все більше користувачів з усього світу. Однак, разом з цим зростає і кількість кіберзагроз, які роблять

гравців та платформи вразливими до атак, що робить питання протидії кіберзлочинності одним із ключових у сучасному цифровому світі [7].

1.2 Основні типи кіберзлочинів в ігровій індустрії

Ігрова індустрія стає привабливим об'єктом для кіберзлочинців через велике залучення користувачів, значні обсяги фінансових операцій та величезні бази даних з особистими даними. Віртуальні світи та цифрові активи відкривають нові можливості для різних видів шахрайства, що стає викликом для розробників і компаній у галузі. Кіберзлочинці користуються різними техніками та інструментами для отримання незаконного прибутку, і кожен тип атаки має свої особливості та наслідки.

Соціальна інженерія є одним з найпоширеніших методів атаки в ігровій індустрії. Мета таких атак – ввести користувача в оману та змусити надати свою конфіденційну інформацію або перейти за шкідливим посиланням.

Фішинг [8].

– Кіберзлочинці часто використовують фальшиві сайти та повідомлення, що імітують офіційні ресурси, наприклад, відомі платформи для онлайн-ігор або магазини контенту, такі як Steam чи Epic Games Store. Жертвам пропонують ввести облікові дані або завантажити підозрілий файл, який згодом заражає пристрій шкідливим програмним забезпеченням.

– Імітація облікових записів. Кіберзлочинці можуть створювати підроблені акаунти, що імітують відомих гравців або навіть представників компанії, з метою отримання доступу до акаунтів інших користувачів.

– Фішинг в соціальних мережах. Оскільки багато гравців взаємодіють у соціальних мережах, кіберзлочинці активно використовують ці платформи для фішингових атак. Вони надсилають повідомлення з пропозиціями знижок, бонусів або "подарунків" в обмін на облікові дані.

Зі зростанням обсягів транзакцій в ігрових системах кіберзлочинці почали активно полювати на платіжні дані користувачів. Ігри часто використовують

мікротранзакції, підписки та інші форми монетизації, що відкриває можливості для викрадення платіжних даних.

– Викрадення банківських даних. Шахраї можуть отримати доступ до банківських карток або облікових записів у платіжних системах користувачів, використовуючи шкідливе програмне забезпечення або фішингові атаки. Викрадені дані часто використовують для покупок у самій грі або продажу на чорному ринку.

– Підробка внутрішньоігрових транзакцій. Деякі зловмисники вміють обходити системи захисту внутрішніх транзакцій, створюючи нелегальні об'єкти або валюти, які згодом продаються на спеціалізованих майданчиках.

– Фейкові сайти з продажу ігрових товарів. Сайти, що імітують офіційні магазини або ринки цифрових товарів, можуть обманювати користувачів, збираючи їхні платіжні дані для подальшого використання у шахрайських цілях.

Крадіжка акаунтів стала однією з найпоширеніших загроз в ігровій індустрії. Акаунти гравців часто містять рідкісні ігрові предмети, внутрішньоігрові валюти або досягнення, які можуть бути перепродані на чорному ринку.

Брутфорс [9].

– Один з найпростіших методів зламу облікових записів — брутфорс, при якому автоматизовані системи намагаються зламати пароль, використовуючи всі можливі комбінації. Цей метод є особливо ефективним проти слабких або повторно використовуваних паролів.

– Шкідливе програмне забезпечення. Віруси та трояни, зокрема кейлогери, можуть встановлюватися на комп'ютери користувачів та відстежувати їхні дії. Це дає змогу кіберзлочинцям отримати доступ до введених логінів та паролів, а також до інших конфіденційних даних.

– Викрадення ігрових предметів та валюти. Після зламу акаунту зловмисники часто продають ігрові активи на вторинному ринку або

використовують їх у інших шахрайських схемах. Популярні об’єкти та валюти мають високу цінність на чорному ринку, де вони продаються за реальні гроші.

– DDoS-атаки є ще одним видом кіберзлочину, який має значний вплив на ігрову індустрію. Мета такої атаки – перевантажити сервери гри, що призводить до відмови в обслуговуванні для звичайних користувачів. Це може спричинити значні втрати як для гравців, так і для розробників ігор.

– Зниження якості обслуговування. DDoS-атаки роблять гру недоступною для гравців або значно знижують її продуктивність. Це спричиняє негативні відгуки користувачів, що впливає на репутацію компанії та знижує її прибутки.

– Шантаж розробників. Кіберзлочинці можуть використовувати DDoS-атаки для шантажу розробників, вимагаючи викуп за припинення атаки. В ігровій індустрії це може призвести до значних втрат, особливо якщо атака відбувається під час великих релізів чи оновлень.

– Ураження конкурентів. Іноді DDoS-атаки використовуються як засіб боротьби між конкурентами. Окремі компанії можуть використовувати подібні методи для підриву іміджу суперників, створюючи проблеми з доступом до їхніх серверів.

– Онлайн-ігри, зокрема багатокористувацькі, часто стикаються з проблемою шахрайства серед гравців. Ці шахраї можуть використовувати скрипти, боти та спеціальне програмне забезпечення для отримання несправедливої переваги над іншими гравцями, що руйнує баланс гри та знижує інтерес до неї [10].

– Використання ботів. Деякі гравці використовують автоматизовані скрипти або боти, щоб здійснювати рутинні дії, що дозволяє їм швидше досягти прогресу в грі. Таке шахрайство є особливо популярним у мобільних іграх та MMORPG, де багато задач є циклічними.

– Альтернативні акаунти та "підкидання" рейтингу. У деяких іграх гравці можуть створювати альтернативні акаунти або використовувати шахрайські

методи для підвищення рейтингу основного акаунта. Це спотворює систему рейтингу та може вплинути на враження від гри для інших гравців.

– Використання "читів" та експлойтів. Шахраї знаходять та використовують уразливості у грі, що дозволяє їм отримати несправедливу перевагу над іншими гравцями, як-от безсмертя, швидше пересування, або посилення зброї. Це негативно впливає на ігровий досвід чесних гравців та може призвести до зниження кількості активних користувачів.

Кіберзлочини в ігровій індустрії є серйозною загрозою, яка впливає на всі аспекти гри — від безпеки користувачів до економічної стабільності компаній. Різноманітність кіберзагроз вимагає від розробників створення надійних систем захисту, а від користувачів — підвищення їх обізнаності про можливі ризики. Оскільки технології та злочинні методи постійно вдосконалюються, індустрія потребує комплексного підходу до захисту ігрового середовища та активної боротьби з кіберзлочинцями.

1.3 Відомі методи протидії кіберзлочинам

Кіберзлочинність у ігровій індустрії залишається серйозною загрозою, що потребує комплексного підходу до захисту та боротьби із зловмисниками. Компанії та платформи, які займаються розробкою ігор, використовують різноманітні технології для захисту своїх продуктів і користувачів. У цьому розділі розглянемо найпоширеніші методи протидії кіберзлочинності, їхні переваги та недоліки.

Методи автентифікації та шифрування даних.

Одним із головних підходів для запобігання несанкціонованому доступу до облікових записів користувачів та захисту їхньої особистої інформації є використання надійних методів автентифікації та шифрування.

1. Двофакторна аутентифікація (2FA). Двофакторна аутентифікація є ефективним методом захисту облікових записів, який вимагає підтвердження особи користувача за допомогою другого фактору (наприклад, SMS-коду або

генератора кодів). Одними із найпопулярніших сервісів котрі реалізують двофакторну автентифікацію є:

2. Google Authenticator: безкоштовний додаток для двофакторної автентифікації, що широко використовується у різних платформах, зокрема ігрових сервісах.

3. Authy: інший популярний додаток 2FA, що дозволяє синхронізувати коди на різних пристроях та забезпечує додатковий рівень безпеки [11].

4. Steam Guard: система двофакторної автентифікації від Steam, що дозволяє захистити обліковий запис користувача за допомогою коду, який надсилається на мобільний додаток [12].

Переваги:

– Значно підвищує рівень безпеки акаунта, оскільки захист не обмежується лише паролем.

– Зменшує ризик зламу акаунта при використанні слабкого або вкраденого пароля.

Недоліки:

– Не всі користувачі готові використовувати додаткові засоби підтвердження, що може знизити зручність входу.

– При втраті доступу до другого фактору (наприклад, телефону) користувач може зіткнутися з труднощами під час входу.

Шифрування даних.

Використання шифрування для захисту персональних даних гравців та фінансової інформації є стандартним методом захисту, який робить інформацію недоступною для неавторизованих осіб. Для шифрування найчастіше використовуються такі програми:

1. OpenSSL: бібліотека з відкритим вихідним кодом, яка забезпечує криптографічні функції для шифрування та захисту даних, що широко використовується у різних онлайн-іграх та платформах [13].

2. VeraCrypt: інструмент для шифрування даних на рівні файлів та папок, який може бути інтегрований у внутрішні системи для шифрування конфіденційної інформації.

Переваги:

– Навіть якщо дані будуть викрадені, їх неможливо прочитати без ключа дешифрування.

– Захищає особисту інформацію користувачів від втрат та зловживань.

Недоліки:

– Процес шифрування та дешифрування може підвищувати навантаження на сервери, що збільшує затримки у великих іграх з високою кількістю користувачів.

– Ризик компрометації ключів шифрування, що відкриває доступ до зашифрованих даних.

Антивірусні та антишпигунські програми.

Антивірусне та антишпигунське програмне забезпечення допомагає виявляти та блокувати шкідливі програми, які можуть впливати на ігровий процес або викрадати дані користувачів.

Переваги:

– Виявляє та блокує потенційно небезпечні програми, зменшуючи ризик втрати персональних даних.

– Допомагає знизити активність ботів та інших шкідливих програм, що покращує якість ігрового досвіду.

Недоліки:

– Потребує регулярного оновлення, щоб бути ефективним проти нових вірусів та троянів.

– Може призводити до зниження продуктивності, особливо на менш потужних пристроях, через високе навантаження на систему.

Використання технологій штучного інтелекту (ШІ) для виявлення підозрілої активності.

ШІ дозволяє створювати системи, які аналізують поведінку гравців і здатні виявляти відхилення, які можуть свідчити про шахрайство або спроби зловмисників отримати доступ до облікових записів.

Переваги:

– ШІ здатний виявляти нові види атак та розпізнавати шаблони, що раніше не фіксувалися.

– Системи на основі ШІ можуть працювати в реальному часі, що дозволяє швидко реагувати на потенційні загрози.

Недоліки:

– Навчання та вдосконалення ШІ можуть бути дороговартісними процесами, що потребують значних ресурсів.

– Можливі помилкові спрацьовування, коли система блокує звичайну активність як потенційно шкідливу, що може призвести до негативного досвіду користувачів.

Брандмауери та системи запобігання вторгненням (IPS).

Брандмауери та IPS контролюють трафік, що надходить до серверів, і можуть блокувати підозрілу активність, захищаючи мережеву інфраструктуру від атак, таких як DDoS.

Переваги:

– Захищають сервери та інфраструктуру від атак, що здійснюються через мережеві підключення.

– Знижують ризик недоступності сервісів через атаки типу "відмова у доступі".

Недоліки:

– Брандмауери потребують налаштування і моніторингу, оскільки при неправильних налаштуваннях можуть блокувати законний трафік.

– У разі великої кількості DDoS-атак можуть перевантажувати систему захисту, особливо якщо ресурси сервера обмежені.

Методика захисту від ботів та античітинг-програми.

Онлайн-ігри особливо вразливі до ботів, які автоматизують ігровий процес, та читерів, що використовують програми для отримання переваги. Античит-програми допомагають виявити та заблокувати шахраїв.

Системи захисту від ботів. Блокують ботів, що виконують автоматизовані дії, наприклад, автоматичний фарм або виконання квестів.

Переваги:

– Захищає від некоректного використання ігрового контенту та забезпечує справедливі умови гри.

– Сприяє збереженню позитивного досвіду чесних гравців, що допомагає залучати нових користувачів.

Недоліки:

– Деякі системи можуть помилково ідентифікувати звичайних гравців як ботів, що викликає незадоволення.

– Складність у виявленні нових видів ботів, що можуть адаптуватися до механізмів захисту.

Античітинг-програми. Використовуються для виявлення та блокування програм, що надають переваги в грі (безсмертя, швидкість, покращення зброї тощо).

Переваги:

– Дозволяє забезпечити чесну гру, що є важливим для збереження балансу у багатокористувацьких онлайн-іграх.

– Блокує доступ до гри читерам, що стимулює чесну конкуренцію між гравцями.

Недоліки:

– Може впливати на продуктивність гри, оскільки потребує додаткових ресурсів для моніторингу.

– Іноді неправильно ідентифікує читерські програми, що призводить до випадкових блокувань звичайних гравців.

Серед популярних рішень, що надають захист від ботів та чіт-програм можна виділити наступні рішення:

1. Easy Anti-Cheat: широко використовувана античітинг-програма, що ефективно бореться з шахрайством у багатокористувацьких іграх, таких як Fortnite, Apex Legends та інші [14]. BattleEye: популярний античіт, який інтегровано в ігри, такі як PlayerUnknown's Battlegrounds (PUBG) та Rainbow Six Siege, для боротьби з читами та нелегальною активністю [15].

2. VAC (Valve Anti-Cheat): система, розроблена Valve, інтегрована в ігрову платформу Steam, що автоматично виявляє підозрілі програми та блокує доступ до гри.

Моніторинг та аналіз активності користувачів.

Методи моніторингу активності користувачів дозволяють розпізнавати шаблони поведінки, що можуть свідчити про шахрайські дії або порушення правил гри. Для моніторингу активності користувача часто використовують:

1. FairFight: система моніторингу та аналізу поведінки, що працює в реальному часі та виявляє підозрілі дії гравців за шаблонами, використовується в Battlefield, Tom Clancy's The Division.

2. IBM QRadar: платформа безпеки, що використовується для моніторингу активності та виявлення аномалій у поведінці користувачів, що може бути адаптована для ігрових платформ.

3. Splunk: програмне забезпечення для моніторингу та аналізу даних у реальному часі, яке дозволяє виявляти та аналізувати аномальні дії, потенційно пов'язані з кіберзлочинами [16].

Переваги:

– Дозволяє швидко ідентифікувати підозрілу поведінку, що може свідчити про спроби злому або використання шахрайських методів.

– Покращує безпеку гри, допомагаючи виявити ризиковані патерни дій.

Недоліки:

– Потребує високого рівня конфіденційності даних, що вимагає дотримання законодавчих норм, таких як GDPR.

– Може викликати обурення у користувачів через можливе втручання у їхню приватність.

Сучасні методи протидії кіберзлочинам дозволяють зменшити ризики зламу акаунтів, викрадення даних та маніпулювання ігровим процесом. Кожен метод має свої переваги та недоліки, тому для ефективного захисту ігрового середовища необхідно використовувати комплексний підхід, поєднуючи декілька методів для максимального ефекту. Збалансована стратегія захисту допомагає забезпечити комфортне і безпечне середовище для гравців, одночасно зберігаючи лояльність користувачів до продукту.

1.4 Статистика поширення читів в ігровій індустрії

Ігрова індустрія на сьогодні продовжує активно розвиватися, але разом з тим загострюється і проблема читів у багатокористувацьких іграх. Користувачі читів використовують різні методи, щоб отримати перевагу, зокрема аімаботи, ESP (система візуалізації ворогів) і спуфери (приховують справжній ID гравця). Чітінг призводить до значних втрат для індустрії, оскільки воно погіршує користувацький досвід та негативно впливає на імідж розробників.

Статистика поширення читів в іграх.

У 2023-2024 роках проблема чітінгу набула нових масштабів. За даними досліджень, понад 30% геймерів зіштовхувалися з чітингом у популярних онлайн-іграх, таких як Apex Legends та Escape from Tarkov. У звіті GameAnalytics йдеться про збільшення кількості заборонених акаунтів через читі: PUBG заблокувала понад 1 мільйон акаунтів, а Apex Legends регулярно проводить бан-хвилі, під час яких заблоковано десятки тисяч акаунтів. Водночас, використання хаків та ботів у деяких іграх збільшується на 5-10% щороку.

Види читів та їх вплив.

Основні види читів, які фіксуються сьогодні, включають:

– Аімботи: допомагають у точному прицілюванні на ворогів.

– ESP/Wallhack: дозволяє бачити ворогів крізь стіни.

– Спуфери: запобігають блокуванням через підміну даних пристрою.

– Макроси: автоматизують дії, що дає перевагу в швидкості.

Ці види чітів особливо поширені в іграх-шутерах та іграх на виживання, де точність і швидкість реакції грають вирішальну роль. Це впливає на баланс ігор, через що користувачі часто відмовляються від гри або вимагають більш активних заходів від розробників. Біль детальну статистичну інформацію з приводу кількості заблокованих користувачів та видів чітів наведено в табл. 1.1.

Боротьба з чітами: технології та стратегії.

Таблиця 1.1 – Статистика заблокованих гравців та типів чітів в популярних іграх

Гра	**Кількість заблокованих акаунтів (2024)**	**Популярні типи чітів**	**Античіт-системи**
Apex Legends	120,000+	Аімбот, ESP, спуфер	Easy Anti-Cheat, власний
Escape from Tarkov	90,000+	Макроси, ESP, спуфер	BattleEye, власний
PUBG	1,000,000+	ESP, аімбот	BattleEye
Call of Duty: MWII	250,000+	Аімбот, ESP, спуфер	RICOCHET
Fortnite	500,000+	Аімбот, ESP, макроси	Easy Anti-Cheat
Valorant	50,000+	Аімбот, спуфер	Vanguard
Counter-Strike 2	700,000+	Аімбот, ESP, спуфер	VAC
Rainbow Six Siege	130,000+	ESP, макроси	BattleEye

Компанії продовжують розробляти нові античіт-системи, щоб відслідковувати і блокувати нечесних гравців. Наприклад, BattleEye та Easy Anti-Cheat використовують поведінкові моделі для виявлення аномальної активності

гравців. Окрім того, компанії активно використовують штучний інтелект для прогнозування й запобігання новим видам чітів. За даними GameAnalytics, лише у 2024 році інвестиції в античіт-технології зросли на 15% порівняно з попереднім роком.

Джерела статистичної інформації:

1. Apex Legends та Call of Duty – Game Analytics Report .

2. PUBG – Звіти PUBG Corp. на офіційних каналах та новини з Polygon [17].

3. Escape from Tarkov – Дані з форумів та офіційних соціальних мереж Escape from Tarkov [18].

4. Fortnite – Статистичні дані з античіт-системи Easy Anti-Cheat та огляди Techopedia [19].

5. Valorant – Офіційна статистика від Riot Games на Riot Games [20].

6. Counter-Strike 2 – Оновлення VAC та статистика на Steam [21].

7. Rainbow Six Siege – Новини Ubisoft на офіційних каналах та звіти на Ubisoft [22].

Економічні наслідки чітінгу.

Збитки від чітів в індустрії сягають сотень мільйонів доларів. Компанії втрачають через зниження рівня задоволення користувачів, особливо у тих іграх, де багато користувачів чітів. Деякі гравці залишають гру, не витримуючи постійного нечесного суперництва, що призводить до втрат через нездійснені покупки. Гравці, які бачать регулярні блокування користувачів чітів, більше схильні довіряти розробникам і витрачати на ігровий контент.

Отже, боротьба з чітінгом залишається важливим напрямом для ігрової індустрії, і в 2024 році очікується подальший розвиток технологій протидії, зокрема із застосуванням штучного інтелекту і нових протоколів безпеки, щоб зменшити вплив чітингу на досвід користувачів та імідж компаній.

1.5 Висновки до розділу 1

Ігрова індустрія є привабливою для кіберзлочинців через великий обсяг фінансових операцій, баз даних користувачів та віртуальні активи. Серед найпоширеніших кіберзагроз — фішинг, викрадення облікових даних та платіжної інформації, атаки на сервери (DDoS) і внутрішньоігрові шахрайства. Кіберзлочини у цій галузі можуть призвести до значних збитків для гравців та компаній, створюючи серйозні виклики для безпеки ігрових платформ і стимулюючи розробників посилювати системи захисту.

Сучасні методи захисту в ігровій індустрії включають двофакторну автентифікацію, шифрування даних, а також системи виявлення шахрайства та захисту від DDoS-атак. Використання надійної автентифікації та шифрування підвищує захищеність облікових записів і даних гравців, а системи відстеження активності гравців дозволяють ефективніше реагувати на шахрайські дії. Однак, незважаючи на ці заходи, частина користувачів залишається вразливою через відсутність технічної обізнаності або інші завади скоєними порушниками.

Після проведення аналізу даної теми, було виявлено одну з най вразливіших областей, а саме, втручання у внутрішні файли гри. Для забезпечення більшої надійності необхідно додаткові методи шифрування на стороні клієнта та методи обробки отриманої інформації зі сторони ігрового серверу. Також необхідно розглянути можливість ізоляції частини користувача, для запобігання небажаного вручання в роботу коду та ігрового процесу в цілому.

2. РОЗРОБКА МЕТОДІВ ПРОТИДІЇ КІБЕРЗЛОЧИННОСТІ В ІГРОВІЙ ІНДУСТРІЇ

2.1 Формування загальної моделі для передачі даних

Для реалізації захисту інформації, що генерується зі сторони клієнта та відправляється на серверу необхідно побудувати модель шифрування даних. Протокол передачі інформації між клієнтом та сервером передбачає виконання таких дій:

1. Формування пакету даних.
2. Процес зашифрування пакету даних.
3. Формування HMAC.
4. Передача даних до серверу.
5. Розшифрування пакету.
6. Перевірка цілісності даних.

Сервер розшифровує отримане повідомлення на своєму секретному ключі. Ключ $К_е$ зберігається сервером в списку унікальних ключів на персональному акаунті користувача.

Пакет даних, що передається від клієнта на сервер, має такі складові:

M — вихідне повідомлення, яке складається з даних користувача, параметрів сеансу та іншої інформації, що необхідно відправити на сервер.

T — мітка часу, яка забезпечує унікальність пакета та захист від атак повторного відтворення.

HMAC – код автентифікації даних, що передаються.

Зашифрування пакету даних реалізується за алгоритмом блокового шифрування AES з використанням секретного ключа $К_е$.

HMAC формується для унікальних даних таких як, UserId, GameData та міткою часу.

Дані передаються за допомогою протоколу UDP із внутрішньою перевіркою цілісності RUDP, що забезпечує високу швидкість під час передачі.

Розшифрування пакету даних відбувається за допомогою алгоритму AES та секретного ключа $К_е$.

Після отримання пакету даних, сервер формує валсний HMAC та перевіряє його відповідність. Процес генерації HMAC відбувається на основі актульних даних з бази даних серверу та списку можливих команд для обробки.

Якщо значення HMAC збігаються, то дані автентифіковано. У випадку невідповідності, дані не збережуться і запит буде надіслано повторно. Якщо протягом певного часу, отриманий HMAC не буде співпадати з HMAC, що формується сервером, користувач, що надсилає пакети буде розглядатись вбудованою античіт-системою.

Такий пакет формується для забезпечення одночасно кількох властивостей безпеки:

– Конфіденційність: зашифроване повідомлення M гарантує, що зміст даних не буде доступним для сторонніх осіб під час передачі.

– Автентичність та цілісність: HMAC забезпечує можливість серверу перевірити, що пакет був сформований клієнтом і не був змінений.

– Захист від атак повторного відтворення: мітка часу T дозволяє серверу ідентифікувати застарілі або дубльовані пакети.

Загальна модель для передачі даних між користувачем та сервером зображено на рис. 2.1.

2.2 Розробка методу реєстрації користувача

Щоб користувач мав доступ до гри його необхідно зареєструвати. Для реєстрації користувача необхідно створити акаунт з унікальним ідентифікаторм «нікнеймом» та секретним ключем, сформованим на основі характеристик апаратного забезпечення.

Рисунок 2.1 – Загальна модель для передачі даних

Процес реєстрації користувача передбачає наступні дії:

– Відправка унікального ідентифікатора на сервер.

– Отримання відкритого ключа для зашифрування секретного ключа.

– Генерація секретного ключа.

– Зашифрування секретного ключа ключа.

– Відправка секретного ключа на сервер.

– Розшифрування секретного ключа.

Користувач самостійно вводить унікальний ідентифікатор «нікнейм» та відправляє його на сервер.

Сервер формує 2 ключі, приватний та публічнмий, для зашифрування та розшифрування та відправляє публічний ключ користувачу.

Секретний ключ $К_е$ формується на основі апаратної складової персонального комп'ютера користувача.

Транспортування секретного ключа $К_е$ на сервер відбувається з використанням шифру RSA. Клієнт зашифровує $К_е$ на відкритому ключі сервера і надсилає його серверу.

Отриманий зашифрований секретний ключ $К_е$ розшифровується приватним ключем серверу. Отриманий секретний ключ зберігається та використовується для предачі ігрових даних між користувачем та сервером.

Алгоритм реєстрації користувача представлено на рис. 2.2.

Якщо античіт-система чи сервер виявлять незаконні дії з боку користувача, можливо заблокувати не тільки ігровий акаунт, котрий можливо зареєструвати повторно витративши від 2 до 10 хвилин, а заборонити доступ до гри самій апаратній частині. Таке рішення значно ускладнить процес повторного незаконного втручання, оскільки необхідно програмно замінити серійні номери апаратної частини ПК, чи придбати нові комплектуючи.

Рисунок 2.2 – Алгоритм реєстрації користувача

2.3 Розробка методу генерування секретного ключа на основі апаратних даних

Генерування секретного ключа на основі апаратних даних забезпечує високий рівень безпеки, оскільки ключ стає залежним від характеристик фізичної машини. Такий підхід стає все більш популярним у кібербезпеці, адже забезпечує стійкість до несанкціонованого копіювання ключів, а також захищає конфіденційні дані від витоку.

Нехай S — набір унікальних ідентифікаторів апаратних компонентів комп'ютера. Даний набір включає такі параметри:

S_{CPU} — серійний номер процесора,

S_{Disk} — серійний номер жорсткого диска,

S_{MAC} — MAC-адреса мережевої карти.

Метод генерування ключа передбачає, виконання таких дій:

1. Збирання ідентифікаторів: $S = \{ S_{CPU}, S_{Disk}, S_{MAC}\}$.

2. Генерування унікального геш-значення K на основі набору ідентифікаторів, який використовується як секретний ключ:

$K = Hash (S_{CPU} \| S_{Disk} \| S_{MAC})$.

Оскільки користувач має можливість використовувати різні комп’ютери для входу в ігровий акаунт, то потрібно сформувати набір секретних ключів для відповідного набору комп’ютерів (апаратних засобів). Тому для кожного окремого комп’ютера, з урахуванням його унікальних характеристик, потрібно сформувати секретний ключ.

Таким чином, сервер має зберігати набір секретних ключів $M = \{ K_1, K_2, K_3, \ldots\}$, з якими користувач входить в ігровий акаунт.

Також необхідно зазначити, що залежно від мови програмування та операційної системи методи отримання серійних номерів комплектуючих ПК можуть видавати різні значення [23]. Таким чином, методи отримання

інформації про апаратну складову можуть працювати на різних рівнях доступу до системних ресурсів. Наприклад:

– На рівні ОС: високорівневі мови, такі як Java, зазвичай покладаються на операційну систему для отримання апаратної інформації. Вони викликають системні функції, доступні в ОС, але ці функції можуть не завжди надавати точний доступ до апаратних серійних номерів.

– На рівні драйверів: мови низького рівня, такі як C++, можуть безпосередньо працювати з драйверами обладнання і навіть зчитувати дані з BIOS, що забезпечує більш точні та надійні результати.

Вплив налаштувань BIOS/UEFI

Деякі серійні номери апаратного забезпечення, як-от серійні номери процесора або жорсткого диска, зберігаються в BIOS/UEFI і можуть бути нечитабельними через певні налаштування безпеки:

– BIOS може блокувати зчитування серійного номера для певних API.

– Різні методи можуть мати різні рівні доступу до BIOS, що впливає на можливість зчитати правильний серійний номер.

Також необхідно враховувати обмеження віртуальних середовищ, при використанні користувачем віртуальної машини. У віртуальних середовищах обладнання, яке визначається ОС, може бути віртуалізоване або емульоване. В таких випадках:

– Серійні номери обладнання не є справжніми, і вони можуть генеруватися самим гіпервізором (програмою для створення і керування віртуальними машинами).

– Серійний номер може змінюватися при кожному перезавантаженні або налаштуванні віртуальної машини, тому різні запити можуть повертати різні значення, особливо при використанні різних мов або бібліотек.

2.4 Розробка методу формування HMAC ігрових даних

Дані, які клієнт буде відправляти на сервер для подальшої обробки та зберігання, повинні надходити в цілісності. Для забезпечення цілісності до стандартного пакету необхідно додати додаткову інформацію у вигляді контрольної суми. Контрольна сума має містити в собі унікальні дані користувача, які також відомі серверу, ігрову інформацію, що необхідно передати, та мітка часу для забезпечення унікальності при побудові контрольної суми.

Ігрові дані формуються зі сторони клієнта, але сервер зазвичай має власну базу даних чи механізм для повторної перевірки істинності даних.

HMAC (Hash-based Message Authentication Code) — це метод створення коду автентифікації повідомлення, який використовує геш-функцію (наприклад, SHA-256) та секретний ключ. HMAC дозволяє перевірити справжність та цілісність даних, що передаються між клієнтом та сервером [24].

ipad та opad в HMAC

У HMAC використовуються два заповнювачі, звані ipad та opad, які являють собою 64-байтові константи. Ці заповнювачі необхідні для забезпечення додаткової безпеки, запобігаючи потенційним атакам, пов'язаним з прямим використанням ключа.

ipad (внутрішній паддинг) – це значення складається з 64 байтів, кожен із яких дорівнює 0x36 (у двійковому вигляді це 00110110).

opad (зовнішній паддинг) – це значення також складається з 64 байтів, але кожен байт дорівнює 0x5C (в двійковому вигляді це 01011100).

Мета використання цих значень — запобігти розкриттю або використанню ключа у прямому вигляді, оскільки це могло б зробити алгоритм більш уразливим. Натомість HMAC використовує "доповнену" версію ключа, яка змішується з ipad та opad. Ця методика також допомагає захиститись від відомих атак на геш-функції.

Алгоритм HMAC з використанням ipad та opad зображено на рисунку 2.4.

Основні кроки алгоритму HMAC з використанням ipad та opad:

1. Підготовка ключа:

– Якщо ключ К коротше 64 байтів, його доповнюють нулями до довжини 64 байта.

– Якщо ключ довший за 64 байти, то його розбивають на 2 частини, перша частина 64 байти, друга частина – решта байт. Далі кожна з частин гешується К* = H(K) і отримується 64-байтове значення.

2. Виділення ipad та opad:

– ipad та opad - 64-байтові значення, де кожен байт дорівнює 0x36 і 0x5C відповідно.

ipad = 00110110 || 00110110 ||… || 00110110.

opad = 01011100 || 01011100 || … || 01011100.

3. Внутрішнє повідомлення:

– Обчислення «внутрішнього ключа» $К_{ipad}$ = K*⊕ipad.

– Поєднання з повідомленням M* = $К_{ipad}$ || M.

– Обчислюється геш–значення h = H($К_{ipad}$ || M).

4. Зовнішнє повідомлення:

– Обчислення «зовнішнього ключа» $К_{opad}$ = K*⊕opad.

$К_{opad}$ = $K \oplus opad$.

– Поєднання з результатом попереднього гешування

M** = $К_{opad}$ || h.

– Обчислюється геш-значення h* = H(M**).

5. Результат: HMAC (K, M) = h*.

Завдяки використанню HMAC забезпечується захист від зміни даних. У випадку якщо користувач самостійно спробує сформувати пакет та підробити HMAC, це займе більше часу і вбудована античіт-система автоматично розпочне перевірку користувача на можливу фальсифікацію інформації.

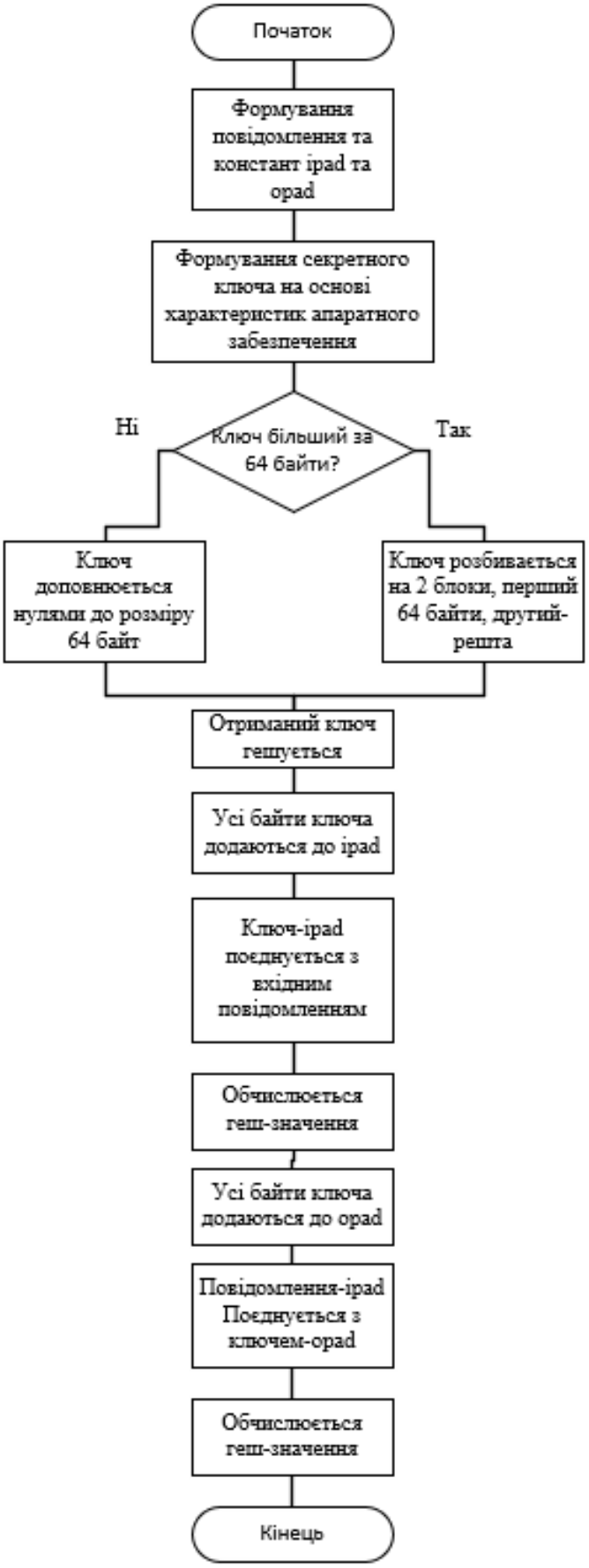


Рисунок 2.4 – Алгоритм формування HMAC

2.5 Ізоляція програми з боку користувача

Однією з найбільших вразливостей будь якої програми – це код, що знаходиться на комп’ютері користувача. Більшість ігор розташовую велику кількість коду та ресурсів на комп’ютері користувача для полегшення роботи серверу чи можливості грати без доступу до мережі інтернет. Гравець може втрутитись у файли ігри самостійно при наявності необхідних знань чи придбати софт, який за замовчуванням не передбачається в файлах гри.

Щоб захистити дані, використовуються методи розбиття програми на різні частини для ускладнення вивчення, шифрування файлів, та винесення максимально можливої частини коду на сервер. Винести весь код на ігровий сервер неможливо, але додатковий захист можливо отримати шляхом використання хмарних сервісів.

Хмарні ігрові платформи дозволяють гравцеві насолоджуватися грою без потреби встановлення локальних копій гри на пристрої. Гравець підключається до сервера хмарного сервісу, де ігровий процес запускається та обробляється на віддалених ресурсах. Це дозволяє зменшити вимоги до обчислювальної потужності клієнтських пристроїв. Користувач, взаємодіючи з грою через клавіатуру, мишу або геймпад, надсилає команди на сервер. Ці команди в режимі реального часу обробляються сервером і відображаються на екрані у вигляді оновленого відеопотоку. Сервери хмарного сервісу використовують асинхронну обробку команд для зменшення затримок, дозволяючи гравцеві миттєво бачити результати своїх дій на екрані. Механізм роботи ізоляції з використанням хмарного сервісу зображено на рис. 2.6.

В якості переваг використання хмарних сервісів для ізоляції коду модна виділити:

1. Дозволяє зберігати більшу частину важливих обчислень та даних на сервері хмарного сервісу.

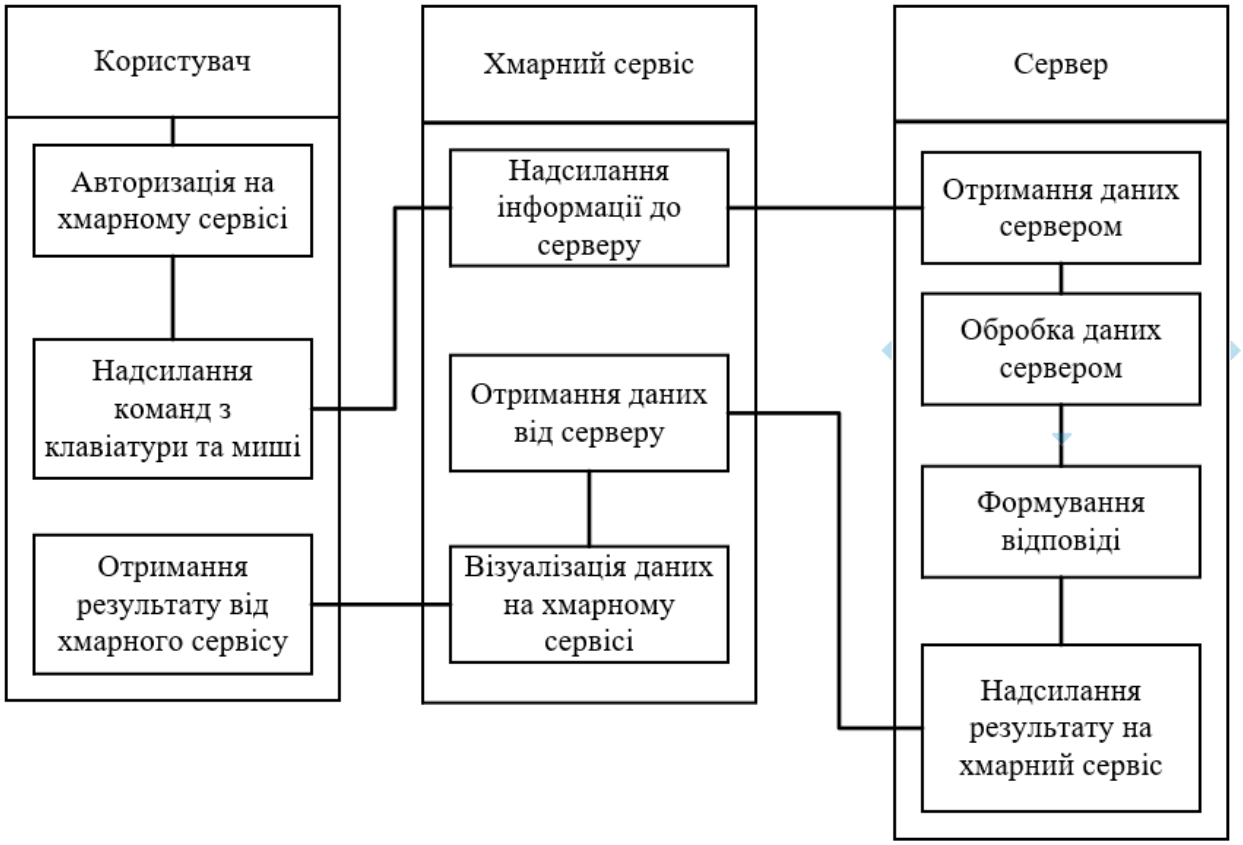


Рисунок 2.6 – Механізм роботи ізоляції

2. Захищає від шахрайства, оскільки користувач не має доступу до ключових даних, а вся важлива інформація обробляється та зберігається на сервері, доступ до якого суворо контролюється.

3. Усі оновлення відбуваються на стороні сервера, що зменшує навантаження на користувацький пристрій.

4. Ізоляція коду дозволяє значно знизити вимоги до обладнання гравця.

5. Полегшує процеси підтримки та оновлення.

6. Розробники можуть здійснювати оновлення в реальному часі без втручання користувача

7. Зниження необхідності в технічній підтримці з боку користувача знижує витрати для компаній, а також підвищує задоволеність гравців, оскільки вони отримують гру в актуальному стані без затримок.

8. Дозволяє значно підвищити доступність ігор для користувачів з різними типами пристроїв та технічними можливостями.

Недоліком використання даної системи являється залежність від інтернет підключення, від якості та швидкості якого буде залежати весь ігровий процес, а також фінансові витрати на оренду додаткового, потужного серверного кластеру для розміщення хмарного сервісу.

2.6 Висновки до розділу 2

Під час виконання дослідження було розглянуто різні методи запобігання кіберзлочинності в ігровій індустрії. Найбільш актуальними серед них є генерація секретного ключа на основі апаратних даних, використання контрольних сум та HMAC для перевірки цілісності даних, а також ізоляція клієнтського коду за допомогою хмарних сервісів. Кожен із цих підходів робить значний внесок у захист ігрових продуктів від шахрайства та інших кіберзагроз.

Генерація секретного ключа на основі апаратних даних забезпечує високий рівень захисту завдяки залежності секретного ключа від унікальних характеристик апаратного забезпечення конкретного пристрою. Основні етапи цього процесу передбачають ідентифікацію унікальних апаратних характеристик, таких як серійний номер процесора, жорсткого диска чи MAC-адреса, та генерацію ключа на основі цих даних. Це дозволяє створити систему, яка захищає ключ від несанкціонованого копіювання та ускладнює можливість повторного входу з інших пристроїв під час спроб порушення безпеки. Додаткова перевірка відповідності апаратного середовища під час використання ключа унеможливлює шахрайські дії, оскільки при виявленні невідповідності сервер може автоматично заблокувати доступ до гри.

Контрольна сума та HMAC є важливими компонентами для забезпечення цілісності та достовірності даних, що передаються між клієнтом і сервером. Використання HMAC забезпечує перевірку, що пакет даних, отриманий сервером, дійсно був створений клієнтом і не був змінений під час передачі.

Завдяки заповнювачам ipad та opad, цей метод забезпечує додатковий захист від потенційних атак на геш-функції. Забезпечення цілісності переданих даних дозволяє HMAC оперативно реагувати на спроби шахрайства та маніпуляції з боку користувачів, що має важливе значення для античіт-систем.

Обмеження доступу до клієнтської частини ігрового коду за допомогою хмарних сервісів.

Обмеження доступу до клієнтської частини ігрового коду є ефективним способом зниження ризику доступу користувача до внутрішніх компонентів гри, які можуть бути потенційно вразливими. Зокрема, хмарні ігрові платформи надають можливість зберігати й обробляти більшу частину важливих даних на сервері хмарного сервісу, що знижує вимоги до клієнтського обладнання та мінімізує ризик втручання користувача у файли гри. Використання хмарних сервісів дозволяє розробникам підтримувати та оновлювати гру без необхідності прямого доступу до пристроїв користувачів, що забезпечує контроль за безпекою та конфіденційністю.

Усі розглянуті методи значно підвищують рівень захищеності ігрових систем, запобігаючи несанкціонованому доступу до облікових записів користувачів та захищаючи конфіденційні дані. Секретний ключ, прив’язаний до апаратних даних, контрольні суми для перевірки цілісності, HMAC для забезпечення автентичності та хмарні платформи для ізоляції коду — усі ці методи дозволяють мінімізувати ризики та запобігти повторним атакам. Вони є важливими складовими для сучасних розробників, оскільки забезпечують баланс між високим рівнем безпеки і зручністю для кінцевих користувачів.

Значення кібербезпеки в ігровій індустрії зростає з кожним роком, оскільки цифрові платформи стають усе більш популярними та привабливими для кіберзлочинців. Тому впровадження й розвиток надійних методів захисту від кіберзагроз є ключовим фактором для успішного функціонування ігрових проєктів і збереження довіри користувачів.

3. РОЗРОБКА ТА ТЕСТУВАННЯ ПРОГРАМНОГО ЗАСОБУ ДЛЯ ПРОТИДІЇ КІБЕРЗЛОЧИННОСТІ

3.1 Вибір та обґрунтування середовища розробки та мови програмування

Розробка сучасного програмного забезпечення вимагає не лише якісної концептуальної розробки, а й відповідального підходу до вибору інструментів розробки, таких як мова програмування та інтегроване середовище розробки (IDE). У даній магістерській роботі для реалізації програмного продукту обрано мову програмування C# та середовище розробки Visual Studio 2022. Обидва інструменти є провідними у своїх категоріях і широко використовуються у галузі програмної інженерії.

Вибір цих технологій ґрунтувався на їхній відповідності специфічним вимогам проєкту, а також на детальному порівняльному аналізі з іншими популярними рішеннями. Далі у цьому розділі буде представлено аналіз характеристик мови та середовища розробки, їх переваги та недоліки, а також аргументовано їхній вибір у контексті завдань дипломної роботи.

Для досягнення поставлених цілей необхідно було обрати мову програмування та середовище, які могли б ефективно забезпечити виконання таких вимог:

– Продуктивність програмного забезпечення. Програма повинна забезпечувати високу швидкість виконання коду, оскільки завдання, які вона виконує, вимагають значних обчислювальних ресурсів.

– Розвинена екосистема. Наявність готових бібліотек, модулів та інструментів є ключовим фактором для прискорення розробки та спрощення реалізації складних завдань.

– Підтримка сучасних технологій. Мова програмування та IDE мають бути актуальними та підтримувати новітні стандарти програмування, що гарантує їх довготривале використання.

– Зручність розробки. Простий та інтуїтивний синтаксис мови, а також функціональність середовища розробки впливають на ефективність роботи програміста та якість кінцевого продукту.

На основі цих критеріїв було проаналізовано кілька популярних мов програмування (C#, Python, Java, JavaScript) та інтегрованих середовищ розробки (Visual Studio, JetBrains Rider, Eclipse, Visual Studio Code).

Мова програмування C#

C# – це мова програмування, яка посідає визначне місце у світі розробки програмного забезпечення. Вона привертає увагу розробників з усього світу завдяки своїй потужності, універсальності та широкому спектру застосування. У цій статті ми розглянемо, що таке середовище програмування сі шарп, та його ключові особливості й переваги в розробці.

Історія C# почалася наприкінці 1990-х років, коли Microsoft відчула необхідність у новій мові програмування, здатній ефективно працювати в середовищі Windows і сумісній із платформою .NET. У результаті зусиль команди розробників під керівництвом Андерса Гейлсберга було створено C#.

Мову було випущено 2000 року і відтоді вона активно розвивається. C# став одним із ключових елементів .NET, платформи розробки додатків, наданої Microsoft. З кожною новою версією C# набував нових можливостей і вдосконалень, роблячи її більш потужним і гнучким інструментом для розробників.

Сьогодні C# широко використовується в безлічі галузей, від створення настільних додатків і веб-серверів до розробки мобільних додатків та ігор. Його популярність і затребуваність на ринку роблять його однією з найперспективніших мов програмування для вивчення і використання [25].

C# — об’єктно-орієнтована мова програмування. Тобто C# — це оновлення, покращення мови C. [26].

C# є однією з найбільш популярних мов програмування, створеною корпорацією Microsoft. Вона використовується у багатьох галузях, включаючи розробку веб-додатків, настільних програм, мобільних застосунків, ігор та хмарних рішень. Основною перевагою C# є її тісна інтеграція з платформою .NET, що забезпечує продуктивність, надійність і масштабованість.

Продуктивність мови C# значною мірою забезпечується її компіляцією у байт-код, який виконується на віртуальній машині .NET CLR. Це дозволяє програмам працювати швидше порівняно з інтерпретованими мовами, такими як Python.

Масштабованість C# також є важливою перевагою. Завдяки своїм функціональним можливостям, таким як підтримка багатопотоковості, LINQ, асинхронного програмування та строгого типізованого синтаксису, C# підходить як для малих проєктів, так і для розробки складних корпоративних систем.

Кросплатформність стала значним досягненням C# з появою .NET Core, що дозволяє виконувати програми на Windows, Linux та macOS. Це робить мову універсальною та придатною для широкого спектру завдань.

У порівнянні з іншими мовами C# демонструє значні переваги:

– У порівнянні з Python, C# має вищу продуктивність і краще підходить для розробки додатків із високими вимогами до швидкості виконання.

– У порівнянні з Java, C# має більш сучасний синтаксис і зручнішу інтеграцію з інструментами Microsoft.

– На відміну від JavaScript, C# орієнтований не тільки на веб-розробку, але й на ширший спектр завдань, включаючи настільні та мобільні додатки.

Середовище розробки Visual Studio 2022

Термін IDE розшифровується як Integrated Development Environment або «інтегроване середовище розробки» (російською мовою іноді використовується абревіатура ІСР). Це набір інструментів, які дозволяють створювати програми.

Іншими словами, якщо говорити дуже просто, то IDE – це програма, де створюються інші програми.

Спочатку, на зорі розвитку комп'ютерної техніки, програмісти записували код на папері, після чого його вводили в ЕОМ за допомогою перфострічок або перфокарт. Одна з найменших помилок могла призвести до непрацездатності програми. Згодом, коли обчислювальна техніка досягла певного рівня розвитку, з'явилася можливість редагування коду прямо на екрані терміналу. А трохи пізніше з'явилися і засоби, що дозволяють набирати текст програми на екрані, уникаючи «паперової тяганини».

Visual Studio 2022 є одним із найпотужніших інтегрованих середовищ розробки (IDE), розроблених корпорацією Microsoft. Це середовище надає широкий спектр інструментів для зручної розробки, налагодження, тестування та розгортання програмного забезпечення.

Visual Studio 2022 підтримує всі сучасні стандарти мови C#, а також надає інструменти для роботи з різними технологіями, такими як .NET Core, WPF, ASP.NET, Xamarin та іншими [27].

Основні переваги Visual Studio 2022:

1. Інтуїтивний інтерфейс. Середовище має зрозумілий дизайн, що робить його зручним навіть для початківців.

2. Розширені інструменти налагодження. Visual Studio дозволяє швидко знаходити та виправляти помилки завдяки розширеному дебагеру, інтеграції з Live Unit Testing та інструментам профілювання продуктивності.

3. Підтримка командної роботи. Visual Studio інтегрується з Git, Azure DevOps та іншими інструментами для роботи над великими проєктами в команді.

4. Розширюваність. Середовище підтримує встановлення плагінів, що дозволяє адаптувати його функціонал під конкретні потреби розробника.

Visual Studio 2022 також має переваги перед іншими IDE. Наприклад, у порівнянні з JetBrains Rider, Visual Studio є безкоштовним для студентів і має повну підтримку всіх функцій платформи .NET. У порівнянні з Eclipse, Visual

Studio пропонує більш зручний інтерфейс і ширший набір інструментів для роботи з C#.

На основі проведеного аналізу можна зробити висновок, що вибір мови програмування C# та середовища розробки Visual Studio 2022 є обґрунтованим і оптимальним для реалізації поставлених завдань. Ці інструменти забезпечують високу продуктивність, зручність у роботі, а також широкий функціонал для реалізації сучасних технологій.

Таке поєднання дозволяє розробити якісний і масштабований програмний продукт, який відповідає вимогам сучасної інженерії програмного забезпечення.

3.2 Розробка програмного засобу

Програмний засіб, що реалізує побудову пакету даних для передачі на сервер складатиметься з трьох основних класів, кожен з яких реалізує власний функціонал. Розділення функціоналу зумовлено обмеженням доступу між компонентами коду з метою запобігання не бажаного втручання із зовні.

Перший клас HardwareBinding являється публічним і містить в собі статичні поля та не має конструктора для створення власного об’єкту. Основним завдання даного класу являється збір даних про апаратне забезпечення персонального комп’ютера користувача та формування секретного ключа на основі отриманої інформації.

При запуску програми користувачу необхідно пройти реєстрацію вказавши свій унікальний ігровий ідентифікатор «нікнейм», який відправляється на сервер. Потім сервер генерує 2 ключі, приватний для розшифрування та публічний для зашифрування, публічний ключ відправляється у користувачу і буде використовуватись для зашифрування секретного ключа створеного на основі характеристик апаратного забезпечення.

Далі йде генерація секретного ключа на основі апаратних даних персонального комп’ютера користувача. В якості формування ключа секретного ключа використовуються наступні параметри:

Серійний номер жорсткого диску, для отримання якого використовується приватний статичний метод GetDiskId(), що повертає в результати стрічку з серійним номером, та не приймає вхідних параметрів.

```
private static string GetDiskId()
{
    using (var searcher = new ManagementObjectSearcher("SELECT SerialNumber FROM
Win32_DiskDrive"))
    {
        foreach (var item in searcher.Get())
        {
            return item["SerialNumber"]?.ToString() ?? string.Empty;
        }
    }
    return string.Empty; }
```

— Серійний номер процесора, для отримання якого використовується приватний статичний метод GetCpuId(), що повертає в результати стрічку з серійним номером, та не приймає вхідних параметрів.

```
  private static string GetCpuId()
  {
      using (var searcher = new ManagementObjectSearcher("SELECT ProcessorId FROM
Win32_Processor"))
      {
          foreach (var item in searcher.Get())
          {
              return item["ProcessorId"]?.ToString() ?? string.Empty;
          }
      }
      return string.Empty;
       }
```

— MAC-адрес мережевої карти, для отримання якого використовується приватний статичний метод GetMacAddress (), що повертає в результати стрічку з MAC-адресом, та не приймає вхідних параметрів.

```
    private static string GetMacAddress()
    {
        using (var searcher = new ManagementObjectSearcher("SELECT MACAddress FROM
Win32_NetworkAdapter WHERE MACAddress IS NOT NULL"))
        {
            foreach (var item in searcher.Get())
            {
                return item["MACAddress"]?.ToString() ?? string.Empty;
            }
        }
        return string.Empty;
    }
```

Викликається метод GenerateHardwareHash() з статичного класу HardwareBinding, який являється публічним та не приймає параметри на вхід.

```
public static string GenerateHardwareHash()
{
    string hardwareId = GetHardwareIdentifier();
    using (SHA256 sha256 = SHA256.Create())
    {
        byte[] hashBytes = sha256.ComputeHash(Encoding.UTF8.GetBytes(hardwareId));
        StringBuilder hash = new StringBuilder();
        foreach (byte b in hashBytes)
        {
            hash.Append(b.ToString("x2"));
        }
        return hash.ToString();
    }}
```

Після збору усіх необхідних даних про апаратне забезпечення необхідно поєднати їх у єдину структуру для подальшого зашифрування. Збір інформації та її поєднання виконує приватний статичний метод GetHardwareIdentifier(), який не приймає параметри та повертає стрічку. Також даний метод містить методи для вивід інформації у консоль. Таке рішення зумовлено необхідністю у перевірці результатів та полегшенню тестування.

```
private static string GetHardwareIdentifier()
{
    string cpuId = GetCpuId();
    string diskId = GetDiskId();
    string macAddress = GetMacAddress();

    Console.WriteLine("cpu ID - " + cpuId);
    Console.WriteLine("disk ID - " + diskId);
    Console.WriteLine("mac address - " + macAddress);

    Console.WriteLine("combined data - " + macAddress);
    return cpuId + diskId + macAddress;
}
```

Для сформованої структури генерується геш-значення. Для генерація геш-значення використовується алгоритм SHA-256, який реалізовано в приватному статичнному методі GenerateHardwareHash(), який не приймає параметри та в результаті повертає стрічку. Після отримання гешу, він перетворюється у шістнадцятковий формат, для зручної передачі сформованого ключа.

```
public static string GenerateHardwareHash()
{
    hardwareId = GetHardwareIdentifier();
```

```
    using (SHA256 sha256 = SHA256.Create())
    {
        byte[] hashBytes = sha256.ComputeHash(Encoding.UTF8.GetBytes(hardwareId));
        StringBuilder hash = new StringBuilder();
        foreach (byte b in hashBytes)
        {
            hash.Append(b.ToString("x2"));
        }
        return hash.ToString();
    }
}
```

Отриманий секретний ключ зашифровується за допомогою відкритого ключа створеного сервером і передається на сервер. Отриманий зашифрований секретний ключ розшифровується приватним ключем сервера для подальшого зберігання та використання у методах блокування та розшифровки отриманих пакетів даних.

Другий клас допоміжний і призначений для генерації тестових даних та використання методів для створення та перевірки контрольної суми, зашифрування та розшифрування створеного пакету даних. Клас AntiCheatSystemTest являється публічним та статичним і містить у собі лише один публічний, статичний метод RunTests(string key), що приймає на вхід стрічку з сформованим секретним ключем та не повертає ніяких значень після виконання.

```
public static void RunTests(string key)
 {
     EncryptedDataPacketCreator antiCheat = new EncryptedDataPacketCreator();
     antiCheat.SetSecretKey(key);

     Console.WriteLine("Enter user name");
     string userId = Console.ReadLine();

     Console.WriteLine("Enter item price");
     string itemCost = Console.ReadLine();
     DateTime timestamp = DateTime.UtcNow;

     string integrityHash = antiCheat.GenerateIntegrityHash(userId, itemCost,
timestamp);
     Console.WriteLine("Checksum - " + integrityHash);

     bool isDataValid = antiCheat.ValidateData(userId, itemCost, timestamp,
integrityHash);
     Console.WriteLine("Validation of data integrity: " + (isDataValid ? "Passed" :
"Failed"));

     string encryptedData = antiCheat.EncryptData(itemCost.ToString(), key);
     Console.WriteLine("Encrypted Data - " + encryptedData);
```

```
    string decryptedData = antiCheat.DecryptData(encryptedData, key);
    Console.WriteLine("Decrypted Data - " + decryptedData);


    Console.WriteLine("Decryption result: " + (decryptedData == itemCost.ToString() ?
"Passed" : "Failed")); }
```

Метод RunTests(string key) створює екземпляр третього класу – EncryptedDataPacketCreator, який відповідає за реалізацію методів побудови контрольної суми, її перевірки, зашифрування, розшифрування та перевірку результатів після розшифрування отриманого пакету даних. Клас EncryptedDataPacketCreator являється публічним, не реалізує конструктори, але для використання його методів необхідно створити екземпляр, оскільки методи даного класу використовують секретний ключ, який може змінитись при використанні спуферу і запобігти блокуванню апаратної складової.

Після генерації ігрових даних для формування пакету викликається публічний метод GenerateIntegrityHash(string userId, int itemCost, DateTime timestamp), який приймає на вхід стрічку з унікальним ідентифікатором користувача, даними для передачі, наприклад вартість ігрового предмета, та мітку часу та в результаті повертає стрічку з сформованою контрольною сумою пакету.

```
public string GenerateIntegrityHash(string userId, int itemCost,
DateTime timestamp)
{
    string data = $"{userId}|{itemCost}|{timestamp.ToUniversalTime():o}";
    return ComputeHmac(data);
}
```

В середині даного методу дані поєднуються в одну структуру та передаються у приватний метод ComputeHmac(string data), який приймає стрічку за даними та повертає стрічку з сформованою контрольною сумою. Метод ComputeHmac виділяє окремий потік для формування HMAC. Для створення об’єкту HMAC необхідно перетворити створений раніше секретний ключ у масив байт за допомогою виразу Encoding.UTF8.GetBytes(_secretKey). Далі на основі отриманого об’єкту формується геш-значення та перетворюється з 8-ми

розрядного масива байт у стрічку. Перетворення у стрічку виконується для полегшення огляду та тестування роботи системи.

```
private string ComputeHmac(string data)
{
    using (var hmac = new HMACSHA256(Encoding.UTF8.GetBytes(_secretKey)))
    {
        byte[] hashBytes = hmac.ComputeHash(Encoding.UTF8.GetBytes(data));
        return Convert.ToBase64String(hashBytes);
    }
}
```

По завершенню процесу, результат виводиться на екран консолі з метою подальшого порівнянн. Далі викликається публічний метод ValidateData(string userId, int itemCost, DateTime timestamp, string receivedHash), який приймає на вхід стрічку з унікальним ідентифікатором користувача, даними для передачі, наприклад вартість ігрового предмета, мітку часу та контрольну суму яку необхідно перевірити. В результаті, метод повертає бульове значеня.

```
public bool ValidateData(string userId, int itemCost, DateTime timestamp, string
receivedHash)
{
    string calculatedHash = GenerateIntegrityHash(userId, itemCost, timestamp);
    return receivedHash == calculatedHash;
}
```

Результат виконання даного методу виводиться на екран консолі.

Наступний крок це зашифрування пакету даних за допомогою AES. AES-256 (Advanced Encryption Standard 256-біт) — це симетричний алгоритм шифрування, який використовується для захисту конфіденційних даних. Він використовує 256-бітний ключ для шифрування та дешифрування, що робить його надзвичайно безпечним і широко використовуваним у різноманітних застосуваннях, включаючи захист даних у стані спокою, захист даних під час передачі та забезпечення захисту конфіденційних комунікацій [28].

Для реалізації алгоритму шифурвання AES використовується публічний метод EncryptData(string data, string encryptionKey), який приймає на вхід дані для зашифрування та секретний ключ, а в результаті повертає стрічку з зашифрованим повідомлення.

```
public string EncryptData(string data, string encryptionKey)
{
    using (Aes aes = Aes.Create())
```

```
    {
        using (var sha256 = SHA256.Create())
        {
            aes.Key = sha256.ComputeHash(Encoding.UTF8.GetBytes(encryptionKey));
        }

        aes.GenerateIV();
        byte[] iv = aes.IV;

        using (var encryptor = aes.CreateEncryptor())
        {
            byte[] dataBytes = Encoding.UTF8.GetBytes(data);
            byte[]    encrypted    =    encryptor.TransformFinalBlock(dataBytes,    0,
dataBytes.Length);

            byte[] result = new byte[iv.Length + encrypted.Length];
            Buffer.BlockCopy(iv, 0, result, 0, iv.Length);
            Buffer.BlockCopy(encrypted, 0, result, iv.Length, encrypted.Length);

            return Convert.ToBase64String(result);
        }
    }
}
```

Після зашифрування даних, реалізується метод розшифрування пакетів, що надходять від сервера. Разшифрування виконує публічний метод DecryptData(string encryptedData, string encryptionKey), який приймає на вхід стрічку з зашифрованими даними та секретний ключ, а в результаті повертає стрічку з розшифрованими даними.

```
public string DecryptData(string encryptedData, string encryptionKey)
 {
     using (Aes aes = Aes.Create())
     {
         using (var sha256 = SHA256.Create())
         {
             aes.Key = sha256.ComputeHash(Encoding.UTF8.GetBytes(encryptionKey));
         }

         byte[] fullData = Convert.FromBase64String(encryptedData);

         byte[] iv = new byte[aes.BlockSize / 8];
         Buffer.BlockCopy(fullData, 0, iv, 0, iv.Length);
         aes.IV = iv;

         int encryptedDataLength = fullData.Length - iv.Length;
         byte[] encryptedBytes = new byte[encryptedDataLength];
         Buffer.BlockCopy(fullData, iv.Length, encryptedBytes, 0, encryptedDataLength);

         using (var decryptor = aes.CreateDecryptor())
         {
             byte[]   decrypted   =   decryptor.TransformFinalBlock(encryptedBytes,   0,
encryptedBytes.Length);
             return Encoding.UTF8.GetString(decrypted);
         }
     }
```

```
}
```

Останнім кроком розшифровані дані перевіряються з істинними, що зберігаються в базі даних серверу та являються статичними. У випадку невідповідності у консоль буде виведено «Failed», а при збіганні даних результат буде «Passed».

Також було реалізовано методи для збереження та зчитування інформації з файлів шляхом використання бінарної серіалізації та десеріалізації. Серіалізація та десеріалізації необхідні для проведення тестування методу автентифікації користувача на основі апаратних характеристик. Результат виконання методу методу GenerateHardwareHash() серіалізується в файл, що надає змогу перевірити дані користувача при запуску програми на іншому пристрої. Така система дає можливість пришвидшити тестування, оскільки при виникненні помилки в її можливо одразу усунути і не витрачати час на повторне встановлення на сервер.

Серіалізацію та десеріалізацію реалізує публічний статичний клас BinarySerialization, який реалізує 2 методи. Метод Serialize(object obj, string path) є публіним і статичним, після виконання не повертає ніяких даних та приймає на вхід дані з універсальним типом object та стрічку з шляхом та до місця збереження файлу та назвою.

```
public static void Serialize(object obj, string path)
 {
     try
     {
         BinaryFormatter formatter = new BinaryFormatter();

         using (FileStream fs = new FileStream(path, FileMode.OpenOrCreate))
         {
             formatter.Serialize(fs, obj);
         }
     }
     catch (Exception e)
     {
         Console.WriteLine(e.Message);
     }
 }
```

Метод Deserialize(string path) є публічним, статичним, після виконання повертає вміст файлу у вигляді списку стрічок та приймає на вхід стрічку із шляхом до файлу та його назву.

```
public static List<string> Deserialize(string path)
 {
     try
     {
         BinaryFormatter formatter = new BinaryFormatter();

         using (FileStream fs = new FileStream(path, FileMode.OpenOrCreate))
         {
             try
             {
                 return formatter.Deserialize(fs) as List<string>;
             }
             catch
             {
                 return new List<string>();
             }
         }
     }
     catch
     {
         return null;
     }
 }
```

Точною входу для запуску усіх подальших процесів являється базовий клас Program. Даний клас створюється автоматично при створені консольного проекту та містить лише один метод static void Main(string[] args), який автоматично запускається при запуску програми.

```
static void Main(string[] args)
{
    List<string> SaveData = new List<string>();

    string hardwareHash = HardwareBinding.GenerateHardwareHash();
    SaveData = BinarySerialization.Deserialize("Save");
    if (!SaveData.Contains(hardwareHash)) SaveData.Add(hardwareHash);
    BinarySerialization.Serialize(SaveData, "Save");
    Console.WriteLine("Generated Hardware Hash: " + hardwareHash + "\n\n");

    bool isVerified = HardwareBinding.VerifyHardwareHash(hardwareHash);
    Console.WriteLine("Hardware Hash Verified: " + isVerified + "\n\n");
    SaveData = BinarySerialization.Deserialize("Save");
    if(!SaveData.Contains(hardwareHash)) SaveData.Add(hardwareHash);

    if (SaveData != null)
    {
        foreach (string s in SaveData)
        {
            Console.WriteLine(s);
        }
```

```
    }

    BinarySerialization.Serialize(SaveData, "Save");
    Console.WriteLine("\n\n");

    AntiCheatSystemTest.RunTests(hardwareHash);

    Console.ReadLine();
}
```

3.3 Тестування програмного засобу

В результаті було отримано консольний програмний засіб, який збирає інформацію про характеристики апаратного забезпечення персонального комп'ютера користувача. Отримані дані поєднуються в одну структуру, гешуються та передаються для збереження на сервері. Результати отриманих характеристик апаратного забезпечення та сформованого геш-значеня зображено на рис. 3.1.

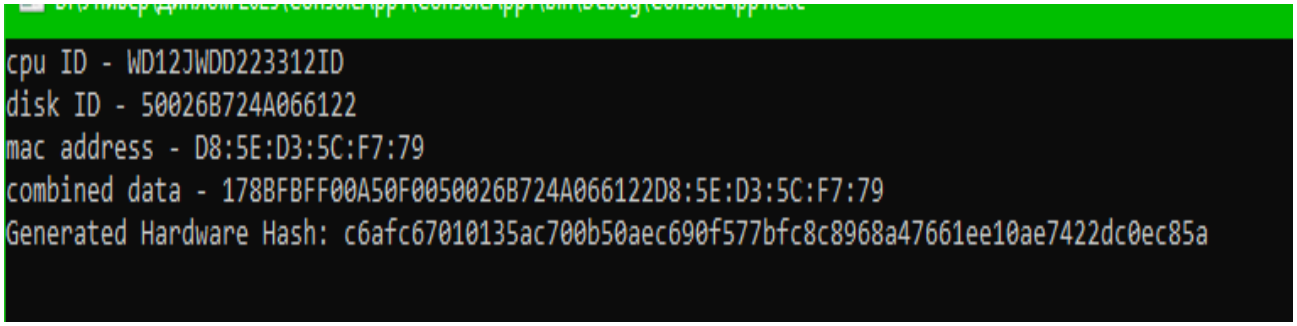


Рисунок 3.1 – Результат отриманих характеристик апаратного забезпечення та сформованого геш-значеня

При повторній спробі авторизації користувача, буде відправлено запит, чи зареєстрований користувач. Якщо користувач зареєстрований, з його акаунту буде взята інформація про дані комп’ютерів, з яких користувач заходив раніше виконував вхід. Якщо поточні дані уже знаходяться в базі на акаунті користувача, доступ до гри буде надано. Результат надання доступу до гри при відповідності апаратних характеристик комп’ютера користувача зображено на рис. 3.2. У випадку використанні нового пристрою, дані характеристик будуть

додані до акаунту користувача і доступ буде надано. Результат надання доступу до гри при використанні іншого комп’ютера зображено на рис. 3.3.

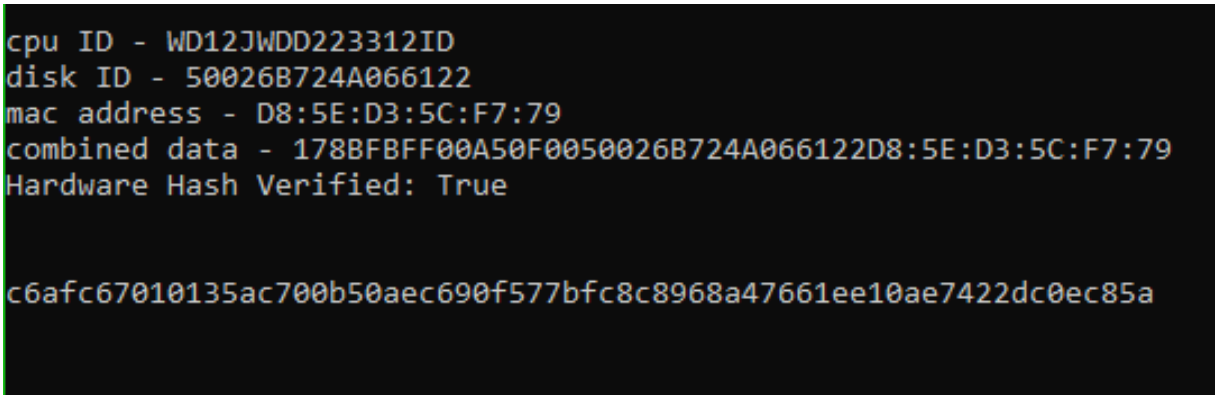


Рисунок 3.2 – Результат надання доступу до гри при відповідності апаратних характеристик комп’ютера користувача

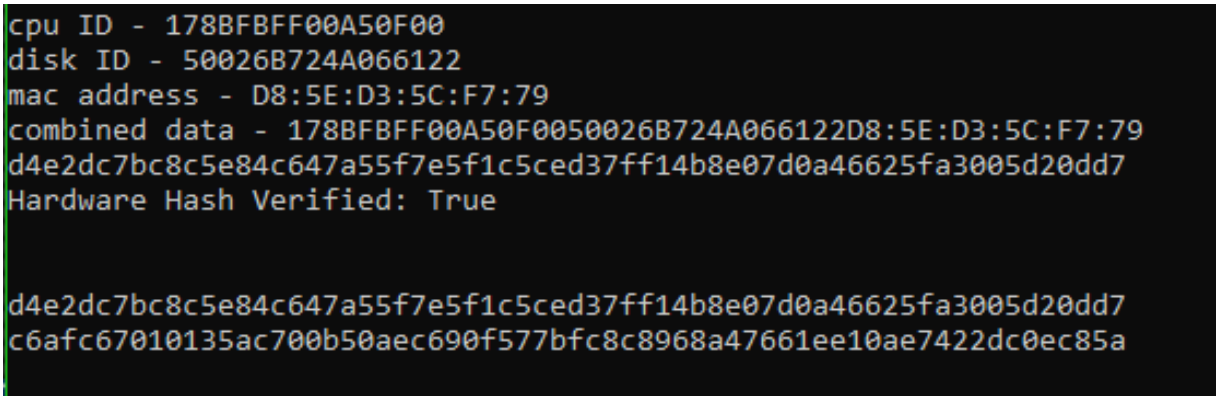


Рисунок 3.3 – Результат надання доступу до гри при використанні іншого комп’ютера

Після отримання доступу до гри котристувач матиме можливість відправляти інформацію на сервер. Для відправки даних необхідно отримати ігрове ім’я користувача та вказати ціну предмету, який користувач хоче продати для отримання ігрової валюти. Дані, що необхідно відправити на сервер, формують HMAC та зашифровуються, після чого поєднуються в один пакет для відправки. Сервер отримує пакет, розшифровує його, формує власний HMAC з використанням інформації із власної бази даних. Якщо HMAC сформований сервером на HMAC відправлений користувачем співпадають, дані проходять

валідацію і можуть використовуватись для подальшої обробки. Результат успішної передачі даних від користувача до серверу зображено на рис. 3.4.

```
Enter user name
jagernaut
Enter item price
1000
Checksum - LPYy8hqA62/ni44vWn6lDMJKxVKXMzyX55ps5XhHyc4=
Encrypted Data - W9bsklZPDcbGdVtbbzNz58mPa1ux9v2oHrvLyvanfkY=
Validation of data integrity: Passed
Decrypted Data - 1000
```

Рисунок 3.4 – Результат успішної передачі даних від користувача до сервера

Якщо користувач намагається відправити хибні дані, наприклад, продати предмет за значну вищу ціну, сервер спробує сформувати HMAC з використанням інформації із бази даних і в результаті. HMAC сформований сервером будуть відрізнятись. В такому випадку, дані не проходять валідацію і не будуть використовуватись в подальшому. Результат передачі неможливих даних зображено на рис 3.5. Сервер відправить повторний запит на отримання інформації. Якщо в подальшому дані не будуть проходити валідацію, користувача буде передано античіт системі, яка встановлений на сервері, для аналізу активності та виявлення втручання в ігровий процес.

```
Enter user name
jagernaut
Enter item price
999999999999
Checksum - X8Dt3QR95I+nr3ERtpDnKsKILSNzoDeIlvQnMD2DYxY=
Encrypted Data - xbCP37pKIayQcOpsh04Y3KIUgvSk3rDKF9cDJ/1sJwY=
Validation of data integrity: Failed
Decrypted Data - 999999999999
```

Рисунок 3.5 – Результат передачі неможливих даних

3.4 Висновки до розділу 3

У межах роботи було реалізовано програмний засіб для підвищення надійності передачі даних між клієнтом і сервером в умовах захисту від можливих маніпуляцій або підробок. Основними аспектами роботи стали вибір мови програмування, середовища розробки та створення алгоритмів перевірки цілісності й захисту даних.

Мова C# була обрана для реалізації розробки з огляду на її відповідність сучасним вимогам до безпеки, продуктивності та гнучкості. Основні переваги, що сприяли вибору:

1. Можливості роботи із сучасними алгоритмами криптографії. Завдяки вбудованим бібліотекам, таким як System.Security.Cryptography, C# дозволяє реалізувати алгоритми HMAC, AES та інші засоби захисту.

2. Зручність роботи з рядковими і бінарними даними. Ця функціональність є ключовою для обробки й шифрування даних.

3. Широке використання у сфері розробки серверних і клієнтських програм. C# забезпечує високу продуктивність при роботі з великими обсягами даних і інтегрується з іншими платформами.

Visual Studio 2022 було обрано як основне середовище розробки завдяки її широкому функціоналу:

1. Зручна інтеграція інструментів аналізу коду дозволяє підвищити якість і безпеку коду.

2. Інструменти для роботи з великими проєктами дають змогу ефективно управляти складними структурами програмного засобу.

3. Підтримка інтеграції з системами контролю версій, такими як Git, значно полегшує управління проєктом і співпрацю з іншими розробниками.

4. Автоматизація тестування та налагодження. Це дозволяє швидко перевіряти працездатність алгоритмів і шукати помилки.

Розроблений програмний засіб

Розроблений програмний засіб забезпечує комплексний захист даних, інтегруючи перевірку цілісності, шифрування та виявлення спроб підробки. Основою перевірки автентичності є алгоритм HMAC на основі SHA-256, який дозволяє створювати унікальні контрольні значення. Ці значення формуються з використанням ідентифікатора користувача, даних про ігровий процес та часової мітки, що забезпечує точну відповідність між переданими даними і їхнім хешем. Такий підхід дозволяє переконатися у незмінності інформації та її автентичності під час передачі.

Для захисту конфіденційності даних використовується алгоритм шифрування AES, який реалізований із динамічним вектором ініціалізації (IV). Завдяки цьому алгоритм ефективно шифрує будь-які текстові дані перед їх відправкою, запобігаючи доступу до них сторонніх осіб. Вектор ініціалізації гарантує унікальність кожного зашифрованого блоку, навіть якщо вихідні дані повторюються. Це значно підвищує рівень безпеки переданих повідомлень.

Програмний засіб реалізовано у вигляді класу EncryptedDataPacketCreator, який включає методи для виконання основних функцій. Метод GenerateIntegrityHash забезпечує створення контрольного хеша, що використовується для перевірки цілісності даних. Для верифікації отриманих сервером даних використовується метод ValidateData, який порівнює отриманий хеш із тим, що було розраховано сервером. Метод EncryptData здійснює шифрування чутливих даних перед їх передачею, тоді як метод DecryptData виконує зворотню операцію — розшифровує отримані на сервері дані для подальшої обробки.

В результаті отримано програмний засіб, який поєднує в собі ефективність алгоритмів криптографії та надійність перевірки даних, що дозволяє значно підвищити рівень захищеності інформації під час передачі між клієнтом і сервером.

4. ЕКОНОМІЧНА ЧАСТИНА

4.1 Аналіз комерційного потенціалу розробки методу обмеження доступу до ігрових даних

Метою проведення технологічного аудиту є оцінка комерційного потенціалу методу обмеження доступу до ігрових даних. Після проведення оцінювання можна зробити висновок щодо організації подальшого впровадження на основі сформованого рейтингу.

Для проведення аудиту залучено трьох незалежних експертів якості першого незалежного експерту представлено керівника магістерської кваліфікаційної роботи, завідувач кафедри, професор Лужецький В. А., головного інженеру у сфері розробки програмного забезпечення Денесюк В. І. та заступник головного інженеру у сфері розробки програмного забезпечення Гнатенко А. А. на підприємстві Parkaudio.

Оцінювання комерційного потенціалу розробки методу обмеження доступу до ігрових даних здійснюється за дванадцятьма критеріями згідно рекомендацій. Результат оцінювання комерційного потенціалу розробки методу обмеження доступу до ігрових даних наведено в табл. 4.1.

За даними табл. 4.1 можна зробити висновки що рівня комерційного потенціалу розробки методу обмеження доступу до ігрових даних. Висновки сформовані за допомогою рекомендацій наведених в табл. 4.2.

На основі рекомендацій наведених у табл. 4.2 модна зробити висновки, що рівень комерційного потенціалу розробки методу обмеження доступу до ігрових даних вище середнього.

Дані результати були досягнені шляхом збільшення рівня захисту передачі даних між користувачем та сервером при використання методу автентифікації даних, та використання методу автентифікації користувача на основі характеристик апаратно забезпечення, що створює додаткову перешкоду для

шахраїв у вигляді блокування не тільки ігрового акаунти, а й персонального комп’ютера в цілому.

Таблиця 4.1 – Результат оцінювання комерційного потенціалу розробки методу обмеження доступу до ігрових даних

Критерії	Експерти		
	ЛУЖЕЦЬКИЙ В.А.	ДЕНЕСЮК В.І.	ГНАТЕНКО А. А.
	Бали, виставлені експертами		
1	2	2	3
2	2	3	3
3	3	3	3
4	4	4	3
5	2	3	4
6	4	5	5
7	3	2	2
8	3	3	3
9	3	3	3
10	3	3	3
11	3	4	4
12	3	3	3
Сума балів	35	38	39
Середньоарифметична сума балів, СБ	37,3		

Таблиця 4.2 – Рівні комерційного потенціалу

Середньоарифметична сума балів СБ, розрахована на основі висновків експертів	Науково-технічний рівень та комерційний потенціал розробки
41…48	Високий
31…40	Вищий середнього
21…30	Середній
11…20	Нижче середнього
0…1	Низький

Економічні показники конкурентоспроможності характеризують сумарні витрати споживачів на задоволення їх потреб цією розробкою. Вони складаються з витрат на придбання і витрат, пов'язаних з експлуатацією виробу. Оцінювання рівня конкурентоспроможності науково-технічної розробки здійснюється в декілька етапів

Процедура визначення якісних одиничних параметричних індексів за технічними показниками здійснюється за відповідними формулами. Якщо збільшення величини параметра свідчить про підвищення якості нової розробки, одиничний параметричний індекс розраховується за формулою:

$$q_i = \frac{P_i}{P_{базі}}.$$

Якщо зменшення величини параметра свідчить про підвищення якості нової розробки, то одиничний параметричний індекс розраховується за оберненою формулою:

$$q_i = \frac{P_{базі}}{P_i},$$

де qi – одиничний параметричний індекс, розрахований за і-м параметром;

Рі – значення і-го параметра розробки;

Рбазі – аналогічний параметр базової розробки-аналога, з якою проводиться порівняння.

Затримка – $q1 = \frac{72}{80} = 0{,}9$;

безпека – $q2 = \frac{4}{3} = 1{,}25$;

ціна – $q3 = \frac{2000}{1700} = 1{,}17$;

Технічні та економічні параметри аналога та нової науково-технічної розробки наведені у табл. 4.3.

Таблиця 4.3 – Технічні та економічні параметри аналога та нової науково-технічної розробки

Параметр	Одиниця виміру	Аналог	Нова розробка	Індекс зміни значення параметра	Коефіцієнт вагомості
Затримка	Мс	80	72	0,9	1
Ціна	Грн	1700	2000	1,17	0,2
Безпека		3	4	1,25	0,7

Груповий показник конкурентоспроможності за нормативними параметрами розраховується як добуток частинних показників за кожним параметром за формулою:

$$I_{НП} = \prod_{i=1}^{n} q_i \ ,$$

де Інп – загальний показник конкурентоспроможності за нормативними параметрами; qi – одиничний (частинний) показник за і-м нормативним параметром; n – кількість нормативних параметрів, які підлягають оцінюванню.

Інп = 0,9 * 1,25 * 1,17 = 1,31

Якщо хоч один з частинних показників дорівнює 0 (тобто не відповідає встановленим нормам), то розробка неконкурентоспроможна. Значення групового параметричного індексу за технічними параметрами визначається з урахуванням вагомості (частки) кожного параметра:

$$I_{ТП} = \sum_{i=1}^{n} q_i \cdot \alpha_i \ ,$$

де Ітп – груповий параметричний індекс за технічними показниками (порівняно з аналогом); qi – одиничний параметричний показник і-го параметра; αi – вагомість і-го параметричного показника:

$$\sum_{i=1}^{n} \alpha_i = 1 ;$$

де n – кількість технічних параметрів, за якими оцінюється конкурентоспроможність.

Ітп = (0,9 * 0,5) + (1,25 * 0,7) = 1,325

На основі групових параметричних індексів за нормативними, технічними та економічними показниками розраховують інтегральний показник конкурентоспроможності за формулою

$$K_{ІНТ} = I_{НП} \cdot \frac{I_{ТП}}{I_{ЕП}}.$$

На основі інтегрального показника формується висновок про конкурентоспроможність оцінюваної розробки. У разі КІНТ < 1 аналізована розробка поступається базовому зразку за конкурентоспроможністю, за КІНТ > 1 – перевищує зразок. За умови рівної конкурентоспроможності КІНТ = 1. Однак потрібно мати на увазі, що за зростання ІТП (тобто поліпшення споживчих показників аналізованої розробки) показник КІНТ збільшується, характеризуючи зростання конкурентоспроможності. За зростання ІЕП (ціни споживання аналізованої розробки порівняно з базовим зразком) показник КІНТ зменшується, відображаючи зниження конкурентоспроможності:

Кінт = 1,31 * (1,325 / 1) = 1,73.

4.2 Розрахунок витрат на здійснення науково-дослідної роботи

Основна заробітна плата кожного із розробників Зо, якщо вони працюють в наукових установах бюджетної сфери:

$$Зо = \frac{М}{Тр} \cdot t,$$

де М – місячний посадовий оклад конкретого розробника (інженера, розробника, науковця тощо), грн.

У 2024 році величини окладів рекомендується брати в межах від 80000 до 14000 грн. за місяць; Тр – число робочих днів в місяці; Тр приблизно дорівнює від 21 до 23 днів; t – число робочих днів роботи розробника. Результат розрахунків наведено в табл. 4.4.

Таблиця 4.4 – Заробітна плата розробників

Посада	Місячний посадовий оклад, грн.	Оплата за робочий день, грн.	Число днів роботи	Витрати на заробітну плату, грн.
Керівник	22000	1000	5	5000
Інженер-програміст	15400	700	20	14000
Всього				19000

Основна заробітна плата розробників Зр, якщо вони беруть участь у виконанні даного етапу роботи і виконують роботи за робочими професіями у випадку, коли вони працюють в наукових установах бюджетної сфери, розраховується за формулою:

$$З_p = \sum_{i=1}^{n} C_i \cdot t_i ,$$

де Сі – погодинна тарифна ставка робітника відповідного розряду, за виконану відповідну роботу, грн/год; ti – час роботи робітника на виконання певної роботи, год. Погодинну тарифну ставку робітника відповідного розряду Сі можна визначити за формулою:

$$C_i = \frac{M_M \cdot K_i \cdot K_c}{T_p \cdot t_{зм}},$$

де Мм – розмір прожиткового мінімуму працездатної особи або мінімальної місячної заробітної плати 8000, грн; Кі – коефіцієнт міжкваліфікаційного співвідношення для встановлення тарифної ставки робітнику відповідного розряду; Кс – мінімальний коефіцієнт співвідношень місячних тарифних ставок робітників першого розряду з нормальними умовами праці виробничих

об’єднань і підприємств до законодавчо встановленого розміру мініма-ль-ної заробітної плати. Тр – середня кількість робочих днів в місяці, приблизно Тр = 21…23 дні; tзм – тривалість зміни, год.

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
Програмувальні	8	5	1.7	127,5	1020
Тестувальні	8	4	1.5	112,5	900
Всього					1920

Додаткова заробітна плата розраховується як 10 … 12% від суми основної заробітної плати дослідників та робітників за формулою:

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

де Ндод – норма нарахування додаткової заробітної плати.

Здод = 0,1 * (19000 + 1920) = 2092 грн.

Нарахування на заробітну плату Нзп розробників та робітників, які брали участь у виконанні даного етапу роботи, розраховуються за формулою:

де Зо — основна заробітна плата розробників, грн.; Зр — основна заробітна плата робітників, грн.; Зд — додаткова заробітна іілата всіх розробників та робітників, грн.; § — ставка единого внеску на загальнообов'язкове державне соціальне страхування, % (приймаемо для 1-го класу професійності ризику 22%).

Нзп = 0,22 * (Зр + Зо + Зд) = 0,22 * (19000 + 1920 + 2092) = 5062,64 грн.

До статті «Амортизація програмних засобів» відносять амортизаційні відрахування по кожному виду обладнання, устаткування та інших приладів і

пристроїв, а також програмного забезпечення для проведення науково-дослідної роботи, за його наявності в дослідній організації або на підприємстві.

В спрощеному вигляді амортизаційні відрахування по кожному виду обладнання, приміщень та програмному забезпеченню тощо можуть бути розраховані з використанням прямолінійного методу амортизації за формулою:

$$A_{обл} = \frac{Ц_б}{T_в} \cdot \frac{t_{вик}}{12},$$

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

tвик – термін використання обладнання, програмних засобів, приміщень під час досліджень, місяців;

Тв – строк корисного використання обладнання, програмних засобів, приміщень тощо, років. Проведені розрахунки наведено в табл. 4.6.

Таблиця 4.6 – Амортизаційні відрахування по кожному виду обладнання

Найменування обладнання	Балансова вартість, грн	Строк корисного використання, років	Термін використання обладнання, місяців	Амортизаційні відрахування, грн
ПК	50000	2	2	4166,67
Приміщення	175000	20	2	1458,33
Всього				5625

До статті «Сировина та матеріали» належать витрати на сировину, основні та допоміжні матеріали, інструменти, пристрої та інші засоби і предмети праці, які придбані у сторонніх підприємств, установ і організацій та витрачені на проведення досліджень.

Витрати на матеріали (М), у вартісному вираженні розраховуються окремо по кожному виду матеріалів за формулою:

$$M = \sum_{j=1}^{n} H_j \cdot Ц_j \cdot K_j - \sum_{j=1}^{n} B_j \cdot Ц_{вj},$$

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

n – кількість видів матеріалів;

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

Кj – коефіцієнт транспортних витрат, (Кj = 1,1 … 1,15);

Вj – маса відходів j-го найменування, кг;

Цвj – вартість відходів j-го найменування, грн/кг.

Таблиця 4.7 - Витрати на матеріали

Найменування матеріалу, марка, тип, сорт	Ціна за 1 кг, упаковку	Норма витрат, кг, уваковку	Величина відходів, кг	Ціна відходів кг/грн	Вартість витраченого матеріалу
Папір А4	230	2	0	0	506
Канцелярське обладнання	175	2	0	0	385
Всього					891

Витрати на силову електроенергію (Ве) розраховують за формулою:

$$B_e = \sum_{i=1}^{n} \frac{W_{yi} \cdot t_i \cdot Ц_e \cdot K_{впі}}{\eta_i},$$

де Wyi – встановлена потужність обладнання на певному етапі розробки, кВт; ti – тривалість роботи обладнання на етапі дослідження, год; Це – вартість 1 кВт-години, на 2024 рік це 10 грн/кВт електроенергії, грн; (вартість електроенергії визначається за даними енергопостачальної компанії); Квпі – коефіцієнт, що враховує використання потужності, Квпі <1.

Ве = 10,5 * 0,4 * 100 * 0,9 = 378 грн.

Інші витрати Ів можна прийняти як (50...100)% від суми основної заробітної плати розробників та робітників, які були виконували дану роботу, тобто:

Ів = 0,75 * (Зо + Зр) = 0,75 * (19000+ 1920) = 15690 грн.

Витрати за статтею «Службові відрядження» розраховуються як 20…25% від суми основної заробітної плати дослідників та робітників за формулою:

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

де Нсв – норма нарахування за статтею «Службові відрядження».

Всв = 0,25 * (19000+ 1920) = 5036,36 грн.

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

Взаг = Всв + Ів + Ве + А + Здод + Зр + Зо + М = 5036,36 + 15690 + 378 + 5625+ 2092 + 19000 + 1920 + 891 = 50632,36 грн.

4.3 Розрахунок економічної ефективності науково-технічної розробки при її можливій комерціалізації потенційним інвестором

В ринкових умовах узагальнюючим позитивним результатом, що його може отримати потенційний інвестор від можливого впровадження результатів тієї чи іншої науково-технічної розробки, є збільшення у потенційного інвестора величини чистого прибутку.

Результати дослідження проведені за темою «Метод анонімізації мережевого трафіку» передбачають комерціалізацію протягом 2-х років реалізації на ринку.

В цьому випадку основу майбутнього економічного ефекту будуть формувати:

ΔN – збільшення кількості споживачів яким надається відповідна інформаційна послуга у періоди часу, що аналізуються. Збільшення кількості споживачів яким надається відповідна інформаційна послуга наведено у таблиці 4.8;

N - кількість споживачів яким надавалась відповідна інформаційна послуга у році до впровадження результатів нової науково-технічної розробки, приймемо 20000 осіб;

Цб - вартість послуги у році до впровадження інформаційної системи, приймемо 200 грн;

±ΔЦо - зміна вартості послуги від впровадження результатів, приймемо зростання на + 40,00 грн.

Таблиця 4.8 – Збільшення кількості споживачів яким надається відповідна інформаційна послуга

Показник	1-й рік	2-й рік	2-й рік
Збільшення кількості споживачів, осіб	100	500	300

Можливе збільшення чистого прибутку у потенційного інвестора для кожного із 4-х років, протягом яких очікується отримання позитивних результатів від можливого впровадження та комерціалізації науково-технічної розробки, розраховуємо за формулою:

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

де λ – коефіцієнт, який враховує сплату потенційним інвестором податку на додану вартість. У 2024 році ставка податку на додану вартість становить 20%, а коефіцієнт λ =0,8333;

ρ – коефіцієнт, який враховує рентабельність інноваційного продукту. Рекомендується брати ρ =0,2...0,5;

ϑ – ставка податку на прибуток, який має сплачувати потенційний інвестор, у 2024 році ϑ =18%.

Цо – основний якісний показник, який визначає ціну реалізації нової науково-технічної розробки в аналізованому році, Цо = Цб ± ΔЦо ;

$\Delta П_1 = (40 * 20000 + 200 * 100) * 0{,}8333 * 0{,}4 * (1 - 0{,}18) =$ 224124,37 грн.

$\Delta П_2 = (40 * 20000 + 200 * 500) * 0{,}8333 * 0{,}4 * (1 - 0{,}18) =$ 245990,16 грн.

$\Delta П_3 = (40 * 20000 + 200 * 300) * 0{,}8333 * 0{,}4 * (1 - 0{,}18) =$ 235057,26 грн.

Приведена вартість збільшення всіх чистих прибутків ПП, що їх може отримати потенційний інвестор від можливого впровадження та комерціалізації науково-технічної розробки:

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

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

Т – 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 =0,05…0,15;

t – період часу (в роках) від моменту початку впровадження науковотехнічної розробки до моменту отримання потенційним інвестором додаткових чистих прибутків у цьому році.

$$ПП = \frac{224124{,}37}{(1+0{,}1)^1} + \frac{245990{,}16}{(1+0{,}1)^2} + \frac{235057{,}26}{(1+0{,}1)^3} = 203749{,}42 + 203297 + 176601{,}99 =$$

583648,41 грн.

Далі розраховують величину початкових інвестицій PV , які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної розробки. Для цього можна використати формулу:

$$PV = k_{інв} \cdot ЗВ,$$

де Кінв– коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію. Це можуть бути витрати на підготовку приміщень, розробку технологій, навчання персоналу, маркетингові заходи тощо; зазвичай інв k =2...5, але може бути і більшим;

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

Загальні витрати ЗВ на завершення науково-дослідної роботи та оформлення її результатів розраховуються за формулою:

$$ЗВ = \frac{В_{заг}}{\eta},$$

де η – коефіцієнт, який характеризує етап виконання науководослідної роботи. Так, якщо науково-технічна розробка знаходиться на стадії: науково-дослідних робіт, то η =0,1; технічного проектування, то η =0,2; розробки конструкторської документації, то η =0,3; розробки технологій, то η =0,4; розробки дослідного зразка, то η =0,5; розробки промислового зразка, то η =0,7; впровадження, то η =0,9.

ЗВ = 50632,36 / 0,5 = 101264,72 грн.

PV = 2 * 101264,72 = 202529,44 грн.

Тоді абсолютний економічний ефект Еабс або чистий приведений дохід (NPV, Net Present Value) для потенційного інвестора від можливого впровадження та комерціалізації науково-технічної розробки становитиме:

$$E_{абс} = ПП - PV,$$

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

PV – теперішня вартість початкових інвестицій, грн.

Якщо величина Еабс буде мати велике додатне значення, то це може свідчити про потенційну зацікавленість інвесторів у впровадженні та комерціалізації цієї науково-технічної розробки. Але для остаточного прийняття

рішення про впровадження науково-технічної розробки та виведення її на ринок цього недостатньо.

$$E_{абс} = 583648{,}41 - 202529{,}44 = 381118{,}97 \text{ грн.}$$

Для остаточного прийняття рішення з цього питання необхідно розрахувати внутрішню економічну дохідність Ев або показник внутрішньої норми дохідності (IRR, Internal Rate of Return) вкладених інвестицій та порівняти її з так званою бар'єрною ставкою дисконтування, яка визначає ту мінімальну внутрішню економічну дохідність, нижче якої інвестиції в будь-яку науково-технічну розробку вкладати буде економічно недоцільно. Внутрішня економічна дохідність інвестицій Ев , які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науковотехнічної розробки, розраховується за формулою:

$$E_в = \sqrt[Tж]{1 + \frac{E_{абс}}{PV}} - 1,$$

де Еабс – абсолютний економічний ефект вкладених інвестицій, грн;

PV – теперішня вартість початкових інвестицій, грн;

Тж – життєвий цикл науково-технічної розробки, тобто час від початку її розробки до закінчення отримування позитивних результатів від її впровадження, роки.

$$E_в = \sqrt[4]{1 + \frac{381118{,}97}{202529{,}44}} - 1 = 0{,}37.$$

Далі визначають бар'єрну ставку дисконтування мін τ , тобто мінімальну внутрішню економічну дохідність інвестицій, нижче якої кошти у впровадження науково-технічної розробки та її комерціалізацію вкладатися не будуть.

Мінімальна внутрішня економічна дохідність вкладених інвестицій мін τ визначається за формулою:

$$\tau_{мін} = d + f,$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2024 році в Україні d =0,1...0,12;

f – показник, що характеризує ризикованість вкладення інвестицій; зазвичай величина f =0,05...0,5, але може бути і значно вищою.

Якщо величина Ев > τмін , то потенційний інвестор може бути зацікавлений у фінансуванні впровадження науково-технічної розробки та виведенні її на ринок, тобто в її комерціалізації.

$$\tau_{мін} = 0,10 + 0,5 = 0,6$$

Далі розраховуємо період окупності інвестицій Ток (DPP, Discounted Payback Period), які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$T_{ок} = \frac{1}{E_в},$$

де Ев – внутрішня економічна дохідність вкладених інвестицій.

$$T_{ок} = \frac{1}{0,37} = 2,7.$$

Якщо Ток < 3-х років, то це свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження цієї розробки та виведення її на ринок.

4.4 Висновки до розділу 4

Згідно з результатами проведених досліджень, рівень комерційного потенціалу науково-технічної розробки за темою «Система протидії кіберзлочинності в ігровій індустрії» становить 37,5 балів, що вказує на її комерційну цінність (рівень комерційного потенціалу перевищує середній).

Оцінка технічних параметрів розробки, відповідно до узагальненого коефіцієнта якості, демонструє, що запропоноване рішення перевищує ефективність існуючих аналогів приблизно у 1,73 рази.

Термін окупності проекту становить 2,7 роки, що є меншим за три роки. Це свідчить про високу інвестиційну привабливість розробки та створює вагомі

передумови для залучення потенційних інвесторів з метою фінансування її впровадження і подальшого виходу на ринок.

Таким чином, проведення науково-дослідної роботи за темою «Система протидії кіберзлочинності в ігровій індустрії» є економічно доцільним. Запропонована розробка має значний потенціал для комерціалізації, є перспективною в умовах сучасного ринку та може сприяти підвищенню рівня безпеки у сфері обробки та передачі даних.

ВИСНОВОК

В результаті виконання даної кваліфікаційної роботи було проведено детальний аналіз кіберзлочинності в ігровій індустрії. Після проведення аналізу даної теми, було виявлено одну з най вразливіших областей, а саме, втручання у внутрішні файли гри. Для забезпечення більшої надійності було розроблено додаткові методи для підвищення надійності під час передачі даних між користувачем та сервером. Також було розглянуто можливість обмеження доступу да ігрового коду, що розміщується на персональному комп'ютері користувача, для запобігання небажаного вручання в роботу коду та ігрового процесу в цілому.

Під час виконання дослідження було розглянуто різні методи запобігання кіберзлочинності в ігровій індустрії. Найбільш актуальними серед них є генерація секретного ключа на основі апаратних даних, використання контрольних сум та HMAC для перевірки цілісності даних, а також ізоляція клієнтського коду за допомогою хмарних сервісів. Кожен із цих підходів робить значний внесок у захист ігрових продуктів від шахрайства та інших кіберзагроз.

Генерація секретного ключа на основі апаратних даних забезпечує високий рівень захисту завдяки залежності секретного ключа від унікальних характеристик апаратного забезпечення конкретного пристрою. Основні етапи цього процесу передбачають ідентифікацію унікальних апаратних характеристик, таких як серійний номер процесора, жорсткого диска чи MAC-адреса, та генерацію ключа на основі цих даних. Це дозволяє створити систему, яка захищає ключ від несанкціонованого копіювання та ускладнює можливість повторного входу з інших пристроїв під час спроб порушення безпеки. Додаткова перевірка відповідності апаратного середовища під час використання ключа унеможливлює шахрайські дії, оскільки при виявленні невідповідності сервер може автоматично заблокувати доступ до гри.

Контрольна сума та HMAC є важливими компонентами для забезпечення цілісності та достовірності даних, що передаються між клієнтом і сервером. Використання HMAC забезпечує перевірку, що пакет даних, отриманий сервером, дійсно був створений клієнтом і не був змінений під час передачі.

Розроблений програмний засіб забезпечує комплексний захист даних, інтегруючи перевірку цілісності, шифрування та виявлення спроб підробки. Для захисту конфіденційності даних використовується алгоритм шифрування AES, який реалізований із динамічним вектором ініціалізації (IV). Завдяки цьому алгоритм ефективно шифрує будь-які текстові дані перед їх відправкою, запобігаючи доступу до них сторонніх осіб. Вектор ініціалізації гарантує унікальність кожного зашифрованого блоку, навіть якщо вихідні дані повторюються. Це значно підвищує рівень безпеки переданих повідомлень.

В результаті отримано програмний засіб, який поєднує в собі ефективність алгоритмів криптографії та надійність перевірки даних, що дозволяє значно підвищити рівень захищеності інформації під час передачі між клієнтом і сервером.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. 10 найбільших ігрових компаній світу URL: https://mezha.media/articles/10-2-naybilshykh-ihrovykh-kompaniy-svitu/

2. Консольні ігри - Блог / Центр Знань URL: https://www.kingston.com/ua/blog/topic/console-gaming

3. Новини і статті по темі «Мобільні ігри» URL: https://itc.ua/ua/tag/mobilni-igri/

4. PC - Game URL: https://www.pcgamer.com/

5. Технологічні інновації в гральній індустрії: Що принесе майбутнє? URL: https://mynizhyn.com/news/info/28783-tehnologichni-innovaciyi-v-gralnii-industriyi-sho-prinese-maibutne.html

6. Моделі монетизації продукту: від безкоштовних до платних моделей URL: ttps://www.londonproduct.academy/ post/modeli-monetizaciyi-produktu-vid-bezkoshtovnih-do-platnih-modeley

7. Відеоігри – король індустрії розваг URL: https://www.epravda.com.ua/publications/2023/05/18/700238/

8. Що таке фішинг? URL: https://top-ai.com.ua/resursy/slovnyk-z-kiberbezpeky/shho-take-fishyng/

9. Що таке брутфорс атака і які є методи захисту URL: https://foxminded.ua/brute-force/

10. Опыт борьбы с распространителями ботов и читов в играх URL: https://web.telegram.org/a/#690839539

11. Обзор Authy URL: https://bitgid.com/authy/

12. МОБІЛЬНИЙ АВТЕНТИФІКАТОР URL: https://help.steampowered.com/uk/wizard/HelpWithSteamIssue/?issueid=809#:~:text=%D0%9C%D0%BE%D0%B1%D1%96%D0%BB%D1%8C%D0%BD%D0%B8%D0%B9%20%D0%B0%D0%B2%D1%82%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D1%96%D0%BA%D0%B0%D1%82%D0%BE%D1%80%20Steam%20G

uard%20%E2%80%94%20%D1%86%D0%B5,%D0%B4%D0%BE%D0%B4%D0%B0%D1%82%D0%BA%D0%BE%D0%B2%D0%B8%D0%B9%20%D1%80%D1%96%D0%B2%D0%B5%D0%BD%D1%8C%20%D0%B7%D0%B0%D1%85%D0%B8%D1%81%D1%82%D1%83%20%D0%B0%D0%BA%D0%B0%D1%83%D0%BD%D1%82%D0%B0%20Steam.

13. Що таке OpenSSL URL: https://www.openssl.org/

14. Easy Anti-Cheat URL: https://www.easy.ac/en-US

15. Що таке VAC (Valve Anti-Cheat) URL: https://steamcommunity.com/sharedfiles/filedetails/?l=ukrainian&id=483204174

16. Підручник Splunk для початківців URL: https://www.guru99.com/uk/splunk-tutorial.html

17. polygon URL: https://www.polygon.com/gaming

18. Tarkov Ban Tracker URL: https://tarkovbot.eu/tools/bantracker

19. Статистичні дані з античіт-системи Easy Anti-Cheat та огляди URL: https://www.techopedia.com/

20. Офіційна статистика від Riot Games URL: https://www.riotgames.com/en

21. VAC та статистика URL: https://vaclist.net/

22. Новини Ubisoft URL: https://www.ubisoft.com/en-gb/game/rainbow-six/siege/news-updates/1Zo3SN6YLn05kMy2f6kwb6/anticheat-status-update-february-2024

23. Серійні номера та інформація про несправні жорсткі диски URL: https://beehosting.pro/ru/kb/serijnye-nomera-i-informacziya-o-neispravnyh-zhestkih-diskah/

24. Підписуємо дані: HMAC URL: https://habr.com/ru/articles/262341/

25. Про середовище програмування C# URL: https://foxminded.ua/seredovyshche-prohramuvannia-si-sharp/

26. C#: що це за мова та де її використовують URL: https://robotdreams.cc/uk/blog/284-s-chto-eto-za-yazyk-i-gde-ego-ispolzuyut

27. Розробка привабливих кросплатформових додатків URL: https://visualstudio.microsoft.com/ru/vs/

28. Що таке AES-шифрування URL: https://www.vpnunlimited.com/ua/help/cybersecurity/aes-256?srsltid=AfmBOorRdqeOKQ4dcbwCx88y389a4naVm0Ry2tjFjnm34DTsuFllczDF

76

Додаток А

ПРОТОКОЛ ПЕРЕВІРКИ НАВЧАЛЬНОЇ (КВАЛІФІКАЦІЙНОЇ) РОБОТИ

Назва роботи: Система протидії кіберзлочинності в ігровій індустрії

Автор роботи: Лазуренко Іван Дмитрович
Тип роботи: магістерська кваліфікаційна робота
Підрозділ кафедра захисту інформації ФІТКІ
Керівник Лужецький Володимир Андрійович, д.т.н., професор

Показники звіту подібності

Turnitin	
Оригінальність	93 %
Загальна схожість	7 %

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 автора. Роботу направити на доопрацювання.

□ 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.

Заявляю, що ознайомлений з повним звітом подібності, який був згенерований Системою щодо роботи.

Автор роботи _____ Іван ЛАЗУРЕНКО
(підпис) (прізвище, ініціали)

Особа, відповідальна за перевірку _____ Валентина КАПЛУН
(підпис) (прізвище, ініціали)

Керівник роботи _____ Володимир ЛУЖЕЦЬКИЙ
(підпис) (прізвище, ініціали)

77

ІЛЮСТРАТИВНА ЧАСТИНА

СИСТЕМА ПРОТИДІЇ КІБЕРЗЛОЧИННОСТІ В ІГРОВІЙ ІНДУСТРІЇ

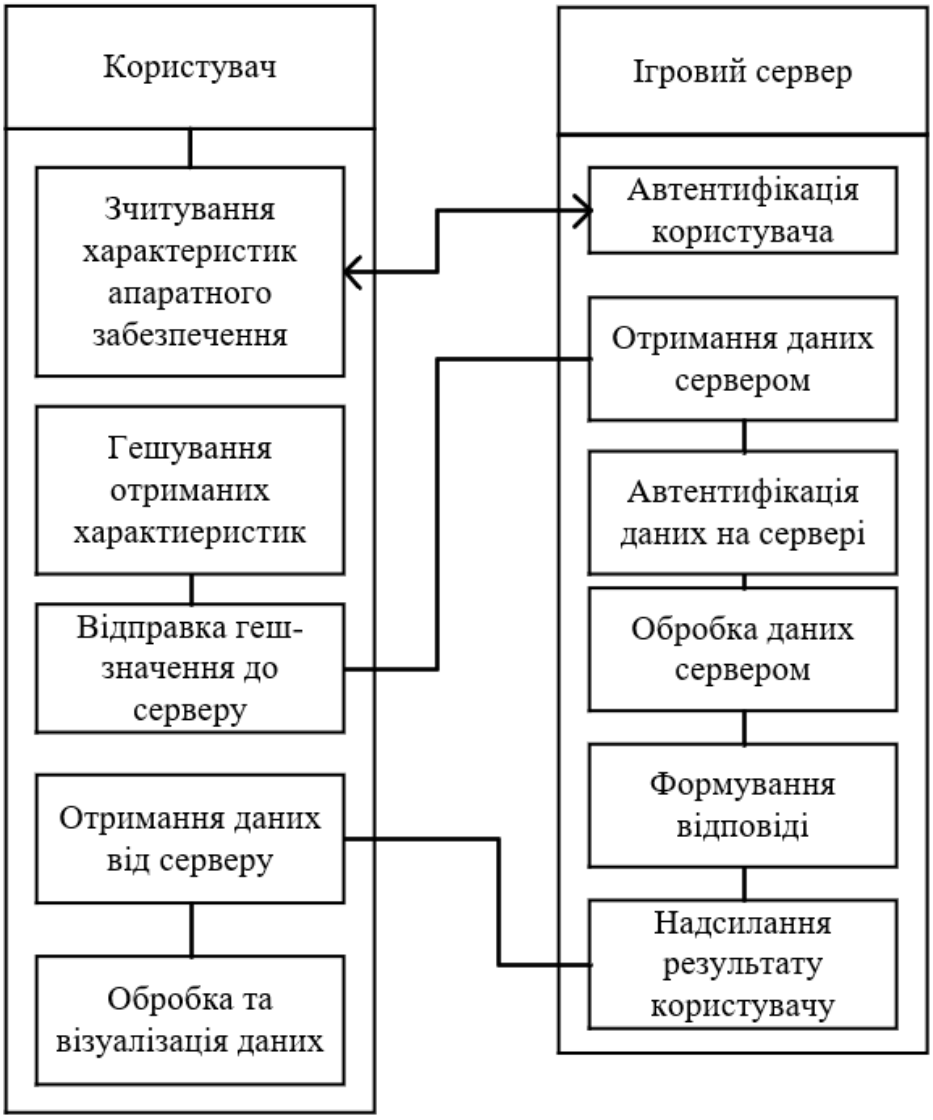
(Назва магістерської кваліфікаційної роботи)

Виконав: студент групи 1БС-23 м
спеціальності 125 Кібербезпека та захист
інформації
_____ Іван ЛАЗУРЕНКО
13.12 2024 р.

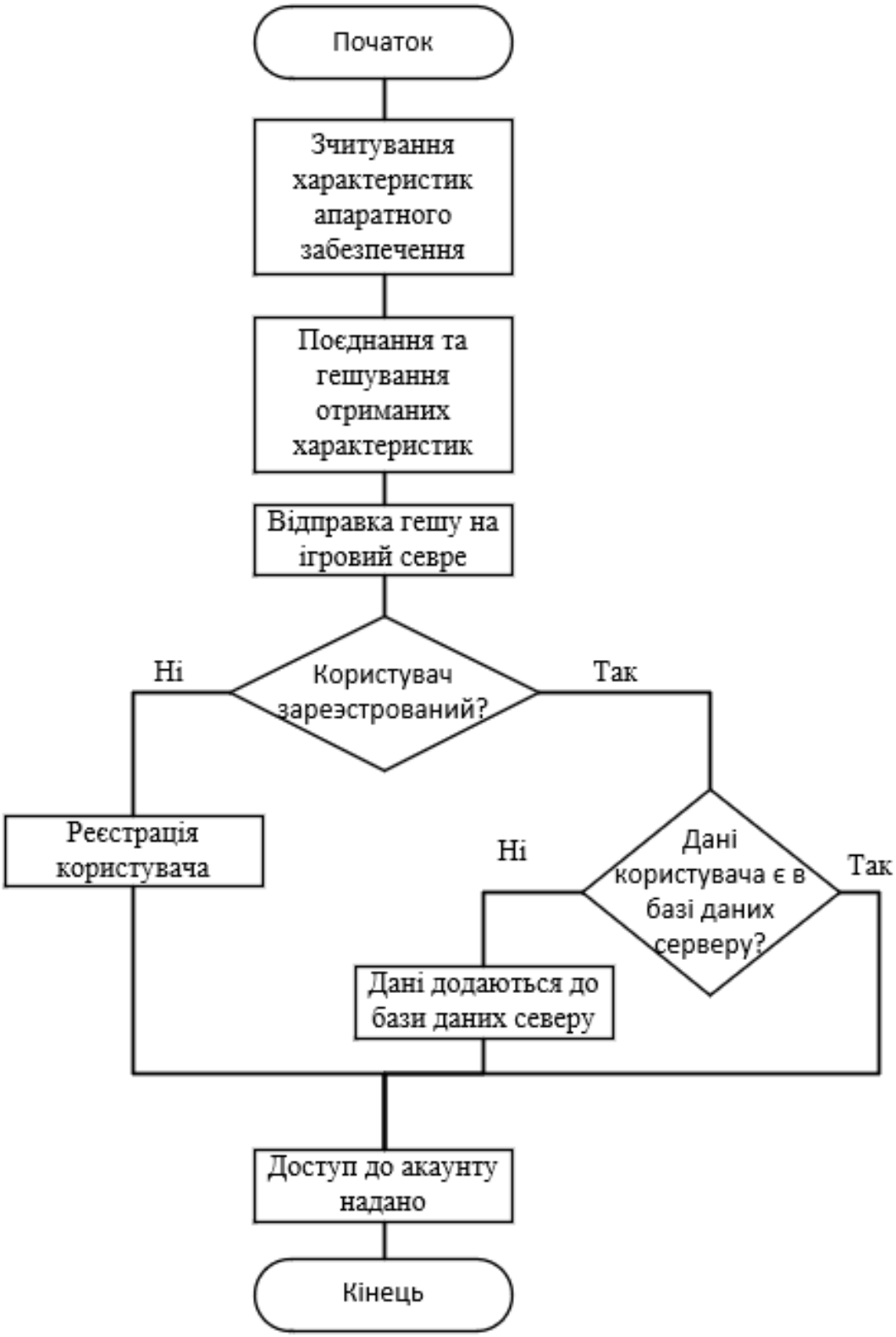
Керівник: завідувач кафедри ЗІ, д. т. н.,
_____ Володимир ЛУЖЕЦЬКИЙ
13.12 2024 р.

МЕТОД 1 ОБМЕЖЕННЯ ДОСТУПУ ДО ІГРОВИХ ДАНИХ

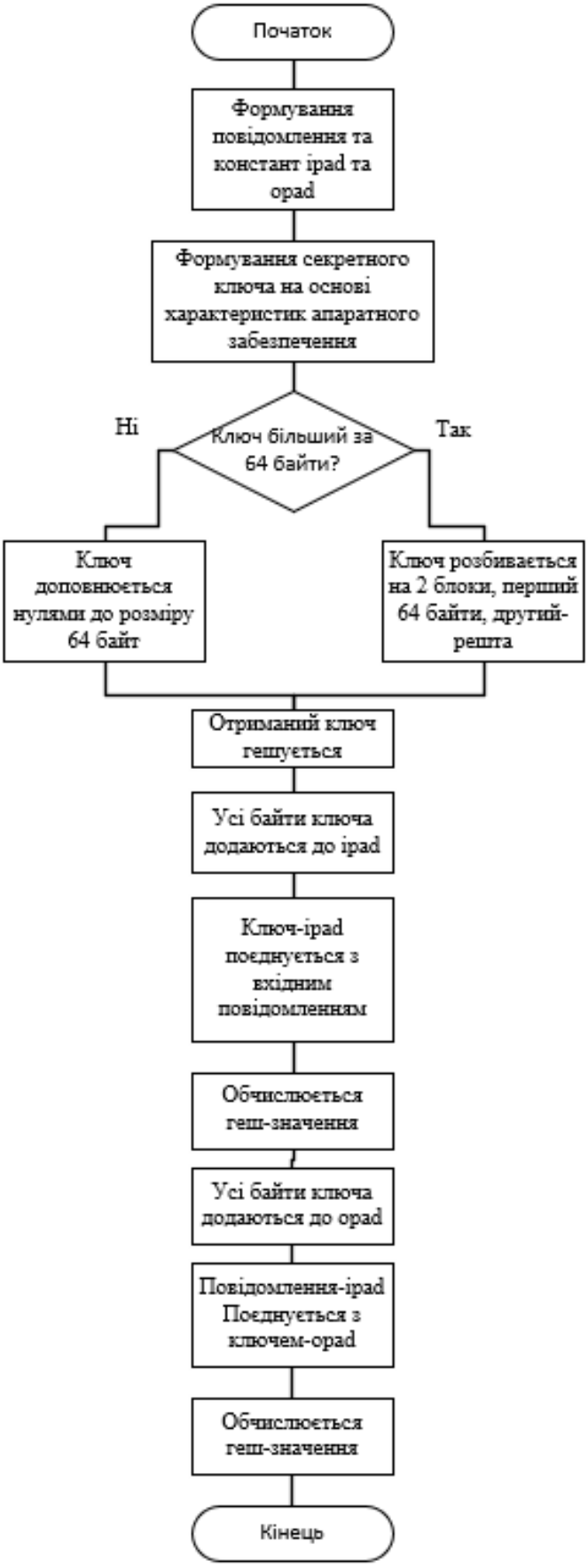
МЕТОД 2 ОБМЕЖЕННЯ ДОСТУПУ ДО ІГРОВИХ ДАНИХ



МЕТОД АВТЕНТИФІКАЦІЇ КОРИСТУВАЧА НА ОСНОВІ ХАРАКТЕРИСТИК АПАРАТНОЇ СКЛАДОВОЇ



АЛГОРИТМ ФОРМУВАННЯ HMAC



РЕЗУЛЬТАТИ АВТЕНТИФІКАЦІЇ КОРИСТУВАЧА

```
cpu ID - WD12JWDD223312ID
disk ID - 50026B724A066122
mac address - D8:5E:D3:5C:F7:79
combined data - 178BFBFF00A50F0050026B724A066122D8:5E:D3:5C:F7:79
Generated Hardware Hash: c6afc67010135ac700b50aec690f577bfc8c8968a47661ee10ae7422dc0ec85a
```

```
cpu ID - WD12JWDD223312ID
disk ID - 50026B724A066122
mac address - D8:5E:D3:5C:F7:79
combined data - 178BFBFF00A50F0050026B724A066122D8:5E:D3:5C:F7:79
Hardware Hash Verified: True


c6afc67010135ac700b50aec690f577bfc8c8968a47661ee10ae7422dc0ec85a
```

```
cpu ID - 178BFBFF00A50F00
disk ID - 50026B724A066122
mac address - D8:5E:D3:5C:F7:79
combined data - 178BFBFF00A50F0050026B724A066122D8:5E:D3:5C:F7:79
d4e2dc7bc8c5e84c647a55f7e5f1c5ced37ff14b8e07d0a46625fa3005d20dd7
Hardware Hash Verified: True


d4e2dc7bc8c5e84c647a55f7e5f1c5ced37ff14b8e07d0a46625fa3005d20dd7
c6afc67010135ac700b50aec690f577bfc8c8968a47661ee10ae7422dc0ec85a
```

РЕЗУЛЬТАТИ АВТЕНТИФІКАЦІЇ ГРОВИХ ДАНИХ

```
Enter user name
jagernaut
Enter item price
1000
Checksum - LPYy8hqA62/ni44vWn6lDMJKxVKXMzyX55ps5XhHyc4=
Encrypted Data - W9bsklZPDcbGdVtbbzNz58mPa1ux9v2oHrvLyvanfkY=
Validation of data integrity: Passed
Decrypted Data - 1000
```

```
Enter user name
jagernaut
Enter item price
999999999999
Checksum - X8Dt3QR95I+nr3ERtpDnKsKILSNzoDeIlvQnMD2DYxY=
Encrypted Data - xbCP37pKIayQcOpsh04Y3KIUgvSk3rDKF9cDJ/1sJwY=
Validation of data integrity: Failed
Decrypted Data - 999999999999
```

ВІДГУК ОПОНЕНТА

на магістерську кваліфікаційну роботу

студента Лазуренко Івана Дмитровича
(прізвище, ім'я, по батькові)

на тему "Методи протидії кіберзлочинності в ігровій індустрії"

У сучасному цифровому світі ігрова індустрія є однією з найбільш швидкозростаючих і впливових галузей світової економіки. Тому вона є об'єктом особливої уваги кіберзлочинців. Створення засобів протидії кіберзлочинності є важливою задачею для фахівців з кібербезпеки, що й обумовлює актуальність теми магістерської кваліфікаційної роботи.

У першому розділі проаналізовано основні типи кіберзлочинів в ігровій індустрії та існуючі методи протидії кіберзлочинності.

У другому розділі описується розробка методів підвищення рівня захисту ігрових даних шляхом обмеження доступу до них гравців. Ці методи передбачають використання шифру AES для забезпечення конфіденційності даних, автентифікацію даних на основі HMAC і генерування секретного ключа на основі апаратних характеристик персонального комп'ютера гравця.

Третій розділ присвячено розробці та тестуванню програмного засобу для захищеного передавання даних між персональним комп'ютером гравця та сервером.

Текстову та ілюстративну частини роботи виконано у повному обсязі згідно з індивідуальним завданням, їх оформлення відповідає вимогам, що висуваються до магістерських кваліфікаційних робіт.

Зауваження до змісту магістерської кваліфікаційної роботи.

1. У пояснювальній записці відсутня інформація про ігрові дані, які є найбільш критичними з точки зору необхідності їх захисту. Крім того, не відзначено, що є більш важливим - конфіденційність чи цілісність ігрових даних.

2. Запропоновані два методи обмеження доступу до ігрових даних порівняно лише на якісному рівні. Доцільно було б навести кількісні оцінки часу доступу, рівня захисту та вартості реалізації.

Вважаю, що магістерська кваліфікаційна робота заслуговує на оцінку С за ECTS, а її автор - Лазуренко Іван Дмитрович заслуговує на присвоєння кваліфікації ступінь вищої освіти «магістр», галузь знань 12 «Інформаційні технології», спеціальність 125 «Кібербезпека та захист інформації», освітня програма «Безпека інформаційних і комунікаційних систем».

Опонент
зав. кафедри ПЗ
д.т.н., професор

Олександр РОМАНЮК

ВІДГУК

на магістерську кваліфікаційну роботу студента групи 1БС-23м Лазуренко Івана Дмитровича "Методи протидії кіберзлочинності в ігровій індустрії"

Тема магістерської кваліфікаційної роботи є актуальною, оскільки ігрова індустрія в сучасному світі є однією з найбільш швидкозростаючих та впливових галузей світової економіки.

Науковою новизною роботи є метод автентифікації гравців на основі апаратних характеристик персонального комп'ютера, а також метод дворівневого захисту ігрових даних, що базується на використанні алгоритмів шифрування AES та контролю цілісності HMAC.

Практичною цінністю магістерської кваліфікаційної роботи є програмні засоби, що генерують секретний ключ на основі апаратних характеристиках комп'ютера та реалізують захищене передавання ігрових даних між клієнтом (гравцем) та сервером.

Під час роботи студент показав достатній рівень теоретичних знань і практичних навичок. Індивідуальне завдання на магістерську кваліфікаційну роботу в основному виконано. Оформлення пояснювальної записки відповідає вимогам, що висуваються до магістерських кваліфікаційних робіт.

Недоліком роботи є те, що не наведено оцінок можливої затримки передавання даних, обумовленої використанням одночасно шифрування та автентифікації. Наслідком цього є відсутність рекомендацій щодо граничних швидкостей реалізації ігрових примітивів.

Вважаю, що магістерська кваліфікаційна робота заслуговує на оцінку С за ECTS, а її автор - Лазуренко Іван Дмитрович заслуговує на присвоєння кваліфікації ступінь вищої освіти «магістр», галузь знань 12 «Інформаційні технології», спеціальність 125 «Кібербезпека та захист інформації», освітня програма «Безпека інформаційних і комунікаційних систем».

Керівник
магістерської кваліфікаційної роботи
зав. кафедри ЗІ, д.т.н., професор Володимир ЛУЖЕЦЬКИЙ