Хмельницький національний університет
Факультет інформаційних технологій
Кафедра інженерії програмного забезпечення

КВАЛІФІКАЦІЙНА РОБОТА

Метод удосконалення передачі даних протоколом TCP за допомогою алгоритму верифікації повідомлень та алгоритму Діффі-Хелмана

Рівень вищої освіти Другий (магістерський)

Галузь знань 12 «Інформаційні технології»

Спеціальність 121 «Інженерія програмного забезпечення»

Освітня програма Освітньо-професійна програма «Інженерія програмного забезпечення»

Шифр КвРІПЗ.180117.01.07.ПЗ

Виконав студент 2 курсу, група ІПЗм-22-1 _____ Євгеній СЛУТЯК
Підпис Ім'я, ПРІЗВИЩЕ

Керівник канд. техн. наук, доцент _____ Галина РАДЕЛЬЧУК
Науковий ступінь, звання Підпис Ім'я, ПРІЗВИЩЕ

Нормоконтролер канд. техн. наук, доцент _____ Галина РАДЕЛЬЧУК
Підпис Ім'я, ПРІЗВИЩЕ

До захисту допускаю:
Завідувач кафедри
інженерії програмного забезпечення _____ Леонід БЕДРАТЮК
Підпис Ім'я, ПРІЗВИЩЕ

Дата

Хмельницький 2023

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет Інформаційних технологій
Кафедра Інженерії програмного забезпечення
Рівень вищої освіти Другий (магістерський)
Галузь знань 12 «Інформаційні технології»
Спеціальність 121 «Інженерія програмного забезпечення»
Освітня програма Освітньо-професійна програма «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри ІПЗ
Л. П. Бедратюк
01.09.2023 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Слутяку Євгенію Івановичу
Прізвище, ім'я, по батькові здобувача

1. Тема роботи Метод удосконалення передачі даних протоколом TCP за допомогою алгоритму верифікації повідомлень та алгоритму Діффі-Хелмана

Керівник роботи Радельчук Галина Іванівна
Прізвище, ім'я, по батькові, науковий ступінь, вчене звання

кандидат технічних наук, доцент
Затверджена наказом ректора університету від 15.08.2023 р. № 30
2. Строк подання студентом роботи на кафедру 01.12.2023 р.
3. Вихідні дані до роботи Матеріали науково-дослідної практики
4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)
1 Дослідження предметної області та постановка задачі
2 Концепції, моделі та методи вирішення задачі
3 Алгоритми та технології вирішення задачі
4 Реалізація та тестування програмної системи
5. Перелік графічного матеріалу (із зазначенням обов'язкових креслень)
Презентаційні матеріали (слайди)

6. Консультанти розділів кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Антиплагіат	Форкун Ю. В., доцент	25.11.23	8.12.23
Нормоконтроль	Радельчук Г. І., доцент	25.11.23	11.12.23

7. Дата видачі завдання «_01_» __вересня__ 2023 р.

КАЛЕНДАРНИЙ ПЛАН

Назва етапів (розділів) кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1 Вивчення предметної області; формулювання мети та задач дослідження; визначення об'єкта та предмета дослідження; формування логістичної структури кваліфікаційної роботи (КвР)	01.09-10.09.2023	
2 Робота над розділом 1 КвР – аналіз предметної області, останніх досліджень та джерел; порівняльний аналіз переваг та недоліків існуючих рішень; методологічні підходи до вирішення задачі за темою дослідження; висновки, постановка задачі	11.09-25.09.2023	
3 Робота над розділом 2 КвР – концепції передачі даних у мережі Інтернет; моделі та методи покращення передачі даних; висновки.	26.09-10.10.2023	
4 Робота над науковими статтями	11.10-30.10.2023	
5 Робота над розділом 3 КвР – алгоритми вирішення задачі; визначення вимог до програмної системи; проєктування програмної системи; аналіз та вибір засобів реалізації програмної системи; висновки	11.10-26.10.2023	
6 Робота над розділом 4 КвР – програмна реалізація; результати тестування системи та їх аналіз; Оцінка ефективності моделей та методів вирішення задач; висновки	27.10-17.11.2023	
7 Попередній захист КвР	Листопад	згідно графіка
8 Узгодження постановки задачі, отриманих результатів та висновків; написання вступу, загальних висновків, оформлення джерел посилання та додатків; оформлення пояснювальної записки та графічних матеріалів згідно вимог чинних стандартів	18.11-30.11.2023	
9 Перевірка роботи на наявність плагіату; нормоконтроль; брошурування пояснювальної записки; підготовка супровідних документів	01.12-04.12.2023	
10 Здача КвР на кафедру; підготовка КвР для розміщення у репозитарії ХНУ; підготовка до захисту та захист КвР	з 01.12.2023 р.	

Студент _____ Є. І. СЛУТЯК
Підпис Ім'я, ПРІЗВИЩЕ

Керівник роботи _____ Г. І. РАДЕЛЬЧУК
Підпис Ім'я, ПРІЗВИЩЕ

РЕФЕРАТ

Тема кваліфікаційної роботи: «Метод удосконалення передачі даних протоколом TCP за допомогою алгоритму верифікації повідомлень та алгоритму Діффі-Хелмана».

Автор роботи: Слутяк Євгеній Іванович.

Керівник роботи: Радельчук Галина Іванівна.

Пояснювальна записка: 94 с., 18 рис., 7 табл., 4 дод., 19 джерел.

ІНТЕРНЕТ, ПРОТОКОЛ ПЕРЕДАЧІ ДАНИХ, АВТЕНТИФІКАЦІЯ, ВЕРИФІКАЦІЯ, АЛГОРИТМ ДІФФІ-ХЕЛМАНА, TCP.

Об'єктом дослідження є процес передачі даних у мережі Інтернет.

Мета дослідження – удосконалення передачі даних, що понизить ймовірність перехоплення даних користувача під час їх передачі у мережі Інтернет або понизить ймовірність їх розшифрування у разі перехоплення.

У роботі використані наступні методи дослідження та апаратура:

– спостереження, експеримент, абстрагування, аналіз та синтез, формалізація;

– сучасні інструментальні засоби проєктування та програмування;

– персональний комп'ютер.

У кваліфікаційній роботі досліджено процес передачі даних у мережі Інтернет за допомогою різних протоколів та їх типів; визначено проблеми, наявні у сфері, а також визначено шляхи їх вирішення; сформовано основні вимоги до створюваної програмної системи і функції, які вона повинна виконувати.

Під час виконання дослідження було удосконалено наявний метод передачі даних у мережі Інтернет шляхом проєктування покращеного процесу передачі даних з використанням повторюваної автентифікації і верифікації користувача та розбиття даних на окремі транзакції. Обґрунтовано доцільність проєктування та розробки програмної системи для імплементації розробленого методу і удосконалення процесу передачі даних для користувачів мережі Інтернет. Розглянуто протоколи передачі даних, алгоритми шифрування даних та обрано оптимальні рішення для використання в удосконаленому методі. Результатом дослідження є покращений метод передачі даних в мережі Інтернет.

На основі визначених вимог розроблено функціональну модель передачі даних та виконано її програмну реалізацію. Під час розробки програмної системи було використано наступні технології: мову програмування C#, фреймоворки WebAPI, Windows Forms, Entity Framework, LINQ; системи керування базами даних MSSQL, SQLite (для розробки); xUnit (для тестування).

У підсумку проведено апробацію отриманих результатів та реалізації програмної системи. Впроваджена система має високі конкурентні шанси на ринку, оскільки імплементований метод удосконалення передачі даних має значні переваги порівняно з традиційними методами. Тому її рекомендовано інтегрувати компаніям, які зацікавлені у підвищенні якості, надійності та безпеки передачі даних у мережі Інтернет.

01.12.2023

ABSTRACT

Master's thesis: «A method of improving data transfer using the TCP protocol using the message verification algorithm and the Diffie-Hellman algorithm».

Author: Slutiak Yevhenii Ivanovych.

Head of work: Galina Ivanovna Radelchiuk.

Master's thesis consists of: 94 p., 18 pc., 7 tb., 4 add., 19 src.

INTERNET, DATA TRANSFER PROTOCOL, AUTHENTICATION, VERIFICATION, DIFFIE-HELLMAN ALGORITHM, TCP.

The object of research of this thesis is data transmission on the Internet.

The purpose of the research is to improve data transmission, which will reduce the probability of interception of user data during their transmission over the Internet, or reduce the probability of their decryption in the event of interception.

The following research methods and equipment were used in the work:

– observation, experiment, abstraction, analysis and synthesis, formalization;

– modern design and programming tools;

– personal computer.

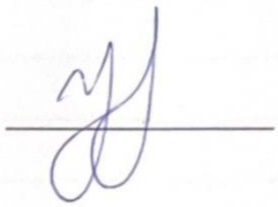
The thesis examines the process of data transmission on the Internet using various protocols and their types; the problems existing in the field are defined, as well as the ways to solve them; the main requirements for the software system being created and the functions it must perform have been formed.

During the research, the existing method of data transfer on the Internet was improved by designing an improved data transfer process using repeated user authentication and verification and splitting data into separate transactions. The feasibility of designing and developing a software system for implementing the developed method and improving the data transmission process for Internet users is substantiated. Data transmission protocols, data encryption algorithms were considered and optimal solutions were selected for use in the improved method. The result of the research is an improved method of data transmission on the Internet.

Based on the defined requirements, a functional model of data transmission was developed and its software implementation was carried out. During the development of the software system, the following technologies were used: C# programming language, frameworks such as WebAPI, Windows Forms, Entity Framework, LINQ; database management systems MSSQL, SQLite (for development); xUnit (for testing).

As a result, the results obtained and the implementation of the software system were tested. The implemented software system has highly competitive chances in the market, because the implemented method of data transfer improvement has significant advantages compared to traditional methods. Therefore, it is recommended to be integrated by companies interested in improving the quality, reliability and security of data transmission on the Internet.

01.12.2023



ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

БД	–	база даних
ІТ	–	інформаційні технології
ПЗ	–	програмне забезпечення
ПС	–	програмна система
AES	–	Advanced Encryption Standard
API	–	Application programming interface
ASCII	–	American Standard Code for Information Interchange
CSS	–	Cascading Style Sheets
DDoS	–	Distributed Denial-of-Service Attack
DES	–	Data Encryption Standard
DNS	–	Domain Name System
DSA	–	Digital Signature Algorithm
EBCDIC	–	Extended Binary Coded Decimal Interchange Code
HTML	–	HyperText Markup Language
HTTPS	–	HyperText Transfer Protocol Secure
IMAP	–	Internet Message Access Protocol
IP	–	Internet Protocol
MSSQL	–	Microsoft Structured Queried Language
OSI	–	Open Systems Interconnection
RC4	–	Rivest cipher 4
SCP	–	Special Containment Procedures
SMS	–	Short Message Service
SQL	–	Structured Queried Language
TCP	–	Transmission Control Protocol
VPN	–	Virtual Private Network

ВСТУП

Однією з найвизначніших відмінностей, яка відрізняє останнє покоління людства від усіх попередніх, є те, що нинішнє покоління взаємодіє між собою найбільш активно, а географія цієї взаємодії не має меж.

Така взаємодія стала можливою завдяки тому, що відбулося 22 жовтня 1969 року [1]. Саме в цей день відбулася перша в історії сучасної мережі Інтернет передача даних між Каліфорнійським університетом та Стенфордом. В той день між двома єдиними користувачами мережі було передано три букви – «LOG». Через понад 50 років після першої відправки мережею Інтернет трьох букв, що являється трьома байтами даних, щодня мережею Інтернет передаватиметься в мільйони разів більше даних [2], які вимірюються ексабайтами даних.

Такий ріст популярності мережі Інтернет зумовлений одночасно кількома факторами. По-перше, це відбувається завдяки розвитку індустрії ІТ. Це пов’язано з ростом популярності мережі Інтернет напряму, оскільки з розвитком технологій розробки різноманітного ПЗ, ігор, офісного програмного забезпечення та інтернету речей в цілому, привабливість усіх можливих сервісів, які можна знайти на теренах мережі Інтернет стає все більш привабливою для користувачів. Найкраще прослідкувати такий розвиток можна завдяки статистиці кількості часу, який користувачі щодня проводять, для прикладу, в соціальних мережах [3]. Станом на середину 2023 року, кількість часу, який кожен користувач проводить в соціальних мережах зріс до 2 годин та 26 хвилин. В той час, коли понад 60% населення світу використовує соціальні мережі.

Ще однією причиною, чому популярність мережі Інтернет значно підвищилася протягом останніх років є те, що більшій кількості населення стала доступною ця мережа. Так, за підрахованою станом на 2023 рік статистико [4], в світі нараховується близько 4.3 мільярдів населення світу мають смартфони, з можливістю доступу до мережі Інтернет. Ще близько 1.5 мільярда населення має можливість доступу до інтернету за допомогою інших пристроїв.

Такий ріст популярності працює взаємно для обох факторів. Завдяки розвитку технологій розробки сервісів для користувачів інтернету росте кількість користувачів і навпаки – завдяки росту кількості користувачів з'являється все більше нових сервісів, підвищується їх ефективність, оптимізуються засоби розробки та, таким чином, розвивається сфера ІТ.

Станом на 1 квартал 2021 року, обсяг інтернет-трафіку розподілявся за такими десятьма категоріями [5]:

– потокове відео (48.9%);
– соціальні мережі (19.3%);
– інтернет-сайти (13.1%);
– повідомлення (6.7%);
– ігри (4.3%);
– ринок (4.1%);
– файлообмінники (1.3%);
– хмарні сховища (1.1%);
– VPN та інше безпекове ПЗ (0.9%);
– аудіо (0.2%).

Звісно ж, такий обсяг інтернет-трафіку, сервісів та користувачів в цілому, став дуже привабливим для зловмисників, які націлили усі свої зусилля на те, щоб використовувати мережу Інтернет, відкриту колись для полегшення повсякденного життя та налаштування комунікації для проведення глобалізації у всьому світі, для власного збагачення за рахунок інших користувачів мережі, які можуть бути недостатньо освіченими та навченими користуванню інтернетом у сфері кібер-гігієни з метою зменшення ризику натрапити на зловмисника, а у випадку натрапляння на зловмисника – з метою припинення взаємодії зі зловмисником не втративши жодних персональних даних чи навіть матеріальних цінностей.

Таким чином, методи удосконалення передачі даних в мережі Інтернет є особливо актуальними в даний час, оскільки методи викрадення даних та вчинення інших злочинних дій розвиваються, тому і методи захисту відповідно необхідно постійно покращувати для покращення безпекової ситуації і мережі Інтернет.

Актуальність роботи полягає у тому, що існуючі користувачі мережі Інтернет весь час знаходяться під ризиком викрадання їх даних. Саме тому необхідно постійно покращувати методи та алгоритми захисту від зловмисників та їх шкідливого програмного забезпечення.

Мета дослідження – удосконалення передачі даних, що понизить ймовірність перехоплення даних користувача під час їх передачі у мережі Інтернет або понизить ймовірність їх розшифрування у разі перехоплення.

Відповідно до визначеної мети дослідження, можна виокремити наступні завдання дослідження:

– провести аналіз сфери безпеки передачі даних;
– дослідити сучасні методи та практики підвищення безпеки передачі даних;
– удосконалити наявні методи та практики для повного або часткового вирішення виявлених проблем;
– на основі створеного удосконаленого методу передачі даних виконати проєктування ПЗ для реалізації цього методу;
– провести програмну реалізацію спроєктованого програмного забезпечення;
– виконати тестування та перевірку отриманих результатів;
– проаналізувати результати проведеного дослідження та сформувати висновок щодо доцільності впровадження удосконаленого методу передачі даних.

Об’єкт дослідження – процес передачі даних у мережі Інтернет.

Предмет дослідження – методи, алгоритми та практики підвищення безпеки передачі даних у мережі Інтернет.

Під час виконання дослідження використані наступні методи.

Емпіричні методи

Спостереження. Темою роботи є побудова моделей та механізмів, пов’язаних із передачею даних в мережі Інтернет, але для того, щоб виділити корисні ознаки, які мають бути імплементовані у розроблюваних рішеннях, слід провести спостереження над існуючими аналогами, визначити властивості та зв’язки між ними.

Експеримент. На етапі дослідження існуючих аналогів слід відтворити певні умови, які потрібні для аналізу імплементованих алгоритмів. Пізніше цей же метод

використовується для аналізу ефективності результативної моделі, яка розробляється та імплементується у ході роботи.

Теоретичні методи:

– абстрагування – один з важливих методів, який дозволяє відкинути несуттєві параметри; від абстрагування напряму залежить ефективність моделі;

– аналіз та синтез – декомпозиція моделі на прості складові, виявлення зв’язків між компонентнами; відповідно, і синтез цих структурних елементів у єдине ціле;

– формалізація – представлення моделі у вигляді програмного коду.

Наукова новизна отриманих результатів полягає в удосконаленні методу передачі даних у мережі Інтернет, який відрізняється розширеними методиками повторюваної автентифікації, повторюваної верифікації та розбиття даних на окремі транзакції, що забезпечує підвищення якості, надійності та безпеки передачі даних у мережі Інтернет.

Практична цінність отриманих результатів полягає в успішній розробці моделей та механізмів удосконалення передачі даних в мережі Інтернет. Завдяки покращеним показникам безпеки та надійності передачі даних, порівняно з традиційними методами та моделями передачі даних, розроблена ПС має високі конкурентні шанси на ринку. Результати перевірки та тестування програмної системи підтверджують її працездатність. Тому її рекомендовано інтегрувати компаніям, які зацікавлені у підвищенні безпеки передачі даних в мережі Інтернет.

Достовірність та обґрунтованість отриманих результатів підтверджується використанням у процесі дослідження таких прийомів:

– перевірка теоретичних положень, нових рішень, ідей експериментальними дослідженнями за допомогою відомих процедур проєктування та тестування;
– працездатність та функціональна придатність розробленої ПС;
– наявність наукової публікації у рецензованому виданні.

За темою кваліфікаційної роботи опубліковано тези доповіді на Всеукраїнській науково-практичній конференції.

1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Аналіз предметної області, останніх досліджень та джерел

Згідно зі звітом Digital 2022 Global Statshot [6], виготовленим аналітичною компанією DataReportal, станом на 2022 рік 63% усього населення світу, тобто понад 5 мільярдів людей, користується мережею Інтернет щодня. Аналогічну статистику було надано і про українців, завдяки опитуванню Київського міжнародного інституту соціології [7]. Так, відповідно до цього звіту, станом на 2022 рік, 82% опитаних українців користуються мережею Інтернет принаймні один раз на тиждень, з яких 78% респондентів користуються інтернетом щодня.

Дана статистика може вказувати, що мережа Інтернет щодня набуває популярності та додає нових користувачів. Це впливає на те, що кількість даних, які курсують інтернетом також постійно росте. Доказом цього є дослідження транснаціональної корпорації Cisco, яке вказує на те, що щоденний обсяг інтернет-трафіку за останні 10 років збільшився з десятків до сотень ексабайт переданих даних.

Такий приріст оброблюваних даних пояснюється декількома факторами:

– постійне зростання кількості користувачів;
– підвищення доступності мобільного доступу до інтернету;
– зростання популярності відео-контенту;
– підвищення якості мультимедійного контенту;
– збільшення використання соціальних мереж;
– підвищення популярності хмарних сервісів.

Проте, одночасно із ростом трафіку та кількості користувачів, в інтернеті також з’являється більше зловмисників, завданням яких є викрадення персональних даних як звичайних користувачів, так і корпорацій. Саме тому, за статистикою [8] кількість кібератак за останні 10 років також зросла, пропорційно до зростання кількості користувачів та трафіку в мережі Інтернет.

Сфера впливу кібератак в інтернеті стосується усіх користувачів, оскільки кібератаки бувають різних типів та різної спрямованості [9]. Серед найпоширеніших видів кібератак є такі:

– фішинг;
– міжсайтові сценарії;
– зловмисні програми з вимогою викупу;
– віддалені мережеві атаки
– розподілені атаки на відмову в обслуговуванні (або DDoS-атаки);
– SQL-ін’єкції.

Кожен із перерахованих типів кібератак або обов’язково вимагає наявність інтернет-з’єднання у користувача, або може бути здійснений, в тому числі, через мережу Інтернет.

Одним з найнебезпечніших видів кібератак є віддалена мережева атака. Віддалена мережева атака – це вид кібератаки, який здійснює руйнівний вплив на певні системи та дані та здійснюється по каналах зв’язку.

Оскільки будь-який пристрій, який підключений до мережі Інтернет, має власну IP-адресу, це означає, що до цього пристрою можна підключитися або обмінятися з ним даними віддалено. Безпосередня передача даних між пристроями в мережі Інтернет відбувається шляхом передачі окремих пакетів з даними, використовуючи протокол передачі даних TCP/IP. Для передачі цих пакетів з даними на розподілених системах, серед яких і мобільні пристрої, і персональні комп’ютери, використовуються відкриті канали передачі даних, доступ до яких як ззовні, так і зсередини, мають доступ інші розподілені системи мережі Інтернет.

Отже, даний тип атак працює таким чином, що зловживає відкритістю розподілених систем під час обміну даними та самими даними. Це дозволяє зловмисникам не лише постійно прослуховувати усю інформацію, якою обмінюється користувач, але і відловлювати, модифікувати та викрадати передаваний трафік.

В свою чергу, за характером впливу віддалені мережеві атаки поділяються на два основні типи:

– атаки, які здійснюють пасивний вплив на систему;
– атаки, які здійснюють активний вплив на систему.

Ціллю атак, які здійснюють активний вплив на систему є пряме втручання в роботу системи, таким чином, це робить простішим виявлення даних атак та внесення

ними змін в роботу системи, оскільки цей активний вплив не залишається непомітним в системі.

Натомість, ціллю атак, які здійснюють пасивний вплив на систему, є мінімальне внесення змін в систему та її роботу, щоб дана атака та внесені нею зміни не були виявлені і зловмисники могли продовжувати отримувати необхідні їм дані з ураженої системи користувача.

Пасивні атаки несуть в собі особливу небезпеку, оскільки з їх допомогою можуть бути атаковані сервери, які містять чутливі дані, що можуть бути пов’язані з персональними даними користувачів або навіть з даними банківських систем чи державних органів. Таким чином, зловмисники можуть отримати доступ до секретних даних або даних, за допомогою яких зможуть незаконно заволодіти фінансами.

Дані атаки не є рідкістю [10]. Для прикладу, в 2016 році відбулася кібератака, яка несе назву «Хакерська крадіжка золотовалютних резервів Бангладеша». Під час даної атаки було проведено спробу викрасти майже 1 мільярд доларів з рахунку Банку Бангладеш у Федеральному резервному банку Нью-Йорка. В результаті атаки зловмисникам вдалося викрасти 101 мільйон доларів.

Серед відомих віддалених мережевих атак на персональні дані атака, проведена на території Німеччини. В результаті проведеної атаки було поширено архіви з документами та електронними листами сотень депутатів Бундестагу та Європейського парламенту. Також, серед них були листи з поштових скриньок Ангели Меркель та Франка-Вальтера Штайнмаєра.

Для сучасної України питання безпеки передачі даних в інтернеті набуло ще більшої важливості ніж раніше, у зв’язку із повномасштабним вторгненням Російської Федерації. Вторгнення ворога також супроводжувалося кібератаками різних типів [11] зі спробами викрадень персональних даних військових, посадовців, блогерів, журналістів та інших верств населення, а також зі спробами викрадення даних, що відносяться до державної таємниці.

Згідно звіту про роботу системи виявлення вразливостей і реагування на кіберінциденти та кібератаки [12], підготованого Оперативним центром реагування на кіберінциденти, за 2022 рік було зареєстровано 415 критичних кіберінцидентів, що

є в 2.8 більшим показником у порівнянні з 2021 роком. З переліку всіх зареєстрованих критичних інцидентів близько 20% підпадають під категорію таких, які можна вважати віддаленими мережевими атаками, а самих атак даного типу відбулося в 2.2 рази більше, аніж за попередній звітний період.

За даними Держспецзв'язку [13], кількість кібератак на об'єкти критичної інформаційної інфраструктури та державні інформаційні системи зросла втричі, порівняно із кількістю кібератак, виявлених до повномасштабного вторгнення. За отриманою інформацією, 90% таких атак здійснюють саме російські та білоруські хакери, чия діяльність фінансується державою.

Як повідомляють у відомстві, основною метою таких атак з боку ворога є об'єкти цивільної інфраструктури [14]. Ці ворожі дії змусили багато установ перенести свої дані до більш безпечних сховищ. Серед таких сховищ, зокрема, хмарні сховища як на території України, так і закордоном.

Вдале проведення таких кібератак може призвести до витоку персональних даних, що в свою чергу загрожує тим, що ворожі спецслужби або військові зможуть в подальшому використовувати ці дані проти українців як на підконтрольній державним органам самоврядування території, так і на непідконтрольній території, що несе навіть більшу небезпеку, оскільки на цій території українці є найбільш вразливими перед ворогом. Також, певні чутливі дані, отримані під час кібератак, можуть бути використані для проведення подальших атак на системи, що пов'язані з роботою органів влади чи критичної інфраструктури.

Саме тому відомством наголошується на важливості дотримання вимог, визначених у Законі України «Про захист інформації в інформаційно-комунікаційних системах» під час дії воєнного стану, оскільки всю відповідальність за виконання цих вимог несе безпосередньо власник системи.

Отже, аналіз даної предметної області показав, на скільки значною є проблема загрози безпеці передачі даних мережею Інтернет та на скільки важливим є вирішення цієї проблеми, шляхом підвищення безпеки передачі даних.

1.2 Порівняльний аналіз переваг та недоліків існуючих рішень

У сфері підвищення безпеки передачі даних в мережі Інтернет існує безліч розроблених рішень на різних архітектурних рівнях. Проте, жодне з наявних рішень не є ідеальним як в плані безпеки, так і в плані швидкодії чи зручності. При розгляді та аналізі існуючих рішень необхідно зазначити, що існує два основних рівня впровадження безпеки передачі даних, які потрібно відокремити. Цими рівнями є рівень протоколу передачі даних та рівень шифрування безпосереднього наповнення процесу передачі даних.

Оскільки у цій ієрархії серед двох об’єктів – протоколу передачі даних та шифрування даних – нижчим рівнем є саме протокол передачі даних, тому розпочнемо аналіз переваг та недоліків уже наявних рішень з існуючих протоколів передачі даних.

Протокол передачі даних – набір правил або угод, які знаходяться на логічному рівні та визначають перебіг процесу обміну даними між програмами та системами. Протоколи передачі даних бувають різних типів та мають складну багаторівневу архітектуру, таким чином забезпечуючи передачу даних як між програмами всередині однієї системи, так і обміном даними між різними системами в спільній локальній мережі або в більш глобальній мережі, що складається з менших, локальніших мереж.

Будь-які протоколи передачі даних розробляються слідуючи певним узгодженим правилам [15]. Таким набором правил є OSI. OSI (Open Systems Interconnection) – модель взаємодії відкритих систем, яка складається з семи основних рівнів, а саме:

- фізичний;
- канальний;
- мережевий;
- транспортний;
- сеансовий;
- рівень представлення;
- прикладний.

Для кожного з рівнів цієї архітектури існують наявні окремі протоколи передачі даних, які використовуються розробниками програм чи систем, залежно від поставлених задач. Розглянемо кожен з цих семи архітектурних рівнів детальніше.

Фізичний рівень – перший і найнижчий рівень. На даному рівні відбувається передача даних у фізичному вигляді, тобто у вигляді оптичних і електричних радіохвиль чи імпульсів. Серед найпоширеніших протоколів на цьому рівні є медіаконвертери, хаби та сигнальні ретранслятори.

Канальний рівень – другий архітектурний рівень, завданням якого є забезпечення передачі даних між пов’язаними між собою вузлами. Також серед завдань протоколів цього рівня є виправлення помилок, які виникають на нижчому, фізичному рівні. Канальний рівень містить у собі два додаткових підрівня: LLC (Logical Link Control) та MAC (Media Access Control).

Мережевий рівень – третій рівень архітектури, за допомогою якого виконується безпосередня передача пакетів даних та проводиться оптимізація маршрутизації передачі цих пакетів мережею. На даному рівні застосовується маршрутизація через мережевий протокол IP.

Транспортний рівень – четвертий рівень в архітектурі OSI, який відповідає за забезпечення передачі пакетів даних між кінцевими адресатами, тобто системами та хостами. Саме на цьому рівні протоколи визначають напрямок, швидкість, обсяг передавання пакетів даних та інші важливі параметри, необхідні для коректної доставки даних в кінцеву точку. Найпопулярнішими протоколами даного рівня зараз є транспортні протоколи передачі даних TCP та UDP.

Сеансовий рівень – п’ятий архітектурний рівень, який був спроєктований з метою керування взаємодією двох пристроїв у мережі, для чого створюється окремий сеанс зв’язку. До функцій даного архітектурного рівня входить відповідальність за встановлення з’єднання між двома системами мережі, подальше координування цього з’єднання та пакетів даних, які будуть передаватися з використанням цього з’єднання, а також, в кінцевому результаті, припинення зв’язку та взаємодії між програмами чи системами, які використовували це з’єднання. Прикладами таких протоколів є протоколи X.235, ZIP та SCP.

Рівень представлення – шостий та передостанній рівень архітектури, представлення даних якого є незалежним від найвищого, прикладного рівня. Завданням протоколів даного рівня є здійснення перетворення передаваних даних з формату програми в мережевий формат і навпаки. Переважно такі перетворення виражаються у шифруванні та дешифруванні даних для більш безпечного передавання. Найвідомішим протоколом, який використовується на даному рівні є протокол SSL.

Прикладний рівень – сьомий та найвищий рівень архітектури OSI. Даний рівень є найближчим до користувача. Основним завданням протоколів даного рівня є отримання даних, які були передані від протоколів нижчого рівня, тобто від системи відправника та відображення отриманих даних кінцевому користувачу. На даному рівні використовується найбільше різноманіття протоколів. Серед найвідоміших протоколів даного рівня такі, як: FTP, BitTorrent, IMAP, HTTP, POP3, TELNET, SOAP та інші.

Оскільки кожен з рівнів архітектури має свої виразні особливості та власні протоколи передачі даних, проаналізуємо деякі протоколи більш детально. Для детальнішого аналізу було обрано такі протоколи, як TCP, UDP, SSL, FTP, HTTP. Дані протоколи є представниками трьох архітектурних рівнів системи OSI, зокрема протоколи TCP та UDP знаходяться на четвертому, транспортному, рівні. Протокол SSL знаходиться на шостому рівні, рівні представлення. Представниками сьомого, прикладного, рівня архітектури є протоколи FTP та HTTP. Більш детальний огляд обраних протоколів дозволить визначити, протоколи якого рівня є більш піддатливими для необхідних змін, а також допоможе обрати протокол, який найкраще підходитиме під поставлену задачу.

TCP – основоутворюючий протокол мережі Інтернет, який працює разом із протоколом мережевого рівня IP, за що протокол TCP також отримав назву TCP/IP. Даний протокол знаходиться на транспортному рівні архітектурної моделі OSI, таким чином протокол призначений для управління передачею та обміном даних між системами к мережах.

Протокол TCP/IP відноситься до класу протоколів зі встановленим з'єднанням. Протоколи цього класу відрізняються від інших протоколів тим, що вони намагаються встановити стабільний логічний зв'язок між тими хостами, між якими необхідно буде проводити обмін даними. Таким чином гарантується більш безпечна та надійна доставка пакетів з даними від хоста-відправника до хоста-отримувача.

При відправці пакету даних з системи хоста-відправника дані з цього проходять через усі рівні архітектури OSI, починаючи з найвищого рівня і закінчуючи найнижчим рівнем. Таким чином, протокол TCP отримує потік даних, перетворений одним із протоколів рівня представлення. Для коректної роботи такого принципу обробки даних кожному протоколу рівня представлення виділяється окремий TCP-порт, для того щоб протокол міг розуміти, який протокол звертається до нього та яким чином обробляти ці дані.

UDP (User Datagram Protocol) – протокол, який знаходиться на транспортному рівні архітектурної моделі OSI, так само як і протокол TCP. Проте, між цими протоколами є значні відмінності. Хоча протокол UDP підпадає під мережеву модель (стек) TCP/IP, але вважається одним з найпростіших протоколів цього стеку. В першу чергу це пов'язано з тим, що даний протокол виконує передачу даних без попереднього встановлення стабільного та надійного з'єднання між хостом-відправником та хостом-отримувачем. Також, даний протокол є простішим за інші протоколи даного архітектурного рівня моделі OSI, оскільки його концепція передачі даних відрізняється від інших. Протокол передбачає відправлення повідомлень, або так званих «датаграм», без підтвердження та гарантії доставки. Таким чином, він знімає з себе відповідальність за коректність та цілісність доставлених кінцевому користувачу даних, перекладаючи цю відповідальність на протоколи вищого рівня. Такі відмінності від інших протоколів є невипадковими, оскільки такий підхід оптимізує обмін даними, які не є чутливими, обсяг яких є не значним та у випадку, коли частота передачі даних є високою. Переважно даний протокол використовується на серверах, який надсилає невеликі повідомлення великій кількості клієнтів, а також для передачі у мережі потокового відео, аудіо. Для взаємодії з даним протоколом

використовуються такі протоколи вищого рівня, як TFTP, SNMP, DHCP, а також служба доменних імен DNS.

FTP (File Transfer Protocol) – протокол прикладного рівня, призначенням якого є передача файлів у мережі. Цей протокол використовується у клієнт-серверних мережах. Це означає, що файли, які передаються цим протоколом, не можуть бути передані напряму з системи одного клієнта в систему іншого клієнта, а лише через посередника – сервер – який займатиметься маршрутизацією та кешуванням, або зберіганням, цих файлів, а також підвищує безпеку під час передачі даних.

Обмін файлів між клієнтом та сервером відбувається шляхом створення у мережі між ними окремих каналів, по яких і проводиться обмін файлами та командами. Для підвищення безпеки передачі даних даний протокол має можливість автентифіковувати клієнтів на боці сервера за допомогою відкритого тексту, яким переважно є ім’я користувача та його пароль. Такий спосіб забезпечення передачі даних дозволяє перевіряти перед початком передачі даних, чи не була скомпрометована або викрадена система користувача. Також, є можливість налаштувати роботу протоколу без автентифікації, тоді дані будуть передаватися у режимі так званих «анонімних сеансів».

Протокол FTP має окремі типи даних для передаваних файлів, а саме:

– ASCII – для передачі файлів у форматі ASCII;
– EBCDIC – для передачі файлів у кодуванні EBCDIC;
– IMAGE – для передачі бінарних файлів, не вносячи змін до байтів файлу;
– Local – для передачі бінарних файлів, байти яких не є октетами.

Для додаткового підвищення захисту даних протокол FTP було розширено, додавши до нього функціонал протоколу SSH. Цей протокол має назву SFTP (SSH File Transfer Protocol).

Оскільки протокол FTP знаходиться на прикладному рівні, для коректної роботи та маршрутизації він використовує TCP-порти. Зокрема, даний протокол працює на 21-му TCP-порті для обміну командами, на 20-му TCP-порті для обміну даними, а також може динамічно використовувати додаткові порти з діапазону 49152-65534-их TCP-портів.

HTTP (HyperText Transfer Protocol) – один з найпопулярніших протоколів передачі даних прикладного рівня архітектурної моделі OSI. Даний протокол застосовується в комп'ютерних мережах для передачі гіпертекстових документів.

Основною задачею протоколу HTTP є передача даних різних форматів, типів, а також передача веб-сторінок. За допомогою протоколу HTTP можна передавати дані будь-яких типів, оскільки ці дані можна перевести й текстовий формат та відправити у HTTP-запиті. Таким, чином HTTP створює конкуренцію протоколу FTP, оскільки FTP є складнішим у застосуванні, аніж HTTP, і в той же час HTTP може цілком замінити функціонал протоколу FTP.

Використання даного протоколу передбачає, що клієнтським застосунком, тобто програмою, яка обробляє вхідні повідомлення, відповіді від сервера – це веб-браузер. Веб-браузери мають можливість відображати веб-сторінки, наповнення яких є відмінним від будь-якого іншого наповнення та дозволяє веб-браузерам точно ідентифікувати його. Оскільки HTTP може передавати не лише веб-сторінки, для ідентифікації іншого наповнення своїх повідомлень протокол дозволяє застосунку, який приймає ці відповіді, ідентифікувати це наповнення шляхом перевірки унікальних значень мови, кодування символів, типу наповнення та інших параметрів, які надає клієнту саме протокол HTTP.

У випадку, якщо необхідно отримати якісь дані від сервера (виконати запит), або надати відповідь клієнту, яка не є веб-сторінкою, цей запит або відповідь зобов'язаний буде мати певне значення одного з параметрів для ідентифікації клієнтом або сервером, а саме – метод запиту.

Існує дев'ять методів запиту, кожен з яких повинен оброблятися по-різному, залежно від значення цього метода. Основними з них є такі позначення методів, як GET, POST, PUT, DELETE. Ці позначення означають, що інформація, обмін якою проводиться між клієнтом та сервером в тому чи іншому напрямку, підлягає звичайному отриманню (для метода GET), створенню або створенню даних в системі (для метода POST), оновленню або зміні конкретних даних в системі (для метода PUT) або видаленню певних даних з системи (для метода DELETE).

Також, похідним протоколом від HTTP є протокол HTTPS. Він містить у собі усі принципи, які містить і протокол HTTP, проте має додаткову надбудову – а саме усі дані, які передає цей протокол, проходять через рівень представлення, таким чином підвищуючи безпеку передачі даних за допомогою використання протоколів SSL або TLS всередині.

Оскільки даний протокол знаходиться на прикладному рівні, який є вищим за рівень, на якому знаходиться протокол TCP, та використовується у моделі TCP/IP, для його функціонування він використовує у своїх звертаннях (запитах та відповідях) 80-ий TCP-порт. Похідний від нього протокол HTTPS використовує 443-ій TCP-порт.

SSL (Secure Sockets Layer) – протокол, який знаходиться на архітектурному рівні представлення, згідно моделі OSI. Даний протокол відноситься до протоколів криптографічного типу, оскільки основним його завданням є забезпечення встановлення безпечного з’єднання між системами у мережі, зокрема між вузлами клієнта та сервера.

Підвищення безпеки передачі даних відбувається завдяки тому, що протокол забезпечує анонімність передачі даних між системами у мережах, які базуються на моделі TCP/IP, шляхом шифрування даних за допомогою асиметричного алгоритму шифрування з відкритим ключем. Цей алгоритм шифрування передбачає використання двох ключів, кожен з яких може бути використаний для шифрування повідомлення. Відповідно, другий ключ буде використовуватися для дешифрування повідомлення. Такий підхід дозволяє тримати публічним один із ключів, в той час, коли інший ключ тримається в таємниці.

В результаті проведеного аналізу існуючих рішень у сфері мережевих протоколів даних, отримано достатню кількість даних для підведення коротких підсумків стосовно переваг та недоліків кожного із наявних рішень. Таким чином, коротка характеристика усіх проаналізованих протоколів передачі даних, у вигляді переваг та недоліків, подано у таблиці 1.1.

Таблиця 1.1 – Порівняння популярних мережевих протоколів передачі даних

Протокол	Переваги	Недоліки
TCP	– безпечне з'єднання між системами – стабільне з'єднання між системами – забезпечення цілісності даних – відсутність зайвого функціоналу	– важливість стабільного інтернет-з'єднання
UDP	– можливість доставляти дані без стабільного з'єднання між системами – простота передачі даних – оптимізація розміру даних	– низька безпека передачі даних – можливість втрати даних
FTP	– безпечне з'єднання між сервером та клієнтом – можливість автентифікації клієнта – вбудована можливість шифрування даних	– вузька спрямованість – низька універсальність
HTTP	– безпека передачі даних – висока універсальність – широка сфера застосування	– велика кількість потенційно зайвих надбудов
SSL	– висока безпека передачі даних – зручність у використанні	– не є самостійним повноцінним протоколом передачі даних

На вищому рівні за ієрархією двох об'єктів – протоколів передачі даних та шифрування даних – знаходиться шифрування даних. Хоча існують протоколи передачі даних, в які вже вбудовані певні алгоритми шифрування, але коли необхідно безпосередньо контролювати процес шифрування та дешифрування, корисно буде використовувати особисто обраний алгоритм (чи декілька алгоритмів) шифрування.

Шифрування є складним математичним процесом, який містить у собі різні математичні формули, вимагає серйозних обчислювальних потужностей [16]. Саме тому, для вивчення та покращення методів шифрування та захисту інформації в цілому існує окрема наука – криптографія.

Криптографія – наука, яка вивчає математичні засоби захисту інформації, її конфіденційності, а також цілісності. Криптографія була створена з метою захисту важливої інформації, яку потрібно було передати іншій особі, проте під час передачі даної інформації був великий ризик її втрати і перехоплення зловмисниками.

Вперше шифрування даних за допомогою механічних пристроїв було проведено у 1883 році. Класичними багато років вважалися алгоритми шифрування з симетричним ключем. Такі алгоритми шифрування передбачали, що для шифрування та дешифрування необхідно було використовувати один і той самий ключ.

Проте, починаючи з середини 1970-х років було започатковано та вперше використано новий спосіб шифрування даних, а саме асиметричне шифрування, тобто таке, застосування якого передбачало використання пар ключів, один з яких був відкритим, а інший – закритим, тобто приватним. Таким чином, кожен з цих ключів виконує свою роль: за допомогою відкритого ключа інформація шифрується, а за допомогою приватного ключа – розшифровується.

Для визначення, який зі способів шифрування є найоптимальніший, а також для визначення переваг та недоліків кожного зі способів, необхідно провести детальний аналіз кожного із варіантів існуючих рішень. Спершу буде проведено аналіз симетричного методу шифрування даних.

Симетричне шифрування – спосіб шифрування, який використовує єдиний ключ, яким обмінюються усі учасники обміну даних, та за допомогою якого шифрують та розшифровують передані дані усі учасники обміну даними [17].

Будь-яке шифрування, в тому числі і шифрування симетричним способом, в першу чергу забезпечує конфіденційність даних за допомогою двох складових:

– секретний ключ;
– шифр.

Секретний ключ – певне кодове слово, число, чи набір випадкових даних певної довжини, який відомий усім учасникам обміну даних та використовується для шифрування да розшифрування інформації. Секретний ключ повинен бути прихованим від сторонніх користувачів.

Шифр – алгоритм, який використовується для шифрування і дешифрування даних. Самі по собі алгоритми шифрування не є секретними та їх приховування не має сенсу. Важливим є приховування саме секретного ключа, за допомогою якого відбувається шифрування і розшифрування даних.

Симетричні алгоритми шифрування в свою чергу поділяються на два типи [18], а саме:

– потокові шифри;

– блочні шифри.

Потоковий шифр – тип симетричного шифру, який передбачає переведення даних в бітовий вигляд, після чого на цьому наборі бітів проводиться побітова операція XOR (виключна диз'юнкція), використовуючи ключ такої ж довжини, як і саме повідомлення.

Оскільки повідомлення може бути досить довгим, відповідно і ключ для такого повідомлення потрібен буде довгий. Зазвичай для того, щоб отримати такий довгий ключ використовується генератор псевдовипадкових чисел. Таким чином можна згенерувати і отримати ключ для шифрування повідомлень будь-якої довжини. Проте, цей ключ можна буде використовувати лише один раз, оскільки маючи один і той же ключ та два різні зашифровані повідомлення, зловмисники збільшують ймовірність розкриття цього ключа та в результаті зможуть розшифрувати абсолютно всі, зашифровані цим ключем, дані.

Серед найпопулярніших алгоритмів потокового шифрування такі алгоритми, як: A8, Pike, RC4, eSTREAM, SEAL.

Блочний шифр – це тип шифрування, який передбачає використання ключа фіксованої довжини та шифрує за допомогою нього повідомлення будь-якої довжини, циклічно шифруючи по частинах це повідомлення, використовуючи при кожному циклі новий, «раундовий ключ».

Проблема блочного шифрування в тому, що у випадку, якщо 16-байтова частина повідомлення, яка обробляється ключем відповідної довжини, повторюється, то шифротекст також повторюватиметься. Це означає, що зловмисники зможуть з легкістю розділити повідомлення, яке вийде в результаті шифрування на частини, а після цього – з легкістю визначити ключ, з яким було зашифроване ця невелика 16-байтова частина повідомлення, що дозволить визначити не лише «раундовий ключ», а й згодом – основний шифрувальний ключ.

Серед найпопулярніших алгоритмів блочного шифрування такі алгоритми, як: DES, AES, TripleDES.

Наступним видом шифрування є асиметричне шифрування, або шифрування алгоритмами шифрування з використанням відкритого та приватного ключа.

Асиметричне шифрування – система або підхід до криптографічного захисту даних, який ще називають криптосистемою з відкритим ключем. Цей вид криптосистем передбачає використання одного ключа для шифрування, а іншого – для розшифрування.

Відкритий ключ може бути опублікованим для усіх користувачів мережі, які виконують шифрування даних та подальший обмін цими даними. Це може бути виконано, оскільки відкритий ключ не може бути використаний для розшифрування даних. Натомість, для розшифрування даних необхідно мати персональний секретний ключ, або закритий ключ. Також, необхідно зазначити, що закритий ключ є незалежним від відкритого, тобто з відкритого ключа неможливо визначити закритий ключ, що дозволить вільно обмінюватися відкритим ключем.

Найважливішим відкриттям даного підходу до шифрування є те, що він надає можливість користувачам, що не обмінюються між собою заздалегідь закритими ключами, все одно обмінюватися повідомлення та розшифровувати за допомогою власних секретних ключів, передаючи лише відкриті ключі. Завдяки такому підходу до шифрування даних, більше немає необхідності попередньо затверджувати секретні ключі по окремих зашифрованих каналах, що також є небезпечним, оскільки цей ключ може бути перехоплений зловмисниками, а самі дані можуть бути в подальшому також викрадені та розшифровані.

Серед найпопулярніших алгоритмів шифрування є такі алгоритми, як алгоритм RSA, алгоритм Діффі-Хелмана, DSA та Схема Ель-Гамаля.

В результаті проведеного аналізу та отримавши достатню кількість інформації, це дає змогу підвести підсумки та сформувати наочну таблицю з перевагами та недоліками кожної з систем та підсистем, у випадку з симетричним шифруванням.

Коротка характеристика систем шифрування, у вигляді переваг та недоліків, подана у таблиці 1.2.

Таблиця 1.2 – Порівняння найпопулярніших систем шифрування

Система шифрування	Переваги	Недоліки
Симетричне (потокове)	–простота в реалізації	–потенційно довгий, складний ключ –відносно легкий у розшифруванні –висока ймовірність перехоплення ключа –висока ймовірність розшифрування зловмисниками
Симетричне (блочне)	–простота в реалізації –легкий ключ –необмежений розмір повідомлення відносно ключа	–важчий у розшифруванні за потоковий, проте все ще відносно легкий –висока ймовірність перехоплення ключа –висока ймовірність розшифрування зловмисниками
Асиметричне	–відсутність ризику перехоплення ключа –легкість у обміні зашифрованими даними з іншими користувачами –доступність у обміні даними, за допомогою відкритого ключа –високий рівень безпеки	–складніший у реалізації

1.3 Методологічні підходи до вирішення задачі за темою дослідження

Існує декілька методологій, які спрямовані на удосконалення безпеки передачі даних. Серед даних методологій є наступні:

– шифрування даних;

– мультифакторна аутентифікація;

– файєрволи;

– використання віртуальних приватних мереж.

Кожна з даних методологій є корисною для застосування під час передачі даних у мережі Інтернет. Розглянемо детальніше кожну з методологій.

Шифрування даних є найпоширенішою практикою, яка використовується для підвищення безпеки передачі даних. За статистикою, більше 90% даних в мережі Інтернет шифруються перед передачею. Шифрування є обов'язковим процесом при передачі даних, оскільки він значно підвищує час, необхідний зловмисникам для розшифрування та отримання доступу до передаваних даних. Для розшифрування даних, зашифрованих 128-бітним ключем, необхідно перебрати 3.4×10^{38} можливих ключів. Якщо для розшифрування такого ключа використовувати персональний комп'ютер середньостатистичного користувача, для розшифрування може знадобитися 2^{128}секунд. Проблемою будь-якого шифрування є те, що воно не може повністю забезпечити дані від зловмисників. Хоч зловмисникам і може знадобитися багато часу для розшифрування даних, проте це все ще залишається можливим.

Мультифакторна аутентифікація – методологія підвищення безпеки передачі даних, яка передбачає підтвердження особи отримувача даних для запевнення, що дані не отримають зловмисники. Зазвичай це підтвердження відбувається шляхом введення коду, який приходить в SMS-повідомленні чи на електронну пошту, секретним паролем чи іншими способами. Проте, такий спосіб ідентифікації отримувача не є ідеальним, оскільки зловмисники можуть перехопити таке повідомлення і провести верифікацію самостійно.

Файєрвол – методологія підвищення безпеки, яка передбачає встановлення спеціального програмного забезпечення, яке контролює вхідний та вихідний трафік пристрою. Це дозволяє керувати даними, які відправляються та надходять на пристрій, зокрема блокувати трафік, який є підозрілим, тобто таким який потенційно може бути спричинений зловмисниками і має за мету викрасти персональні дані користувача. Проте, дуже часто зловмисники маскують такий трафік під корисне програмне забезпечення, яке звичайному файєрволу складно відрізнити від справжнього корисного програмного забезпечення, що дозволяє обходити цей захист.

Використання віртуальних приватних мереж – методологія підвищення безпеки, яка створює безпечне зашифроване з'єднання між користувачем та іншою мережею, будучи посередником у цьому спілкуванні, та забезпечивши користувачу анонімність у мережі Інтернет. Проте, знаючи що користувач потенційно може користуватися

такою віртуальною мережею, зловмисники можуть спробувати дістатися до кінцевого користувача, що знівелює такий спосіб забезпечення.

Проаналізувавши найпоширеніші наявні методології підвищення безпеки передачі даних, можна зробити висновок, що жодна із наявних методологій не може повністю забезпечити користувача та його персональні дані. Проте, можна підвищувати безпеку, використовуючи якомога більше цих методологій, тим самим зменшуючи ймовірність попадання даних до зловмисників і їх подальшого розшифрування та використання у злочинних цілях.

1.4 Висновки. Постановка задачі.

Передача та обмін даними у всесвітній мережі Інтернет є однією з найважливіших частин у сфері інформаційних технологій. Саме тому, не менш важливим завданням є створення максимально безпечного середовища для передачі даних, оскільки мережею Інтернет користуються не лише звичайні користувачі, а й корпорації та державні установи, чиї дані є надзвичайно чутливими, і їх втрата чи перехоплення може коштувати мільярдів доларів.

Проте, навіть після усвідомлення цієї необхідності, створивши безліч протоколів передачі даних та описавши не один алгоритм для шифрування даних, статистика відображає, що кількість злочинів в інформаційній сфері не зменшується, а кількість викрадених даних та коштів значно росте з кожним роком.

Провівши аналіз предметної області та дослідивши найпопулярніші з наявних реалізацій протоколів передачі даних та алгоритмів шифрування, було виявлено, що жодне з рішень, або жодна з наявних комбінацій рішень даних рішень, має свої недоліки в контексті певних способів застосування. Наприклад, протокол передачі даних TCP є досить надійним та швидким, оскільки забезпечує стабільне та пряме з’єднання між вузлами системи. Проте, він не може захищати дані які передає самостійно, для нього необхідно реалізовувати надбудови у вигляді протоколів вищого рівня.

Актуальність роботи полягає у тому, що існуючі користувачі мережі Інтернет весь час знаходяться під ризиком викрадання їх даних. Саме тому необхідно постійно покращувати методи і алгоритми захисту від зловмисників та їх шкідливого програмного забезпечення.

Об’єкт дослідження – процес передачі даних у мережі Інтернет.

Предмет дослідження – методи, алгоритми та практики підвищення безпеки передачі даних.

Мета дослідження – удосконалення передачі даних, що понизить ймовірність перехоплення даних користувача під час їх передачі у мережі Інтернет, або понизить ймовірність їх розшифрування у разі перехоплення.

Відповідно до визначеної мети даного дослідження, можна виокремити наступні завдання дослідження:

– провести аналіз сфери безпеки передачі даних;
– дослідити сучасні методології та практики підвищення безпеки передачі даних;
– удосконалити наявні методології та практики для повного або часткового вирішення виявлених у них проблем;
– на основі створеного удосконаленого методу передачі даних виконати проєктування програмного забезпечення, котрий зможе реалізувати отриманий метод;
– провести програмну реалізацію спроєктованого програмного забезпечення;
– виконати тестування та перевірку отриманих результатів;
– проаналізувати результати проведеного дослідження та сформувати висновок щодо доцільності впровадження удосконаленого методу передачі даних.

Наступним кроком даної роботи є проведення аналізу існуючих концепцій та методів вирішення виявлених проблем, визначення того, який саме протокол передачі даних та алгоритм шифрування буде застосовано для вирішення поставленої задачі і вдосконалення безпеки передачі даних.

2 КОНЦЕПЦІЇ, МОДЕЛІ ТА МЕТОДИ ВИРІШЕННЯ ЗАДАЧІ

2.1 Концепції передачі даних у мережі Інтернет

Традиційна модель передачі даних у мережі інтернет складається із двох основних складових, які також одночасно можна вважати шарами цієї моделі:

– протокол передачі даних;

– алгоритм шифрування даних.

Відповідно кожна із цих складових традиційної моделі виконує різні функції. Зокрема, протокол передачі даних відповідає безпосередню передачу даних в мережі, а алгоритм шифрування в свою чергу відповідає за шифрування даних. Шифрування даних є обов’язковою складовою традиційної моделі передачі даних, оскільки без використання шифрування, ані передаватимуться мережею у відкритому вигляді, що значно полегшить роботу зловмисникам, яким для повного заволодіння інформацію достатньо буде лише перехопити повідомлення.

Проте така модель не є ідеальною та має значну кількість недоліків. Розглянемо детальніше обидві складові традиційної моделі передачі даних, дослідивши їх контекст, проблему та способи вирішення цих проблем.

Протокол передачі даних – набір інструкцій та алгоритмів, необхідних для того, щоб дані пройшли повний цикл від їх створення на апаратному рівні до отримання та відображення кінцевому користувачу крізь мережу Інтернет. Як було досліджено у попередньому розділі, протоколи бувають різних рівнів та можуть відповідати за різні етапи формування та передачі даних. Завдяки такій архітектурі передачі даних розробники мають можливість користуватися протоколами з будь-якого із рівнів, залежно від потреб. Зазвичай, коли мова йде про етап передачі даних в мережі Інтернет розробники розглядають протоколи найвищих рівнів архітектури. Такі протоколи можуть бути або виключно передавачами даних, тобто такими, задачею яких є виключно доставка даних кінцевому користувачу, або протоколами безпечної передачі даних, тобто такими, які окрім доставки даних імплементують певні засоби для більш безпечної передачі даних.

Контекст і проблематика

Вибір між протоколами передачі даних є не очевидним за рядом причин. При виборі транспортного протоколу передачі даних, завданням якого є виключно доставка даних в мережі, можна скористатися такими перевагами, як простота реалізації, вибір типу з’єднання або ж відсутність прямого з’єднання між абонентами при передачі даних, а також власний вибір алгоритму шифрування даних та в цілому власноручна розробка засобів захисту даних.

У той же час, розробники стикатимуться з такими проблемами протоколів даного рівня, як неможливість використання напрацювань уже наявних популярних протоколів високого рівня. Серед таких напрацювань, наприклад, готові механізми обробки різних типів даних. Це дозволяє спростити передачу даних та дозволяє не займатися імплементацією механізму розрізнення різних типів даних, їх обробки, конвертації та коректного відтворення кінцевому користувачу.

Також важливим є те, що серед транспортних протоколів передачі даних є велика кількість наявних протоколів, які теж можна обирати залежно від потреб розробника. Ці протоколи відрізняються в першу чергу за типом з’єднання між користувачами мережі та стабільністю цього з’єднання.

Проте, протоколи передачі даних транспортного рівня ніяк не шифрують дані, таким чином критично знижуючи безпеку передачі даних.

Рішення

Рішенням цієї дилеми є погляд на проблему в контексті більш глобальної проблеми, а саме проблеми безпеки передачі даних та не ідеальності наявних підходів та методів. Для розробки покращеного методу передачі даних необхідно використовувати існуючі напрацювання у сфері шифрування та безпеки даних, проте накладати їх варто на протоколи нижчих рівнів, оскільки вони не мають ніяких обмежень стосовно імплементації таких засобів, коли протоколи вищих рівнів будуть обмежувати розробників та змушуватимуть покращувати наявні технології виключно в рамках уже існуючих та ускладнюватимуть процес розробки, відлагодження і тестування покращеного протоколу передачі даних.

Протокол передачі даних, який необхідно використовувати як основу для розробки покращеного методу передачі даних повинен встановлювати пряме,

стабільне та надійне з’єднання між вузлами мережі, таким чином дозволяючи проводити більше однієї транзакції даних за сесію, не розриваючи її та контролюючи, чи не було з’єднання, або одного з користувачів, скомпрометовано та викрадено дані.

Також, для вирішення проблеми незахищеності передачі даних протоколами транспортного рівня, необхідно імплементувати один з алгоритмів шифрування.

Другою складовою традиційної моделі передачі даних у мережі Інтернет є алгоритми шифрування. Як було досліджено у попередньому розділі, алгоритми шифрування поділяються на два основні види за типом ключа, який використовується для шифрування даних: симетричні та асиметричні. Симетричні алгоритми шифрування мають перевагу у швидкості та простоті застосування, проте вони значно менш безпечні, оскільки на їх розшифрування, не маючи початкового ключа шифрування, піде значно менше часу, аніж на таке ж розшифрування даних, зашифрованих асиметричним алгоритмом шифрування.

Контекст і проблематика

Проблемою алгоритмів шифрування є також вибір між типами шифрування, а також вибір між конкретними реалізаціями одного з обраних типів алгоритмів шифрування даних.

Кожен з алгоритмів шифрування відрізняється від інших за складністю імплементації, складністю шифрування та розшифрування, а також швидкістю роботи.

Асиметричні алгоритми є складнішими в імплементації, оскільки для їх реалізації необхідно мати не лише власну пару ключів, а й публічний ключ іншого користувача. Також вони є повільнішими, оскільки як обмін ключами, так і саме шифрування або розшифрування, є складнішим та потребує більше часу на прорахування математичних функцій шифрування та розшифрування.

Проте, асиметричні алгоритми шифрування є значно безпечнішими за симетричні алгоритми шифрування за рахунок використання математичних функцій, які знижують ймовірність «відгадати» приватний ключ для розшифрування.

У той же час, проблемою будь-якого з алгоритмів шифрування є те, що жоден з алгоритмів шифрування не зможе захистити дані від розшифрування у випадку, якщо вони потраплять до зловмисників.

Рішення

Оскільки першочерговою проблемою є саме вразлива безпека передачі даних, необхідно в першу чергу звертати увагу на алгоритми шифрування, які можуть надати більш якісну безпеку для зашифрованих даних, не звертаючи увагу на швидкодію, складність та зручність таких алгоритмів.

І хоча безпека передачі даних підвищиться за рахунок шифрування одним із асиметричних алгоритмів шифрування, проте проблема захисту потрапляння даних до зловмисників залишається відкритою у традиційній моделі передачі даних у мережі Інтернет.

Хоча традиційна модель передачі даних все ще є досить популярною серед розробників, вона уже є застарілою. Більш новою є модель, в якій додається ще один шар, а саме шар аутентифікації користувача.

Така модель дозволяє набагато краще контролювати, чи не було скомпрометовано одного з учасників передачі даних. Способи верифікації користувачів є тісно пов’язаними з асиметричними алгоритмами шифрування, оскільки також використовують математичні функції та для функціонування потребують пари публічних і приватних ключів.

Контекст і проблематика

Аутентифікація та верифікація користувача, який є одним із вузлів мережі та ланцюга передачі даних, може значно підвищити безпеку передачі даних. Зазвичай верифікація користувача відбувається таким чином, що спершу відбувається окрема верифікація клієнта і уже потім дані відправляються цьому ж клієнту. Проте, такий спосіб верифікації не захищає дані від перехоплення зловмисниками під час передачі безпосередньо зашифрованих даних. Це дозволить зловмисникам перехопити дані та розшифрувати їх.

Також однією з проблем такого способу верифікації є те, що користувача, або сам клієнтський додаток отримувача, може бути підмінено в проміжку після верифікації користувача та перед початком процесу передачі даних. Таким чином, відправник буде впевнений, що передає дані верифікованому отримувачу, натомість відправляючи їх зловмиснику.

Ця проблема залишається невирішеною в даній моделі передачі даних. Саме тому можна зробити висновок, що доопрацювання саме цієї проблеми може значно підвищити безпеку передачі даних у мережі Інтернет та в цілому покращити існуючу модель передачі даних.

2.2 Моделі та методи покращення передачі даних

Проведемо більш детальний огляд описаний раніше моделей передачі даних. Розпочнемо з детального розгляду традиційної моделі.

На рисунку 2.1 зображено загальний вигляд традиційної моделі передачі даних.

Рисунок 2.1 – Загальний вигляд традиційної моделі

Як можна побачити із зображеного рисунку, традиційна модель передбачає використання трьох складових вузлів при передачі даних: два клієнти, між якими відбуватиметься безпосередня передача даних, та сервер, завданням якого є виключно маршрутизація.

Таким чином, із загального зображення традиційної моделі передачі даних можна зробити висновок, що традиційна модель є простою, не має багато складових, а сама передача даних є односторонньою.

Розглянемо детальніше процес передачі даних за допомогою цієї моделі. Процес передачі даних, передбачений традиційною моделлю зображено на рисунку 2.2.

Процес передачі даних за даною моделлю передбачає односторонню відправку зашифрованих даних від одного клієнта до іншого, використовуючи сервер як маршрутизатор запитів.

Рисунок 2.2 – Процес передачі даних традиційною моделлю

Проблемою даної моделі є відсутність верифікації кінцевого користувача, а саме отримувача, односторонність зв'язку, тобто відправник немає жодної інформації про те, куди, як і головне чи саме запланованому отримувачу надійшли дані, тобто чи не були вони перехоплені зловмисниками. Також важливою проблемою є монолітність відправлених даних, оскільки у разі перехоплення цих даних сторонніми особами, вони матимуть повний доступ до зашифрованих даних, що дасть можливість з часом розшифрувати їх та отримати повний доступ до уже розшифрованих даних, якими можуть бути персональні дані користувача або будь-які інші чутливі дані.

Виходячи з виявлених при дослідженні проблем, було покращено даний метод передачі даних у мережі Інтернет. Розглянемо покращений метод детальніше.

На рисунку 2.3 зображено загальний вигляд покращеної моделі передачі даних.

Рисунок 2.3 – Загальний вигляд покращеної моделі

На перший погляд може здатися, що різниця між загальним виглядом обох моделей є незначною, проте різниця між цими моделями є ключовою та дозволяє вирішити виявлені раніше проблеми. Покращена модель, так само як і традиційна, передбачає використання трьох основних вузлів мережі задіяних у процесі передачі даних: два клієнти, між якими відбуватиметься передача даних та сервер. Проте роль кожного з вузлів є значно ширшою, аніж у традиційній моделі. Відмінністю між

загальними традиційною та покращеною моделями є двосторонність цього зв’язку, що дозволятиме реалізувати вирішення для кожної з досліджених проблем.

Розглянемо покращену модель передачі даних більше детально, а саме процес передачі даних між вузлами мережі, їх взаємодію та усі етапи передачі даних.

На рисунку 2.4 зображено перший етап, необхідний для початку передачі даних отримувачу.

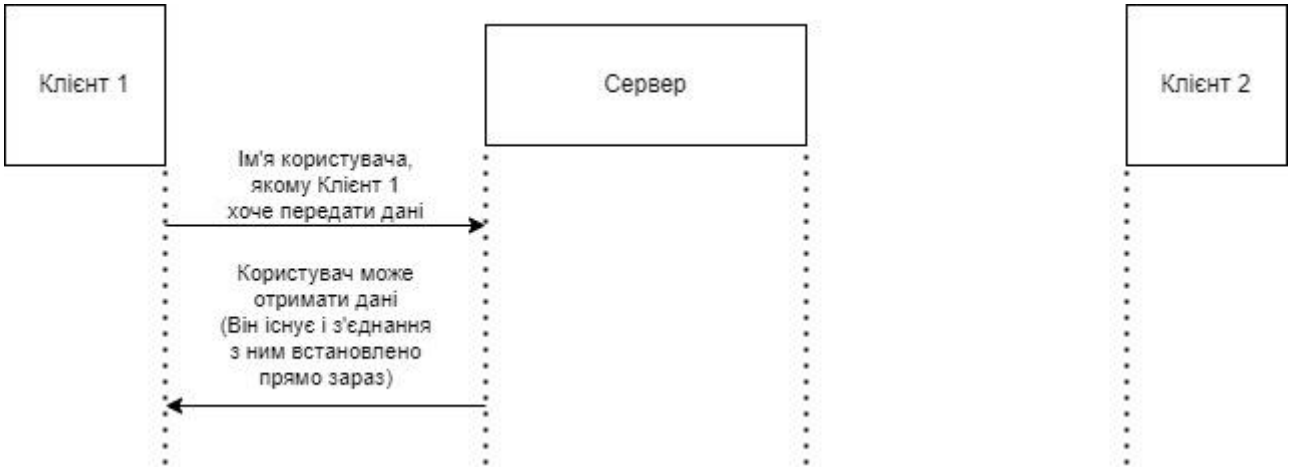


Рисунок 2.4 – Перший етап передачі даних покращеною моделлю

На зображеному рисунку першого етапу покращеної моделі можна спостерігати два кроки, які ініціює клієнт-відправник, та в якому задіяні клієнт-відправник та сервер. Дані кроки виконуються та є необхідними для аутентифікації відправника, оскільки при першій ініціалізації з’єднання клієнтом, сервер зберігає дані про сесію користувача, а саме прив’язку даних користувача, наприклад логін чи ім’я користувача, до даних про сам вузол мережі та його місцезнаходження. Такими даними може бути IP-адреса відправника, або ідентифікатор сокета, через який відбувається передача даних. Також даний етап дозволяє провести обов’язкову верифікацію отримувача перед передачею даних. Таким чином, відправнику у відповідь від сервера надійде інформація, чи є в даний момент можливість доставити дані отримувачу. Це вирішує проблему «сліпої» передачі даних у мережі, що значно зменшує ризик перехоплення цих даних під час цієї «сліпої» передачі, а також дозволить точно визначити, чи не скомпрометований отримувач, оскільки при вході

в мережу отримувач також проходить процес верифікації, щоб мати можливість для отримання даних, навіть якщо він не буде відправляти якісь дані.

На рисунку 2.5 зображено другий етап, необхідний для початку передачі даних.

Рисунок 2.5 – Другий етап передачі даних покращеною моделлю

На зображеному рисунку можна побачити задіяними усі три вузли мережі: як клієнтів, так і сервер. При чому саме на цьому етапі розкривається двосторонність зв’язку покращеної моделі та її важливість. Даний крок передбачає перший обмін даними між відправником та отримувачем. Перша передача даних не містить безпосередньо даних, які необхідно передавати отримувачу, проте містить деяку інформацію про дані, які буде доставлено отримувачу у випадку успішної верифікації даного користувача. Таким чином, дана передача даних містить інформацію про тип даних, які буде передано: зображення, аудіо, текст та інші, що дозволить отримувачу в подальшому розуміти, як обробляти отримані дані. Також перша передача даних містить випадково згенерований текст, так зване повідомлення, яке отримувачу потрібно буде підписати за допомогою власного приватного ключа. Саме цей крок є початком реалізації алгоритму верифікації повідомлень.

Перед попаданням до отримувача, ці дані проходять через сервер. Проте, на відміну від ролі сервера у традиційній моделі передачі даних, за покращеною моделлю роль сервера є значно ширшою. Сервер виконує не лише роль

маршрутизатора, але й аутентифікатора клієнтів. При кожному зверненні до сервера будь-якого з клієнтів, не залежно від того отримувачем даних є даний клієнт в цей момент, чи відправником, сервер проводить аутентифікацію даного клієнта за допомогою порівняння даних про сесію користувача. Якщо дані про сесію, які були збережені на сервері під час першого звертання клієнта, не збігаються із даними про сесію отриманими при поточному звертанні до сервера, іншому користувачу, який приймає участь в передачі даних, приходить сповіщення про те, що іншого користувача було скомпрометовано під час передачі даних, в звя’зку з чим передачу даних буде перервано.

У разі, якщо відправник пройшов дану аутентифікацію успішно, отримувач отримає дані про тип даних, які буде надіслано, та повідомлення, яке йому необхідно буде підписати. Підписане повідомлення даний користувач повертає відправнику, попередньо пройшовши аутентифікацію на сервері.

У разі, якщо отримувач пройшов аутентифікацію на сервері, відправник отримає підписане повідомлення від отримувача. Тепер відправник може також підписати дане повідомлення і звірити результат. Якщо результати не співпадають, це може сказати про те, що отримувача було скомпрометовано, оскільки це означатиме, що приватний ключ отримувача є невірним, що є неможливим при коректній роботі клієнта. Так повноцінно імплементується алгоритм верифікації повідомлень.

Таким чином, передача даних припиниться навіть не розпочавшись повноцінно. І навіть якщо отримувача було скомпрометовано, відправник дізнався про це вчасно і до зловмисника не потрапили жодні важливі дані, розшифрувавши які він отримає якусь користь.

На рисунку 2.6 зображено третій етап передачі даних за допомогою покращеної моделі передачі даних.

Третій етап передачі даних також передбачає задіяння усіх вузлів мережі. Саме на цьому етапі відбувається передача безпосередніх даних. Проте, на відміну від традиційної моделі, передача цих даних не є монолітною, тобто відбувається у кілька транзакцій.

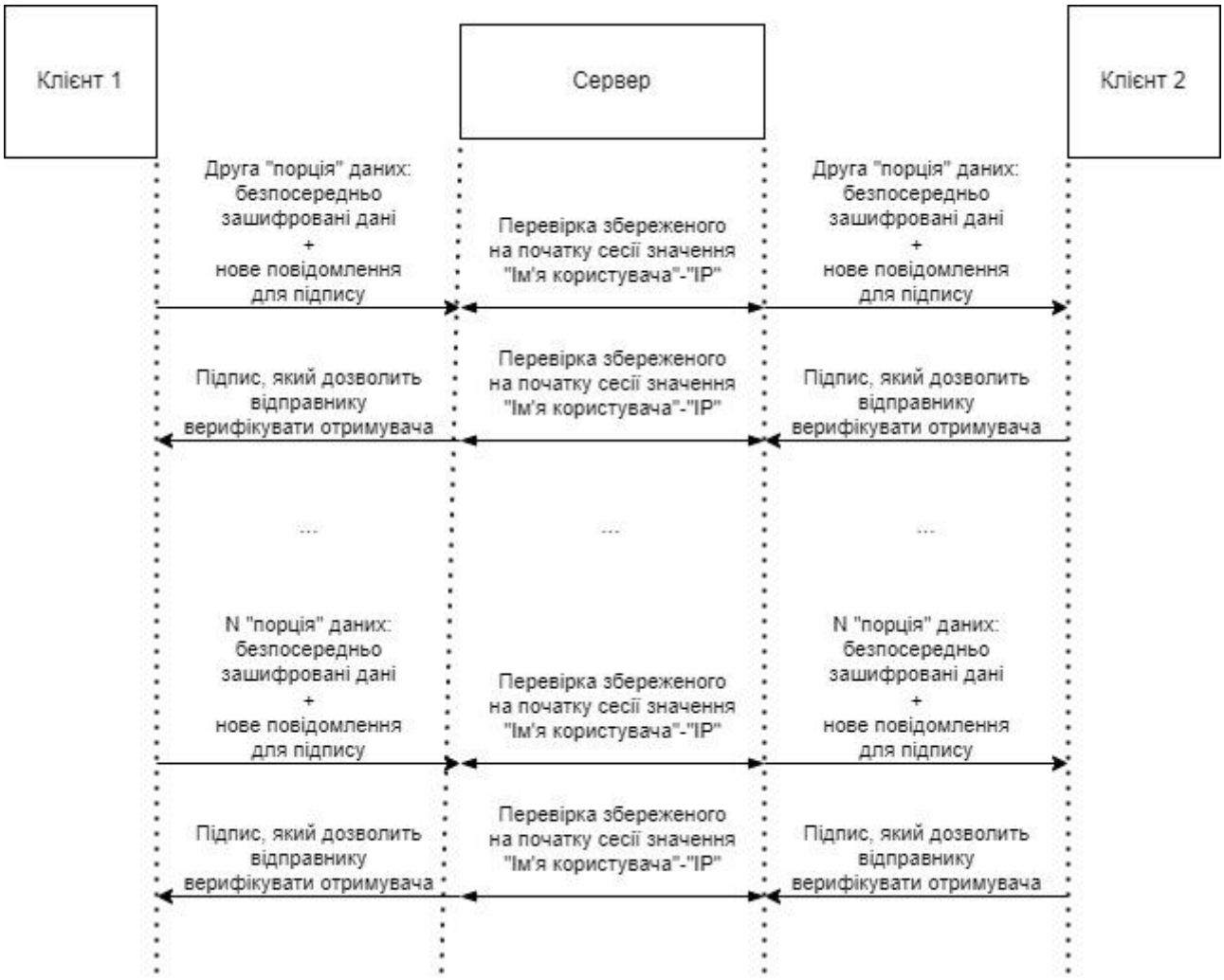


Рисунок 2.6 – Третій етап передачі даних покращеною моделлю

Це дозволяє зменшити ймовірність потрапляння до зловмисників повних даних, їх розшифрування та отримання доступу до розшифрованих даних користувача. Завдяки тому, що під час кожної з транзакцій виконується як аутентифікація обох користувачів на боці сервера, так і верифікація отримувача на боці відправника, це дозволяє якомога швидше відловити втручання зловмисника в процес передачі даних та одразу зупинити передачу даних, у разі виявлення такого стороннього втручання.

Таким чином, навіть у разі отримання зловмисником однієї чи кількох транзакцій даних, заволодіння ними, чи розшифрування, не дозволить отримати доступу до самих даних, оскільки дані можуть бути перемішаними між транзакціями, тобто транзакції, як знаходяться поруч, не дадуть змогу зібрати цілісну частинку повних даних.

На рисунку 2.7 зображено четвертий етап передачі даних, який дозволяє завершити передачу даних.

Рисунок 2.7 – Четвертий етап передачі даних покращеною моделлю

Четвертий етап передачі даних є завершальним. Даний етап передбачає відправлення повідомлення відправником отримувачу про те, що отримувачу було передано усі пакети з даними, що дозволить користувачу розпочати обробку отриманих даних та виведення цілісних даних кінцевому користувачу. На цьому етапі так само, як і на попередніх, передбачена аутентифікація сервером, проте аутентифікується лише один користувач, а саме відправник. Це відбувається тому, що немає потреби аутентифікувати отримувача, оскільки отримувач не повертатиме повідомлення у відповідь. Верифіковувати отримувача на даному етапі також не потрібно, оскільки навіть у випадку отримання повідомлення про завершення передачі даних зловмисником, це не принесе жодної шкоди ні відправнику, ні справжньому отримувачу.

Таким чином, запропонована покращена модель передачі даних мережею Інтернет вирішує усі виявлені при дослідженні проблеми та значно підвищує безпеку передачі даних.

2.3 Висновки

У ході написання даного розділу було здійснено дослідження наявних методів та моделей передачі даних у мережі Інтернет. Під час дослідження та аналізу наявних моделей було виділено основну традиційну методологію передачі даних та відносно нову, модернізовану, модель передачі даних.

Традиційна модель є простою, з трьома вузлами мережі, зв’язок між якими відбувається односторонньо. Такий підхід є легким в реалізації та впровадженні, не вимагає значних витрат у ресурсах систем як клієнтів, так і сервера, а також є досить швидким, оскільки передача даних відбувається за одну транзакцію, без потреби додаткових перевірок, верифікацій, автентифікацій чи розділення даних на окремі транзакції з подальшою перевіркою даних користувача.

Модернізована модель має деякі зміни відносно традиційної моделі, зокрема додається етап попередньої верифікації користувача. Така зміна позитивно впливає на безпеку передачі даних, оскільки перед тим, як передати кінцевому користувачу якісь дані, відправник зможе запевнитися в тому, що отримувача не було скомпрометовано та у тому, що дані буде доставлено і розшифровано коректно. Проте, проблемою безпеки даної моделі є те, що верифікація відбувається лише одного разу. Це означає, що під час процесу передачі даних зловмисник зможе втрутитись у цей процес шляхом компрометації отримувача і перехопити дані. І оскільки верифікацію не буде проведено по ходу процесу передачі даних, відправник не знатиме про те, що дані отримає зловмисник і ніяк не зможе завадити викраденню цих даних.

У контексті виявлених проблем було запропоновано покращену модель передачі даних у мережі Інтернет. Запропонована модель складається з такої ж кількості вузлів мережі, як і традиційна модель, тобто з двох користувачів та сервера. Проте покращена модель передбачає значно більше етапів та кроків для виконання передачі даних, а також дана модель передбачає двосторонність зв’язку, що може забезпечити додаткові перевірки, верифікації та аутентифікації учасників процесу передачі даних.

Запропонована покращена методологія передбачає поділення процесу передачі даних на чотири етапи. Таким чином, забезпечується значно краща безпека даних, неодноразова аутентифікація та верифікація обох кінцевих вузлів мережі, як відправника, так і отримувача.

Перший етап процесу передачі даних необхідний для запевнення відправником, що отримувач підключений до системи та має стабільне з’єднання для безпечної передачі даних.

Другий етап необхідний для того, щоб передати отримувачу інформацію про дані, які буде передано, щоб додаток-клієнт отримувача міг вірно обробляти ці дані, а також для проведення попередньої верифікації користувача з використанням алгоритму верифікації повідомлень.

На третьому етапі відбувається безпосередня передача зашифрованих даних у кілька транзакцій, що забезпечить дані від прямого попадання в руки зловмисників та заволодіння ними повних даних за один раз. Оскільки після кожної передачі даних відбувається верифікація та аутентифікація отримувача, втручання зловмисника буде виявлено та передача даних припиниться.

Останній етап передбачає відправлення повідомлення отримувачу про завершення передачі даних.

Отже, у цьому розділі запропоновано покращену модель передачі даних у мережі Інтернет, яка вирішує усі проблеми, виявлені на етапі дослідження існуючих моделей.

3 АЛГОРИТМИ ТА ТЕХНОЛОГІЇ ВИРІШЕННЯ ЗАДАЧІ

3.1 Алгоритми вирішення задачі

Для виконання поставленого на кваліфікаційну роботу завдання, а саме удосконалення передачі даних у мережі Інтернет, під час реалізації нового методу необхідно використовувати деякі існуючі алгоритми.

Оскільки на етапі дослідження було виокремлено три окремі рівні, або етапи, в архітектурі передачі даних, а саме:

– рівень верифікації;
– рівень шифрування даних;
– рівень передачі даних.

Відповідно, для кожного з даних рівнів буде застосовано існуючі алгоритми. Таким чином, для реалізації рівня верифікації буде імплементовано алгоритм верифікації повідомлень, який використовується для проведення повторюваної верифікації отримувача.

Для реалізації рівня шифрування даних буде імплементовано алгоритм шифрування Діффі-Хелмана. Даний алгоритм використовується для безпечного обміну публічними ключами та подальшого шифрування даних.

Для реалізації рівня передачі даних буде імплементовано протокол передачі даних TCP, для імплементації якого також потрібно притримуватись алгоритму передачі даних цього протоколу.

Розглянемо детальніше кожен із алгоритмів, обраних для реалізації відповідних шарів (етапів) передачі даних.

Алгоритм верифікації повідомлень – алгоритм, який використовується для проведення верифікації отримувача даних шляхом підпису отримувачем попередньо випадково згенерованого повідомлення. Схему алгоритму верифікації повідомлень зображено на рисунку 3.1.

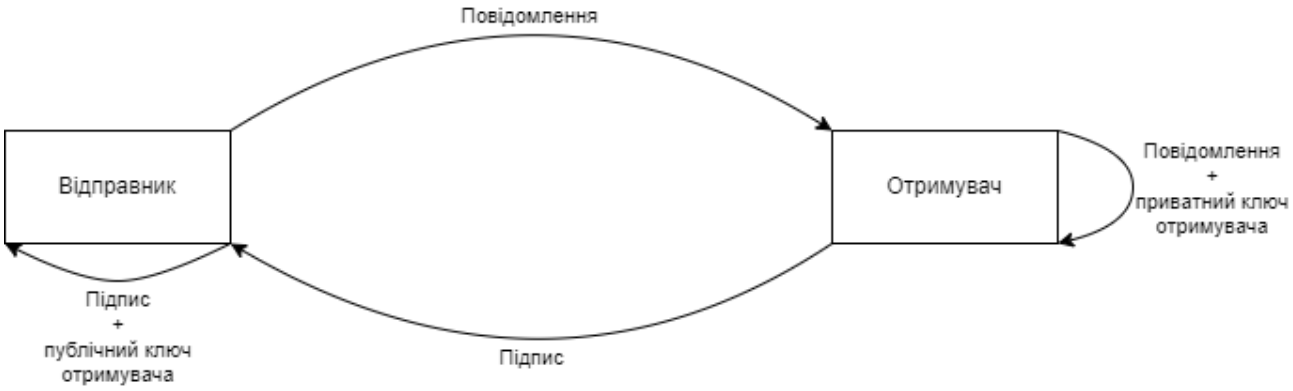


Рисунок 3.1 – Алгоритм верифікації повідомлень

Для початку процесу верифікації відправник генерує випадкове текстове повідомлення. Таким повідомленням може бути звичайний рядок із 64-ма чи 128-ма випадковими символами. Після генерації випадкового повідомлення, відправник надсилає його отримувачу. Після того, як отримувачу надійшло випадково згенероване повідомлення, він проводить «підписання» даного повідомлення, шляхом шифрування, або обрахування хешу даного повідомлення на основі приватного ключа отримувача. Після підписання повідомлення, отримувач повертає це повідомлення відправнику. Отримавши назад згенероване та підписане повідомлення, відправник проводить перевірку правильності підписання даного повідомлення за допомогою публічного ключа, наданого отримувачем. У разі, якщо перевірка коректності підписання повідомлення отримувачем пройшла успішно, процес передачі даних буде продовжено. Інакше, процес передачі даних буде зупинено та повідомлено відправника про те, що отримувача було скомпрометовано.

Наступним розглянемо алгоритм Діффі-Хелмана. Алгоритм Діффі-Хелмана – це алгоритм, який використовується для формування секретного ключа двома учасниками передачі даних, а також для подальшого шифрування даних. Отриманий секретний ключ можуть використовувати для шифрування обидва учасники процесу передачі даних. Схему алгоритму Діффі-Хелмана зображено на рисунку 3.2.

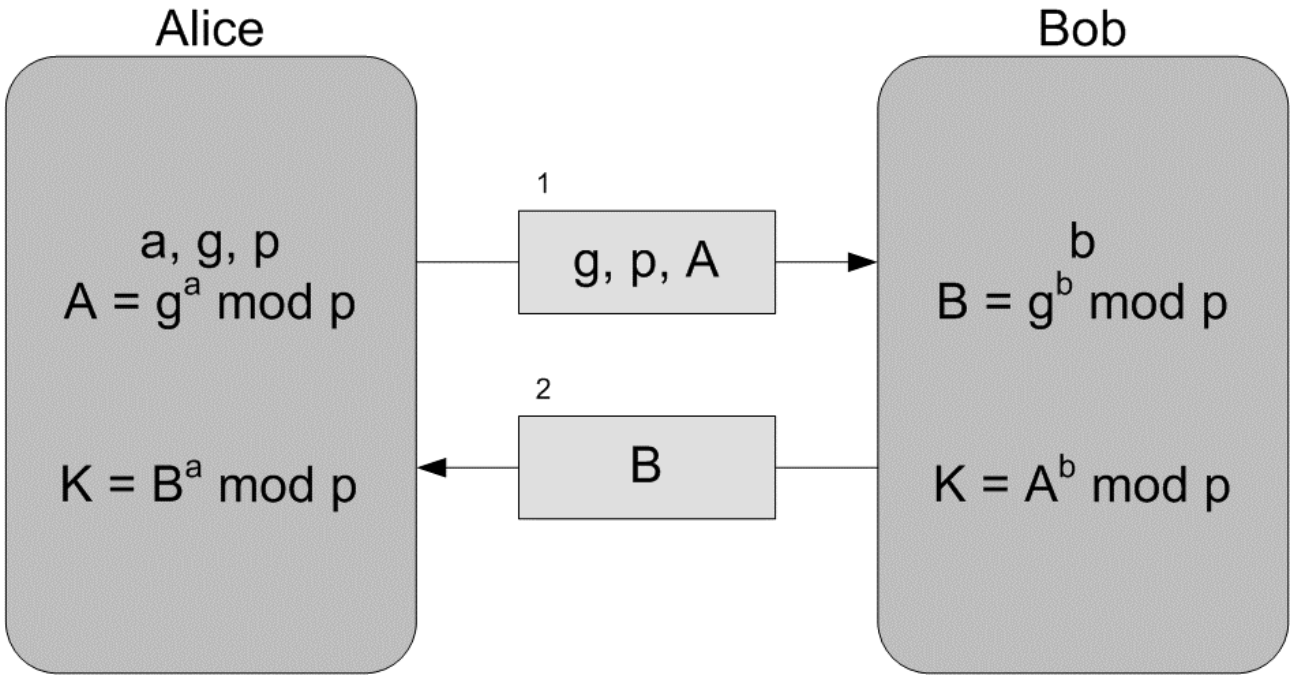


Рисунок 3.2 – Алгоритм Діффі-Хелмана

Згідно з зображеною схемою, p – це велике просте число, про яке знають обидва учасники передачі даних, яке може бути випадковим та змінюватися при кожному формуванні та обміні секретного ключа. Число g – також є числом, проте має знаходитись в межах від 1 до p. Це число може бути визначеним один раз, наприклад, в коді програми. Обидва цих числа можуть бути відкритими та відомими для будь-якого користувача. Число a – це число, згенероване першим учасником процесу передачі даних. Це число є необхідним для подальшого формування спільного секретного ключа, генерується при кожному формуванні секретного ключа та повинно знаходитись в межах від 1 до $p – 1$. Число b – таке ж число, як і число a, проте згенероване другим учасником процесу передачі даних.

Після формування усіх необхідних чисел, перший учасник формування та передачі секретного ключа проводить математичне обчислення за формулою $A = g^a mod_p$. Після отримання результату A, перший учасник відправляє це значення іншому учаснику разом зі значенням чисел g та p, який в свою чергу проводить аналогічне обчислення за формулою $B = g^b mod_p$. Після обчислення значення B, другий учасник повертає першому це значення, а також обчислює для себе секретний

ключ за формулою $K = A^b mod_p$. Після отримання першим учасником значення B, він також проводить обчислення секретного ключа за формулою $K = B^a mod_p$.

Таким чином, в результаті проведених операцій обидва користувачі матимуть секретний ключ для подальшого шифрування.

Наступним алгоритмом є алгоритм передачі даних за використання протоколу TCP. TCP є основним протоколом передачі даних транспортного рівня. Таким чином, алгоритм дій для імплементації даного алгоритму є відносно простим, оскільки він не вимагає жодних додаткових дій з даними, окрім безпосередньої їх передачі. Схему алгоритму передачі даних протоколом TCP зображено на рисунку 3.3.

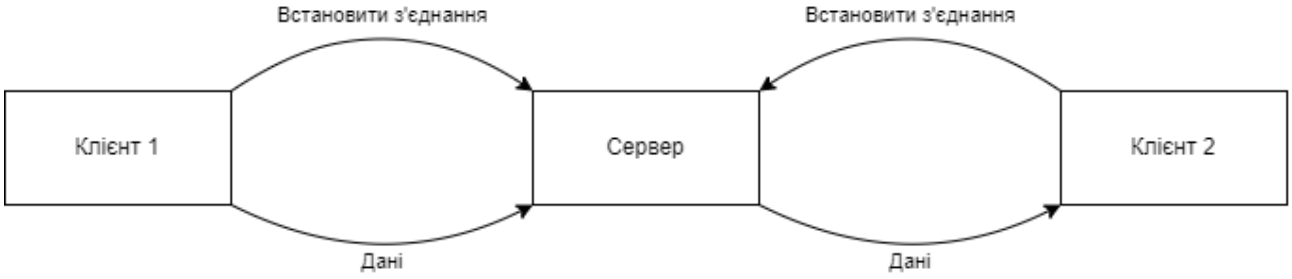


Рисунок 3.3 – Алгоритм передачі даних протоколом TCP

Для передачі даних протоколом TCP спочатку необхідно встановити стабільне з’єднання між клієнтським додатком та сервером. Також, для успішної передачі даних, необхідно щоб отримувач також в цей момент мав наявне стабільне з’єднання з сервером. З’єднання між клієнтським додатком та сервером встановлюється шляхом відкриття та прослуховування спеціальних сокетів на портах, визначених програмним кодом клієнтського додатку та сервера.

Після встановлення з’єднання відправником та отримувачем, відбувається відправка даних від одного користувача до іншого. Ці дані проходять через сервер, який виступає в ролі маршрутизатора, оскільки саме на сервері зберігаються дані про те, на якому саме сокеті знаходиться користувач, якому намагаються відправити дані.

Після доставки даних, стандартним алгоритмом передачі даних протоколом TCP не передбачене повернення відправнику жодного повідомлення про успішність чи не успішність передачі даних.

3.2 Визначення вимог до програмної системи

В результаті проведеного аналізу та дослідження предметної області, аналізу наявних рішень, алгоритмів та досліджень, отримано достатню інформацію для формування та опису вимог до програмної системи.

Для опису мети розробки використовується формування бізнес-вимог програмної системи. Метою розроблюваної системи є реалізація удосконаленого методу передачі даних у мережі Інтернет з використанням алгоритму верифікації повідомлень та алгоритму Діффі-Хелмана для підвищення безпеки передачі даних та надання можливості постійної та повторної верифікації та автентифікації отримувача при відправці даних.

Розроблювану програмну систему необхідно забезпечити усім необхідним функціоналом для коректного та повноцінного користування.

Для проєктування та планування функціоналу необхідно визначити та сформулювати вимоги користувачів системи. В розроблюваній системі існує дві ролі кінцевих користувачів:

– користувач;
– адміністратор.

Оскільки основним функціоналом розроблюваної системи буде надсилання даних різних типів іншим користувачам, необхідно реалізувати достатній функціонал для реєстрації, авторизації та безпосереднього користування функціями передачі даних, щоб вони були зрозумілими та максимально простими для кінцевого користувача. Також, від кінцевого користувача потрібно приховати такі деталі, необхідні для коректної роботи системи, як генерація та передача ключів, вибір порту для прослуховування та відправки даних, а також спростити відображення процесу передачі даних і організувати зручне та зрозуміле виведення тексту про помилки та проблеми, які виникатимуть під час процесу передачі даних або в результаті користування будь-якими іншими функціями програмної системи.

Адміністратору програмної системи повинен бути наданий доступ до серверу системи, а саме – до бази даних користувачів. Усі чутливі дані, такі як логіни та паролі

зберігатимуться у вигляді хешу, тому залишатимуться в безпеці. Проте, цей доступ необхідний для адміністратора системи у зв'язку з високою ймовірністю потреби втручання та модерації саме користувачів системи. Наприклад, це може бути пов'язано з виявленням частого зловживання доступом до системи. Також втручання адміністратора може бути корисним у випадку, коли одного й того ж користувача намагалися скомпрометувати кілька разів. Тоді адміністратор може допомогти у виявленні зловмисника, шляхом відслідковування IP-адреси, з якої зловмисник намагався перехопити передачу даних та, таким чином, незаконно отримати доступ та заволодіти даними.

Опишемо акторів програмної системи, базуючись на сформованих вимогах. Опис акторів системи наведено у таблиці 3.1.

Таблиця 3.1 – Опис акторів системи

Актор	Короткий опис
Користувач	–здійснює реєстрацію; –здійснює авторизацію; –здійснює передачу даних; –здійснює отримання даних; –переглядає власні персональні дані; –вносить зміни до персональних даних; –змінює налаштування передачі даних; –здійснює пошук іншого користувача.
Адміністратор	–здійснює перегляд наявних користувачів системи; –здійснює перегляд журналу помилок системи; –здійснює перегляд журналу підозрілих дій та втручань в процес передачі даних; –здійснює перегляд журналу скарг на користувачів; –здійснює блокування користувачів.

На основі наявних вимог, описаних акторів системи та зважаючи на додаткові вимоги до функціональності програмної системи, сформуємо більш детальний опис вимог та виділимо основні варіанти використання програмної системи.

Діаграму варіантів використання наведено на рисунку А.1 (додаток А).

Опис варіантів використання програмної системи наведено у таблиці 3.2.

Таблиця 3.2 – Опис варіантів використання системи

Актор	Найменування ВВ	Опис ВВ
1	2	3
Незареєстрований користувач	Реєстрація	Створення нового облікового запису
Зареєстрований користувач	Авторизація	Вхід в існуючий обліковий запис
	Перегляд наявних контактів	Перегляд списку наявних контактів, які було додано в цей список, або з якими було ініційовано проведення процесу передачі або отримання даних
	Пошук іншого користувача	Проведення пошуку користувача за його персональними даними, а саме за ім’ям користувача або унікальним ідентифікаційним кодом
	Ініціювання передачі даних	Відправка запиту іншому користувачу на початок передачі йому даних
	Прийом запиту на передачу даних	Отримання запиту на отримання даних від іншого користувача та підтвердження такого запиту
	Відмова від запиту на передачу даних	Отримання запиту на отримання даних від іншого користувача та відхилення такого запиту
	Перегляд персональних даних	Перегляд персональних даних користувача, таких як: логін, пароль, ім’я користувача, електронна пошта
	Зміна персональних даних	Зміна персональних даних користувача, таких як: логін, пароль, ім’я користувача, електронна пошта

Кінець таблиці 3.2

1	2	3
	Перегляд налаштувань передачі даних	Перегляд наявних налаштувань передачі даних, доступний як шаблон для усіх подальших передач даних та перед безпосередньою передачею даних. Серед налаштувань наявні такі опції, як: максимально допустимий розмір отримуваного файлу; максимальний розмір файлу, який відправляється; максимальна та мінімальна кількість транзакцій під час передачі даних
	Зміна налаштувань передачі даних	Зміна налаштувань передачі даних у шаблоні та безпосередньо перед початком передачі даних
Адміністратор	Перегляд користувачів	Перегляд усіх наявних користувачів системи
	Перегляд журналу помилок системи	Перегляд усіх зафіксованих помилок під час роботи системи
	Перегляд журналу підозрілих дій та втручань в процес передачі даних	Перегляд усіх зафіксованих підозрілих дій та втручань в процес передачі даних під час роботи системи
	Перегляд журналу скарг на користувачів	Перегляд усіх зафіксованих скарг на користувачів системи
	Блокування користувачів	Блокування користувачів системи, згідно зі скаргами від інших користувачів системи

Отже, в результаті проведеного дослідження предметної області передачі даних у мережі Інтернет було сформовано та описано вимоги до програмної системи. Після формування вимог до ПС необхідно провести проєктування системи.

3.3 Проєктування програмної системи

3.3.1 Розробка структури програмної системи

Розроблювана програмна система має бути повноцінним програмним засобом, котрий складатиметься з двох окремих частин: сервера та клієнта. Така архітектура програмних систем називається клієнт-серверною. Схему роботи даної архітектури зображено на рисунку 3.4.

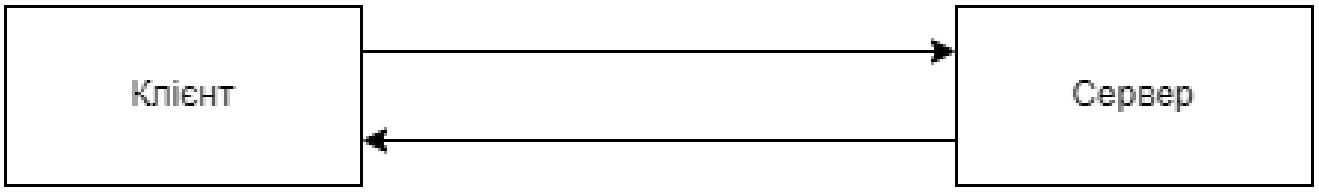


Рисунок 3.4 – Клієнт-серверна архітектура

Завдяки тому, що удосконалений метод передачі даних у мережі Інтернет передбачає використання протоколу TCP як спосіб транспорту даних, сервер зможе в будь-який момент викликати клієнтський додаток, так само як і клієнтський додаток в будь-який момент зможе звернутися до сервера.

В свою чергу, як клієнтська частина системи, так і серверна, поділяється на додаткові модулі. Клієнтський додаток містить три основні модулі, а саме:

– інтерфейс користувача;
– інтерфейс системи;
– модуль роботи з базою даних.

Модуль інтерфейсу користувача відповідає за отримання та відображення даних, отриманих від нижчих рівнів архітектури. Також, інтерфейс користувача виконує роль посередника між користувачем та всіма іншими модулями, оскільки саме цей модуль може передавати значення, які обирав, вводив чи змінював користувач у різні поля, перемикачі та інші елементи інтерфейсу.

Модуль інтерфейсу системи, або модуль бізнес-логіки, відповідає за формування даних отриманих з бази даних, сторонніх сервісів, або безпосередньо від користувача

через модуль інтерфейсу користувача, обробку цих даних згідно з необхідною бізнес-логікою системи та передачу цих даних у інші модулі системи.

Модуль роботи з базою даних містить усі необхідні сутності системи, які використовуються для роботи з базою даних та іншими модулями системи. Також, даний модуль відповідає за отримання, створення, зміну та видалення даних з БД.

Серверний додаток містить два основних модулі, а саме:

– інтерфейс системи;
– модуль роботи з базою даних.

Інтерфейс системи серверного додатку відповідає за усю бізнес-логіку сервера, зокрема за маршрутизацію користувачів, автентифікацію користувачів та іншу логіку програмної системи.

Модуль роботи з базою даних відповідає за збереження необхідної інформації про користувачів, транзакції даних та іншу інформацію, необхідну для адміністрування та відловлення зловмисних дій у системі.

На рисунку 3.5 зображено діаграму загального вигляду взаємодії компонентів програмної системи.

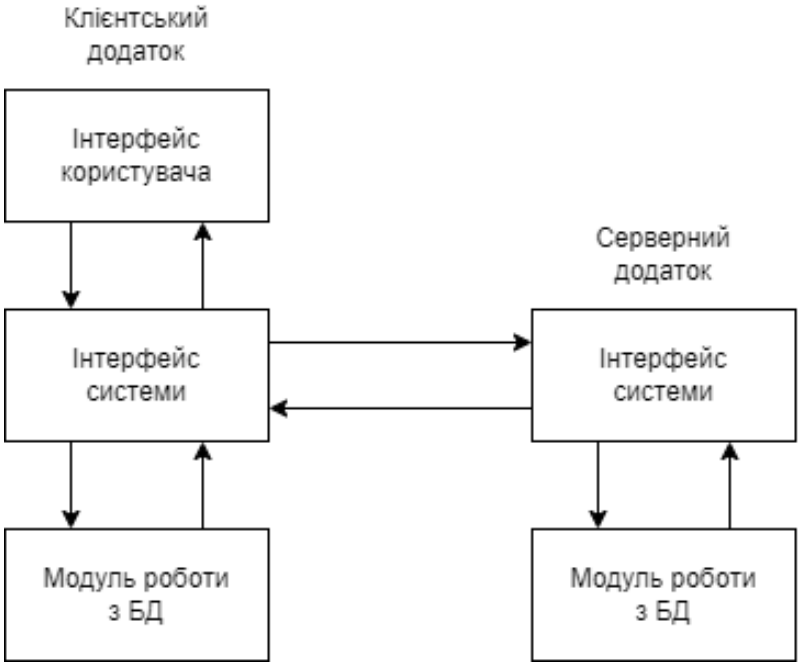


Рисунок 3.5 – Взаємодія компонентів ПС

Розглянемо детальніше складові кожного з додатків та їх модулів.

Розглянемо модулі та функції клієнтського додатку. Модуль інтерфейсу користувача має наступні під модулі:

– модуль відображення даних;
– модуль відловлення помилок;
– модуль обміну даними з рівнем бізнес-логіки.

Модуль відображення даних відповідає за отримання та відображення даних з нижніх рівнів архітектури користувачу, а також за взаємодію з користувачем, шляхом отримання введених даних користувачем чи виклику різного функціоналу системи шляхом взаємодії з різноманітними елементами інтерфейсу.

Модуль відловлення помилок відповідає за перевірку отриманих даних з інших модулів системи, валідацію цих даних, фіксацію та виведення даних про проблеми та помилки у разі їх виникнення.

Модуль обміну даними з рівнем бізнес-логіки відповідає за коректну обробку даних, отриманих від користувача перед передачею цих даних на нижчі архітектурні рівні та за обробку даних, отриманих з нижчих архітектурних рівнів перед відображенням цих даних користувачу.

Модуль інтерфейсу системи містить наступні під модулі:

– модуль обміну даними з інтерфейсом користувача;
– модуль обробки бізнес-логіки;
– модуль обміну даними з іншим користувачем;
– модуль відловлення помилок;
– модуль обміну даними з сервером;
– модуль обміну даними з рівнем роботи з БД.

Модуль обміну даними з інтерфейсом користувача відповідає за обробку даних перед відправкою на рівень інтерфейсу користувача та за подальше отримання та обробку даних від цього рівня.

Модуль обробки бізнес-логіки клієнтського додатку відповідає за основну логіку системи, зокрема за шифрування даних, верифікацію і валідацію отриманих чи відправлених даних та інші.

Модуль обміну даними з іншим користувачем відповідає за транспортування та налаштування передачі даних з іншим користувачем через сервер, а також за очікування та отримання даних від іншого користувача через сервер.

Модуль відловлення помилок відповідає за перевірку отриманих даних з інших модулів системи, валідацію цих даних, фіксацію та виведення даних про проблеми та помилки у разі їх виникнення.

Модуль обміну даними з сервером відповідає за обробку та коректне перетворення даних, які відправляються на сервер, або отримуються від сервера.

Модуль обміну даними з рівнем роботи з БД відповідає за коректну обробку даних, які отримуються та надходять з бази даних, а також за їх перетворення в інші типи даних.

Модуль роботи з базою даних містить такі під модулі, як:

– модуль сутностей додатку;
– модуль відловлення помилок;
– модуль обміну даними з БД.

Модуль сутностей додатку містить в собі усі класи, типи та перечислення, які застосовуватимуться системою в процесі її розробки та функціонування.

Модуль відловлення помилок відповідає за перевірку отриманих даних з інших модулів системи, валідацію цих даних, фіксацію та виведення даних про проблеми та помилки у разі їх виникнення.

Модуль обміну даними з БД відповідає за перетворення та оперування даними, які записуватимуться чи діставатимуться безпосередньо з бази даних, зокрема за коректне створення, зміну даних, а також їх видалення.

Діаграму взаємодії компонентів і модулів клієнтського додатку зображено на рисунку 3.6.

Розглянемо детальніше модулі серверного додатку.

Модуль інтерфейсу системи серверного додатку містить наступні під модулі:

– модуль обміну даними з клієнтським додатком;
– модуль маршрутизації;
– модуль автентифікації користувачів;

– модуль роботи з даними користувачів;
– модуль відловлення помилок;
– модуль обміну даними з рівнем роботи з БД.

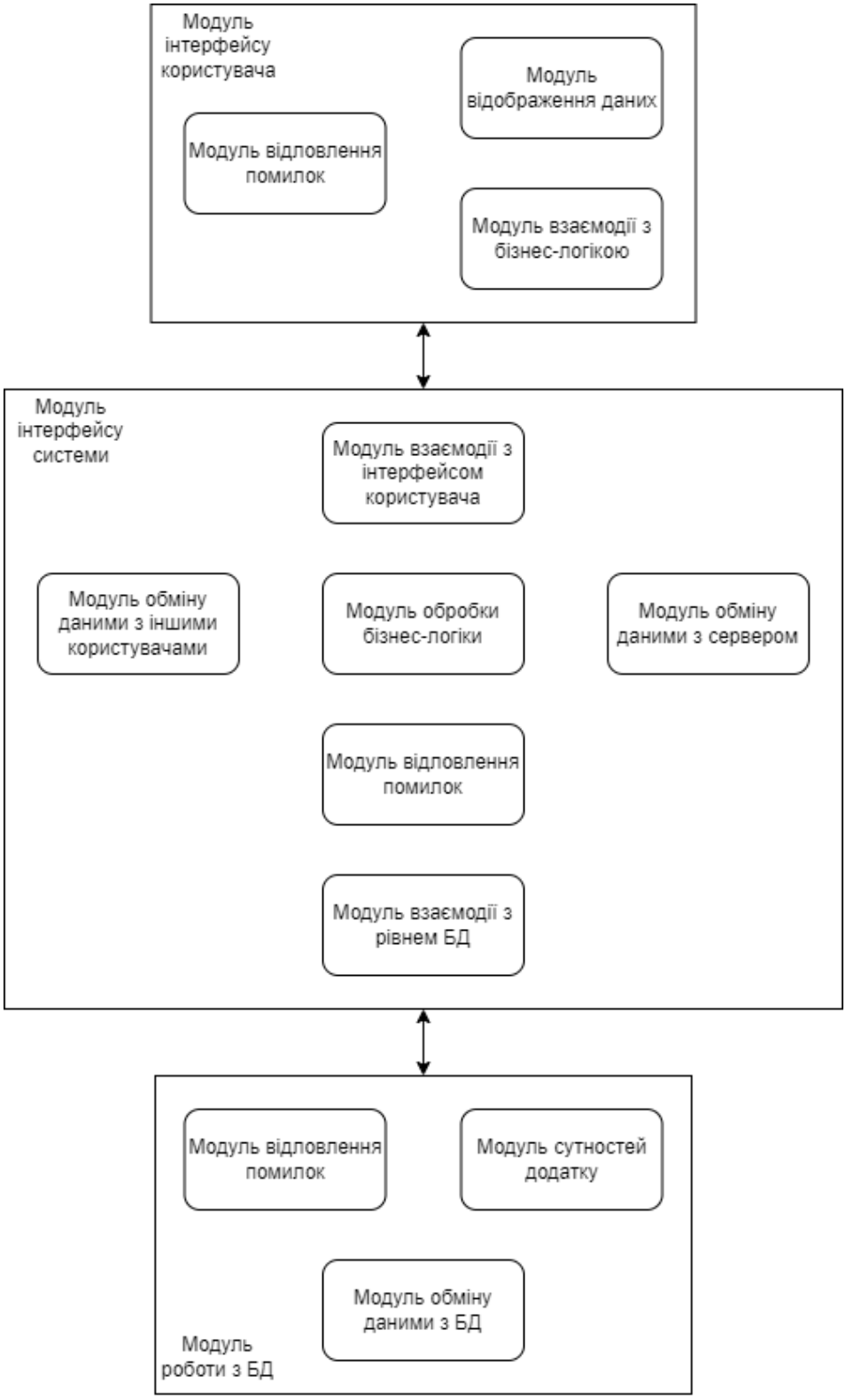


Рисунок 3.6 – Взаємодія елементів клієнтського додатку

Модуль обміну даними з клієнтським додатком відповідає за коректну обробку і перетворення усіх даних, які відправлятимуться клієнтським додаткам та отримуватимуться від них.

Модуль маршрутизації відповідає за транспортування даних клієнтським додаткам, зокрема за створення з'єднань та коректного направлення даних необхідним користувачам через відповідні з'єднання.

Модуль автентифікації користувачів відповідає за бізнес-логіку автентифікації користувачів під час процесу отримання даних та під час встановлення з'єднання із серверним додатком.

Модуль роботи з даними користувачів відповідає за бізнес-логіку отримання, обробки та перетворення персональних даних користувачів, які зберігатимуться в базі даних, зокрема за такі процеси, як реєстрація та авторизація в системі.

Модуль відловлення помилок відповідає за перевірку отриманих даних з інших модулів системи, валідацію цих даних, фіксацію та виведення даних про проблеми та помилки у разі їх виникнення.

Модуль обміну даними з рівнем роботи з БД відповідає за коректну обробку даних, які надходять як із БД, так і до неї.

Модуль роботи з базою даних серверного додатку містить такі під модулі, як:

– модуль обміну даними з рівнем бізнес-логіки;
– модуль відловлення помилок;
– модуль обміну даними з БД.

Модуль обміну даними з рівнем бізнес-логіки відповідає за отримання даних з вищого архітектурного рівня для їх подальшого запису в БД та відправку даних, отриманих з БД, на вищий архітектурний рівень.

Модуль відловлення помилок відповідає за перевірку отриманих даних з інших модулів системи, валідацію цих даних, фіксацію та виведення даних про проблеми та помилки у разі їх виникнення.

Модуль обміну даними з БД відповідає за коректне оперування даними при їх записі в БД та за їх коректне отримання.

Взаємодія компонентів та модулів серверного додатку зображено на рисунку 3.7.

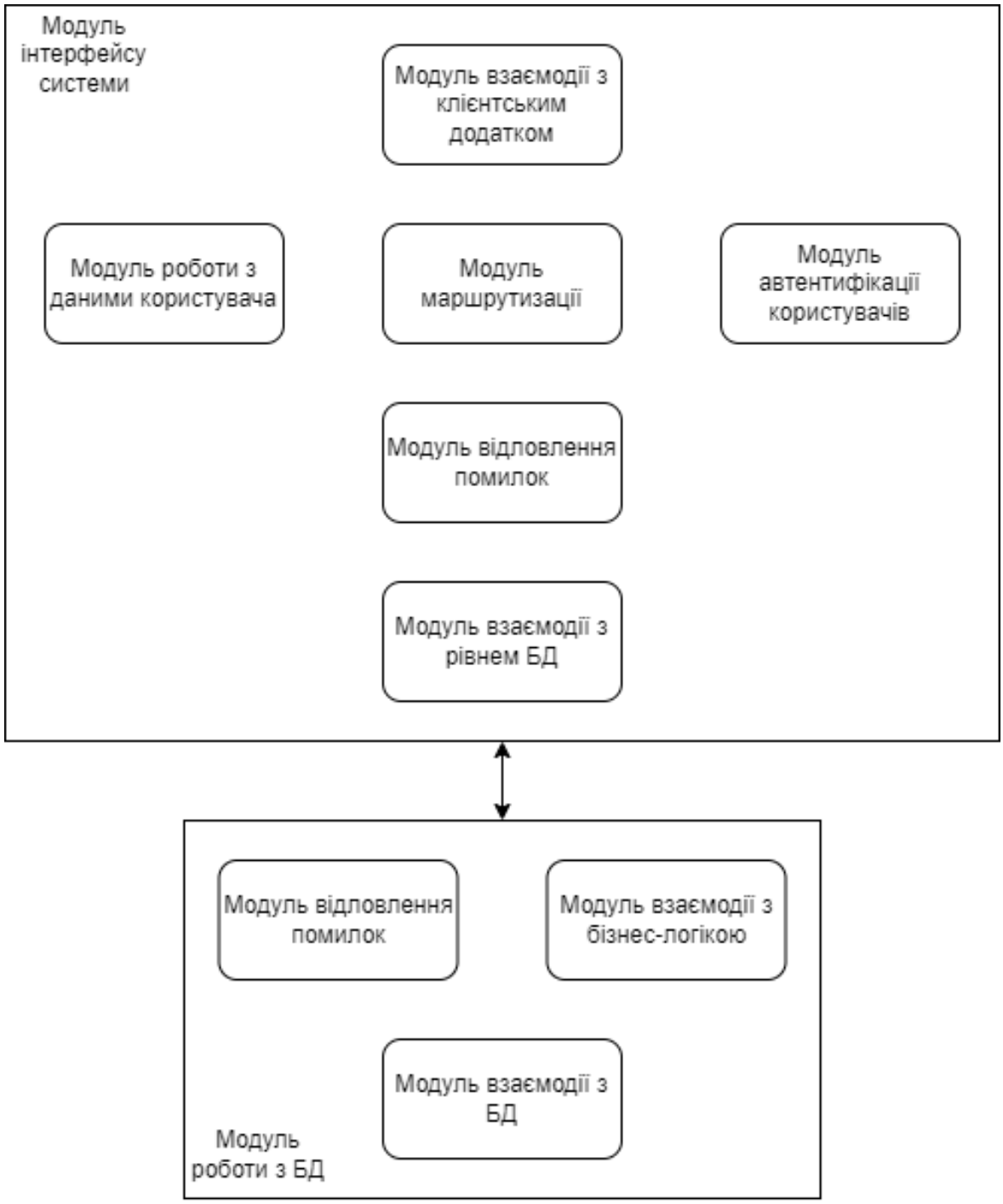


Рисунок 3.7 – Взаємодія елементів серверного додатку

Отже, було сформовано та розглянуто основні компоненти системи та їх складові, встановлено залежність та взаємозв’язок між цими компонентами та елементами всередині них. Така залежність у вигляді модулів та клієнт-серверна архітектура дозволятиме залишатись програмній системі гнучкою та піддатливою до змін і масштабування.

3.3.2 Проєктування структури даних

Наступним етапом проведемо детальніше проєктування та опис структури даних програмної системи. Розроблювана програмна система передбачає використання двох окремих структур даних, які зберігатимуться в БД: структура даних клієнтського додатку та структура даних серверного додатку.

Структура даних клієнтського додатку складається з двох таблиць, які відповідають за збереження в базу даних інформації про отримані від іншого користувача дані на випадок, якщо додаток буде закрито або втрачено з'єднання з мережею Інтернет. В такому випадку, дані не зникнуть, оскільки зберігатимуться не лише у вигляді кешу в оперативній пам'яті. Це дозволить продовжити передачу даних як тільки з'явиться можливість.

Для коректного розділення та збереження дані розділяються на п'ять окремих сутностей: сутність процесу передачі даних, сутність транзакції передачі даних, сутність персональних даних користувача, сутність налаштувань користувача та сутність контакту. Виділимо атрибути для кожної з сутностей.

Процес передачі даних має наступні атрибути:

– ідентифікатор процесу передачі даних;
– ідентифікатор відправника;
– ідентифікатор отримувача;
– тип передаваних даних;
– ознака завершеності процесу передачі даних;
– дата та час початку процесу передачі даних;
– дата та час крайньої транзакції процесу передачі даних;
– дата та час завершення процесу передачі даних.

Транзакція процесу передачі даних має наступні атрибути:

– ідентифікатор транзакції;
– ідентифікатор процесу передачі даних;
– шлях до файлу з вмістом транзакції;
– порядковий номер транзакції.

Сутність контакту містить наступні атрибути:

– ідентифікатор контакту;
– ідентифікатор користувача;
– ім’я користувача;
– відображуване ім’я;
– колір мітки.

Сутність налаштувань користувача містить такі атрибути, як:

– ідентифікатор користувача;
– мінімальна кількість транзакцій;
– максимальна кількість транзакцій;
– максимальний розмір отримуваного файлу;
– максимальний розмір файлу, який відправляється;
– тема інтерфейсу;
– автоочищення даних.

Сутність персональних даних користувача містить атрибути, які забезпечують реалізацію алгоритму верифікації повідомлень, алгоритму Діффі-Хелмана та можливість використання алгоритми асиметричного шифрування в цілому. Дана сутність містить наступні атрибути:

– ідентифікатор користувача;
– ідентифікатор сесії;
– приватний ключ;
– публічний ключ.

Структура даних серверного додатку складається з шести таблиць, які виступатимуть як сутностями бази даних, так і програмними типами даних. Серед сутностей серверного додатку такі, як:

– користувач програмної системи;
– персональні дані користувача;
– роль користувача;
– сесія користувача;
– сесія передачі даних;

– системна помилка;
– скарга на користувача;
– підозріла подія в програмній системі.

Сформулюємо та опишемо атрибути кожної сутності. Розглянемо атрибути кожної сутності окремо.

Сутність користувача програмної системи містить наступні атрибути:
– ідентифікатор користувача;
– ідентифікатор ролі користувача;
– ім’я користувача.

Сутність персональних даних користувача містить наступні атрибути:
– ідентифікатор користувача;
– логін;
– пароль;
– адреса електронної пошти.

Сутність ролі користувача містить такі атрибути, як:
– ідентифікатор ролі користувача;
– назва ролі користувача;
– опис ролі користувача.

Сутність сесії користувача містить такі атрибути, як:
– ідентифікатор сесії користувача;
– ідентифікатор користувача;
– дата та час створення сесії;
– дата та час завершення сесії;
– дата та час останньої активності сесії;
– IP-адреса пристрою.

Сутність сесії передачі даних містить наступні атрибути:
– ідентифікатор сесії передачі даних;
– ідентифікатор сесії користувача-відправника;
– ідентифікатор сесії користувача-отримувача.

Сутність системної помилки містить наступні атрибути:

– ідентифікатор системної помилки;
– ідентифікатор сесії передачі даних;
– код помилки;
– опис помилки;
– стек помилки.

Сутність скарги на користувача містить такі атрибути, як:
– ідентифікатор скарги на користувача;
– ідентифікатор сесії передачі даних;
– ідентифікатор користувача, який подав скаргу;
– причина скарги;
– опис скарги;
– дата та час фіксації скарги.

Сутність підозрілої події містить такі атрибути, як:
– ідентифікатор підозрілої події;
– ідентифікатор сесії передачі даних;
– опис підозрілої події.

Модель структури бази даних клієнтського додатку зображено на рисунку Б.1 (додаток Б). Модель структури бази даних серверного додатку зображено на рисунку Б.2 (додаток Б).

Отже, в результаті проведеного аналізу було спроєктовано структури даних для клієнтського та серверного додатку. На основі спроєктованих структур даних буде розроблено бази даних відповідних додатків.

3.3.3 Проєктування інтерфейсу користувача

Інтерфейс користувача – це певна система, яка використовується для взаємодії між користувачем та комп’ютерною системою, у даному випадку – між користувачем та програмною системою удосконаленого методу передачі даних у мережі Інтернет. Зазвичай інтерфейс користувача працює таким чином, що отримує команди від

користувача, відповідним чином обробляє їх та передає далі програмній системі, після чого може отримати команду від самої програмної системи та відобразити цю команду кінцевому користувачу.

В результаті розробки ІК необхідно отримати систему, яка буде зручною та інтуїтивно зрозумілою користувачу з будь-яким рівнем навичок користування персональним комп'ютером та мережею Інтернет.

Від користувача потрібно приховати зайві деталі реалізації алгоритмів та системи в цілому. Якщо деякі деталі необхідно залишити відкритими, такі як налаштування, їх потрібно зробити зрозумілими, а також реалізувати стандартні універсальні налаштування, щоб спростити користування програмною системою не порушивши коректності її роботи.

Загальна структура компонентів інтерфейсу користувача клієнтського додатку повинна містити наступні компоненти:

– сторінка реєстрації/авторизації;
– основна сторінка (містить перелік контактів та поле, на якому відображатиметься історія передачі даних);
– сторінка додавання нових контактів;
– сторінка налаштувань системи.

Розглянемо детальніше компоненти кожної сторінки системи.

На рисунку 3.8 зображено сторінку реєстрації/авторизації у системі з боку клієнтського додатка.

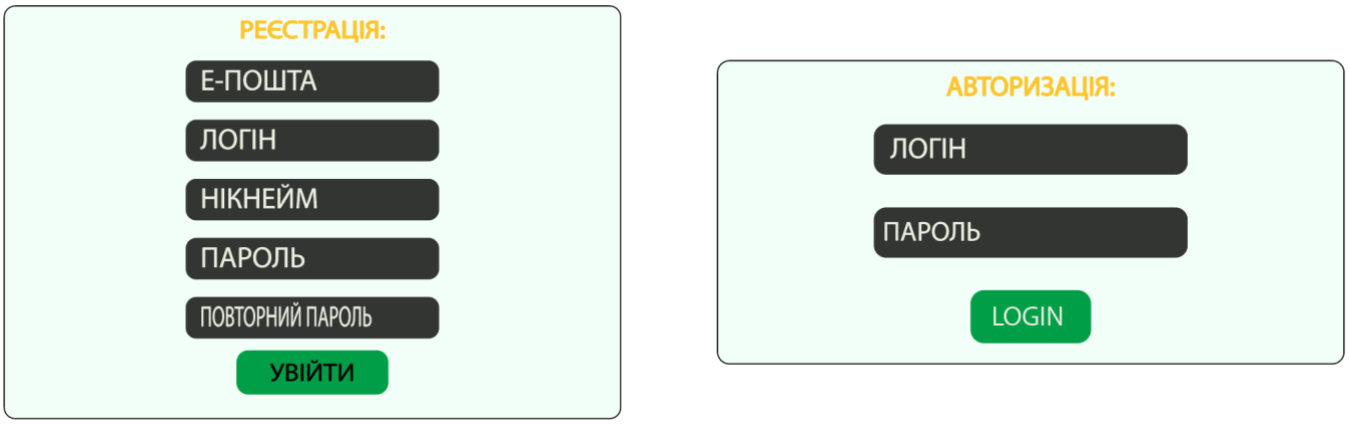


Рисунок 3.8 – Сторінка реєстрації/авторизації користувача

На даній сторінці знаходяться поля, необхідні для реєстрації або авторизації користувача, відповідно до того, що необхідно виконати користувачу. Для реєстрації користувача наявними є такі поля, як ім’я користувача, логін, пароль та адреса електронної пошти. Для валідації пароля при реєстрації, присутнє додаткове поле для повторного введення пароля. Для авторизації користувача знаходяться поля для введення логіну та пароля користувача. Для підтвердження будь-якої з дій користувача присутні кнопки для підтвердження реєстрації та авторизації у відповідних частинах сторінки.

Далі розглянемо основну сторінку інтерфейсу користувача. Основна сторінка клієнтського додатку зображена на рисунку 3.9.

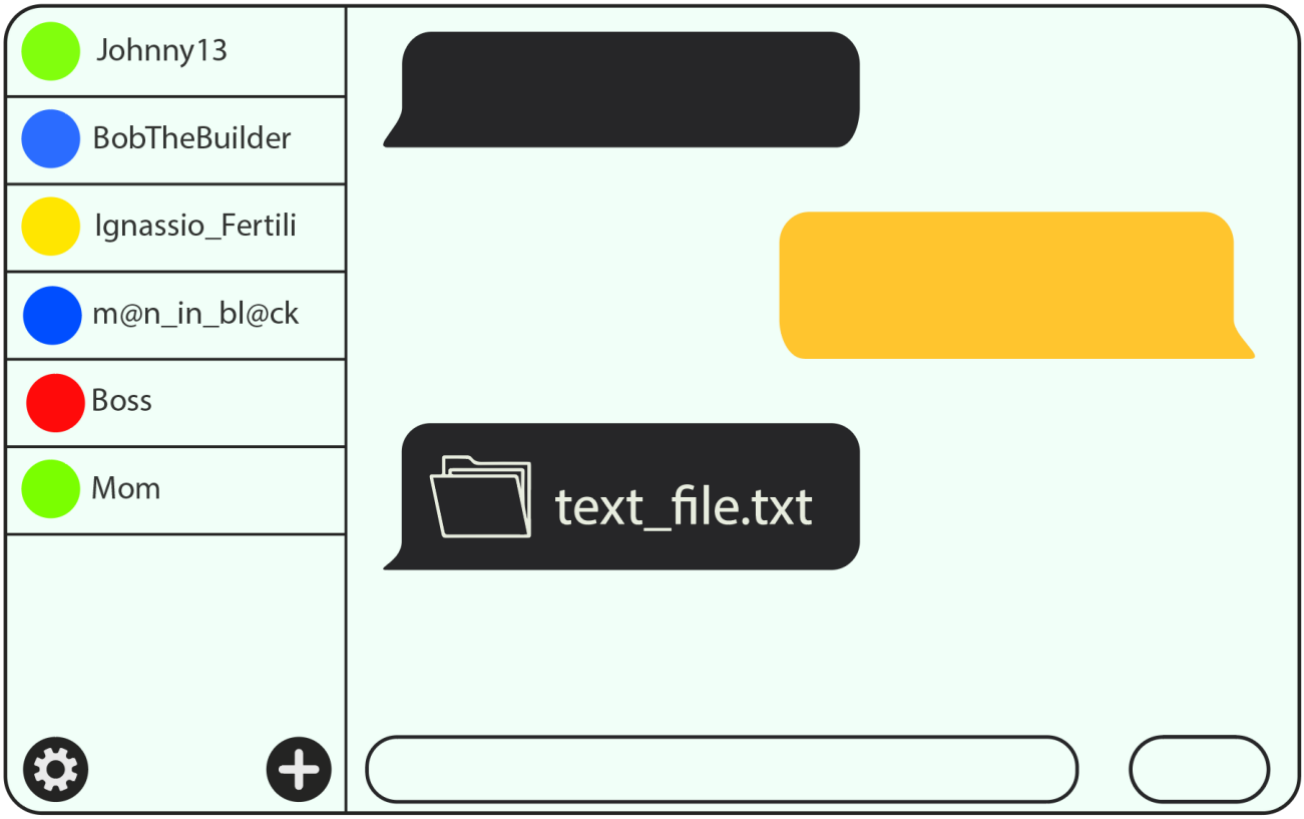


Рисунок 3.9 – Основна сторінка

Основна сторінка інтерфейсу користувача клієнтського додатку розділена на дві частини: ліва частина, в якій міститься список контактів користувача, тобто список користувачів, яких поточний користувач додавав у список контактів, або ті користувачі, з якими він проводив обмін даними та права частина, в якій

відображатиметься історія переданих даних з користувачем, обраним в лівій частині сторінки. Також внизу лівої частини сторінки знаходяться дві кнопки: кнопка додавання нового контакту та кнопка налаштувань. Натискання на ці кнопки викликатиме відкриття двох інших сторінок поверх основної сторінки: сторінки додавання нового контакту та сторінки налаштувань відповідно.

Наступною є сторінка додавання нового контакту. Дану сторінку зображено на рисунку 3.10.

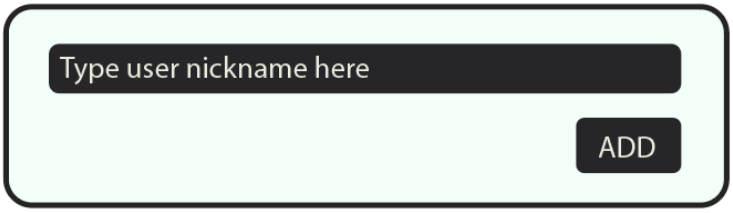


Рисунок 3.10 – Сторінка додавання нового контакту

Дана сторінка необхідна для додавання нового користувача до списку контактів поточного користувача. Сторінка містить два поля для введення даних: поле для введення ім’я користувача та поле для введення відображуваного ім’я даного користувача на основній сторінці інтерфейсу поточного користувача, у секції контактів. Поле відображуваного ім’я є необов’язковим для заповнення. Також на сторінці присутня кнопка, при натисканні на яку відбудеться спроба додавання даного користувача до списку контактів, якщо такий користувач існує.

Останньою сторінкою інтерфейсу користувача клієнтського додатку є сторінка налаштувань. Ця сторінка зображена на рисунку 3.11.

Сторінка налаштувань містить поля для зміни значень основних атрибутів сутності налаштувань користувача клієнтського додатку, зокрема такі поля, як: поле для зміни мінімальної кількості транзакцій передачі даних, поле для зміни максимальної кількості транзакцій передачі даних, поле для зміни максимального розміру вхідного файлу, поле для зміни максимального розміру вихідного файлу, а також перемикачі для значень теми інтерфейсу користувача та значень автоматичного очищення даних.

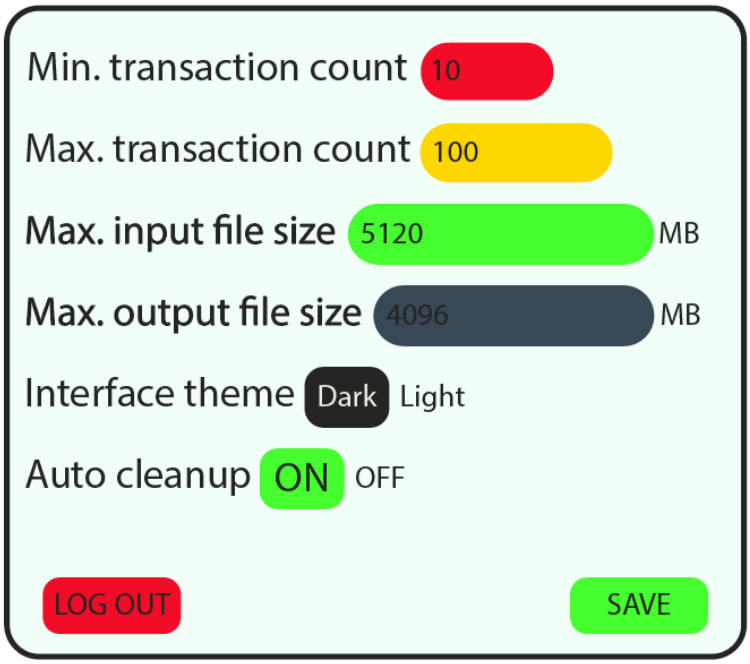


Рисунок 3.11 – Сторінка налаштувань

3.4 Аналіз та вибір засобів реалізації програмної системи

При розробці програмної системи, яка реалізує удосконалений метод передачі даних у мережі Інтернет з використанням алгоритму верифікації повідомлень та алгоритму Діффі-Хелмана, необхідно обрати засоби реалізації для кожного з компонентів системи, а також для деяких модулів цих компонентів.

Усього є три основні компоненти, для яких необхідно обрати засоби розробки програмної системи, а саме: бекенд компонент, як для серверного додатку, так і для клієнтського додатку, фронтенд компонент, як для клієнтського додатку, так і для серверного та компонент бази даних, також для обох додатків.

Обираючи засіб розробки фронтенд-системи основним запитання, на яке повинен відповісти розробник, є те, чи повинен додаток бути звичайним додатком, який встановлюється на пристрій користувача, чи веб-додатком, який працюватиме через браузер. Якщо розроблювана система буде веб-додатком, розробка може проводитись на скриптовій мові програмування JavaScript, з використанням мови розмітки HTML, таблиці стилів CSS та з використанням інших засобів. У випадку, якщо система буде звичайним додатком, розробник має на вибір найпопулярніші

мови програмування та їх засоби реалізації інтерфейсу користувача. Серед таких мов програмування, наприклад, Java, C#, Python та C++. Розроблюваний додаток повинен бути максимально захищеним та мати прямий доступ до даних користувача, які зберігатимуться на пристрої користувача. Саме тому для реалізації було обрано звичайний додаток, який встановлюватиметься на пристрій користувача. Для реалізації додатку було обрано мову програмування C#, оскільки дана мова програмування має багато засобів для розробки програмного забезпечення на різні платформи, такі як: персональні комп'ютери на системі Windows, MacOS, мобільні додатки для систем Android, iOS та WindowsPhone.

При виборі засобу розробки бекенд-системи зазвичай розробники програмного забезпечення обирають з таких найпопулярніших засобів розробки, як: Java, C#, C++ та JavaScript. Розроблювана система повинна бути універсальною, написана на мові програмування, яка є кросплатформною, тобто може працювати на різних типах систем. Бекенд-система серверного додатку повинна бути реалізована у вигляді API, яке може отримувати запити та надсилати відповіді, а також встановлювати з’єднання за допомогою сокетів для подальшої передачі даних протоколом TCP. Саме тому для розробки даної програмної системи було обрано мову програмування C#, яка включає в себе усі необхідні компоненти та засоби розробки такої системи, а також є мовою програмування, яка динамічно розвивається та отримує регулярні оновлення, що підвищують її швидкість та безпеку. Бекенд-система клієнтського додатку може бути реалізована на тій же мові програмування, залежно від вибору засобу реалізації фронтенд-системи. Оскільки для реалізації фронтенд-системи клієнтського додатку було обрано мову програмування C#, для розробки бекенд-системи було обрано ідентичну мову програмування.

При виборі засобів реалізації бази даних звертати увагу потрібно на такі аспекти, як: швидкість, безпека та простота взаємодії БД з іншими програмними системами. Найпопулярнішими базами даних є MSSQL, MySQL, Oracle та SQLite. Оскільки база даних не буде перевантажена зайвою інформацією та не повинна займати зайве місце на пристрої користувача, а також її установка не повинна вимагати серйозних навичок

користування, для реалізації бази даних було обрано саме систему управління базами даних SQLite.

3.5 Висновки

В результаті виконання аналізу предметної області у сфері передачі даних в мережі Інтернет, дослідження наявних способів передачі даних та вдосконалення наявних рішень за допомогою виправлення виявлених проблем було сформовано покращену концепцію та методологію передачі даних у мережі Інтернет.

Під час виконання даного розділу було обрано та описано алгоритми, які використовуватимуться під час програмної реалізації розробленого методу.

Наступним етапом було описано вимоги до програмної системи, визначивши мінімальний необхідний функціонал для коректної роботи програмної системи.

Також було проведено розробку структури програмної системи. Під час розробки структури програмної системи було описано компоненти програмної системи, їх модулі та підмодулі.

Далі було описано структуру даних програмної системи, а саме структуру даних усіх її компонентів, визначено необхідні сутності та атрибути.

В результаті отриманого проєкту програмної системи було проведено аналіз наявних засобів та технологій реалізації схожих програмних систем та обрано технології, які найкраще підходять для реалізації необхідного функціоналу та концепції програмної системи.

Отже, під час написання даного розділу було проведено проєктування ПС на рівні алгоритмів, інтерфейсу користувача, структури даних та архітектури системи.

Наступним кроком після проведеного проєктування є безпосередня реалізація отриманих рішень.

4 РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ ПРОГРАМНОЇ СИСТЕМИ

4.1 Програмна реалізація

4.1.1 Структура та призначення модулів системи, їхній взаємозв’язок

В попередньому розділі було описано основні компоненти розроблюваної програмної системи, а також основні модулі архітектури програмної системи.

Під час програмної реалізації будь-якої програмної системи можна виділити більш детальні модулі, реалізація яких є більш ретельною та важливою для коректного функціонування системи.

Оскільки розроблювана програмна система складається з двох основних компонентів, клієнтського та серверного додатку, кожна з цих складових містить такі модулі, реалізація яких є більш важливою для реалізації концепту та розробки удосконаленого методу передачі даних у мережі Інтернет. Ці модулі знаходяться на рівні бізнес-логіки обох додатків, оскільки відповідають саме за реалізацію алгоритмів визначеного раніше функціоналу. Розглянемо детальніше модулі кожної з систем.

Клієнтський додаток містить чотири модулі, реалізація яких має безпосередній вплив на функціонал та концепт розроблюваної системи, а саме такі модулі, як:

– модуль комунікації з сервером;
– модуль автентифікації на стороні клієнта;
– модуль верифікації на стороні клієнта;
– модуль шифрування даних.

Розглянемо детальніше призначення кожного з цих модулів.

Модуль комунікації з сервером відповідає за встановлення зв’язку з сервером, підготовку даних перед відправкою до сервера шляхом приведення цих даних до вигляду, який буде зрозумілий серверу та іншому користувачу, за коректне приведення даних до зрозумілого вигляду іншим рівням архітектури при отриманні цих даних з сервера, а також за розірвання зв’язку з сервером.

Модуль автентифікації на стороні клієнта відповідає за перевірку правильності персональних даних користувача, порівняння даних, введених при вході в додаток з

тими, які зберігаються в локальній базі даних та за коректне збереження і обробку персональних даних користувача при реєстрації та авторизації користувача в додатку.

Модуль верифікації на стороні клієнта відповідає за генерацію випадкового повідомлення, яке відправлятиметься отримувачу для його верифікації, перевірку отриманого підписаного повідомлення та подальші дії системи у разі не проходження клієнтом-отримувачем процесу верифікації.

Модуль шифрування даних відповідає за шифрування та дешифрування даних на основі приватного та публічного ключа користувача.

Серверний додаток також містить чотири модулі, які описують основні функції програмної системи, необхідні для реалізації розробленого удосконаленого методу передачі даних в мережі Інтернет. Серед цих модулів такі, як:

– модуль маршрутизації;
– модуль автентифікації на стороні сервера;
– модуль верифікації на стороні сервера;
– модуль підключення бази даних.

Розглянемо детальніше кожен з програмних модулів серверного додатку розроблюваної програмної системи.

Модуль маршрутизації відповідає за коректне отримання повідомлення від одного користувача, його перевірку іншими модулями, такими як модуль автентифікації, та відправку цього повідомлення адресату. Для коректної маршрутизації певні дані користувачів, їх сесій та сесій передачі даних зберігатимуться в базі даних.

Модуль автентифікації на стороні сервера відповідає за автентифікацію користувачів клієнтських додатків та за перевірку коректності підключень користувачів перед початком сесії відправки даних.

Модуль верифікації на стороні сервера відповідає за генерацію приватних та публічних ключів користувачів на основі їх персональних даних та перевірку їх коректності при потребі.

Модуль підключення до бази даних відповідає за встановлення з’єднання з базою даних серверного додатку, регулювання доступу до нього та підтримку подальшого масштабування бази даних і підключень до неї під час роботи сервера.

Кожен з даних модулів програмно представлений окремими класами-сервісами, які реалізовують окремі методи для кожної функції з метою дотримання парадигм та концепцій більш універсального, масштабованого, зрозумілішого та «чистішого» програмування.

4.1.2 Розробка програмних модулів

Під час реалізації кожного з перерахованих програмних модулів компонентів розроблюваної системи, необхідно виконувати написання класів та методів для розділення відповідальності та полегшення майбутнього масштабування коду. Розглянемо деякі методи кожного з програмних модулів.

Почнемо з розгляду методів модулів клієнтського додатку. Під час реалізації модулю комунікації з сервером клієнтському додатку необхідно реалізувати метод для встановлення з’єднання з сервером та метод для розірвання з’єднання з сервером при закінченні сесії користувача як шляхом закриття додатку, так і шляхом виходу з облікового запису користувача додатку.

Код методу для встановлення з’єднання з сервером виглядає наступним чином:

```
static void ConnectToServer(string serverIp, int serverPort)
{
    // Створення клієнтського сокету
    _client = new TcpClient(serverIp, serverPort);
    _log.Info("Підключено до сервера.");

    // Отримання потоків для читання та запису
    _stream = client.GetStream();
}
```

Код для розриву з’єднання з сервером виглядає наступним чином:

```
static void DisconnectFromServer()
{
    // Закриття з'єднання
    _stream.Close();
    _client.Close();
    _log.Info("З'єднання з сервером закрито.");
}
```

Для коректної роботи цих методів необхідно помістити їх в один клас, який займатиметься безпосередньо встановленням та налаштуванням з’єднання з сервером. Такий клас повинен бути реалізований за допомогою шаблону проєктування Singleton, оскільки в інакшому випадку такий клас викликатиме зайве навантаження на систему, щоразу створюючи нове з’єднання при кожній спробі відправити дані.

Такі налаштування, як IP-адреса та порт сервера повинні бути вшитими в код, або зберігатися в налаштуваннях додатку в окремих файлах, в зашифрованому вигляді, щоб звичайний користувач не зміг дуже легко дізнатися адресу сервера і скористатися цими даними з шкідливими для системи мотивами.

Процес автентифікації на стороні клієнта передбачає, в тому числі, порівняння введених користувачем даних при вході в систему з тими, які зберігаються в локальній базі даних. Код методу такої перевірки виглядає наступним чином:

```
public bool AuthenticateUser(string enteredUsername, string
enteredPassword)
{
    // Знаходження користувача в базі даних за логіном
    User user = _userService.GetByLogin(enteredUsername);

    if (user != null)
    {
        // Порівняння хешованого паролю
        string enteredPasswordHash = HashPassword(enteredPassword,
user.Salt);

        if (string.Equels(enteredPasswordHash, user.PasswordHash))
        {
            _log.Warning("Ви увійшли в систему успішно.");
            return true;
        }
    }
```

```
    _log.Warning("Помилка входу. Невірний логін або пароль.");
    return false;
}
```

Для реалізації модулю верифікації користувача на стороні клієнтського додатку необхідно, в тому числі, виконати реалізацію методу для генерації випадкового повідомлення для підпису отримувачем. Код методу генерування випадкового повідомлення виглядає наступним чином:

```
static string GenerateRandomMessage()
{
    var envVarName = ClientConstants.MESSAGE_LENGTH;
    var messageLength =
(int)Environment.GetEnvironmentVariable(envVarName);

    // Генерація випадкового повідомлення, наприклад, для
демонстраційних цілей
    return Guid.NewGuid(length: messageLength).ToString();
}
```

Даний метод викликатиметься під час формування звернення для відправки даних, а результат поміщатиметься в «контейнер» запиту разом з іншими даними, після чого повідомлення діставатиметься отримувачем та підписуватиметься за допомогою власного приватного ключа.

Реалізовуючи модуль шифрування даних, необхідно реалізувати метод, який безпосередньо шифруватиме дані за допомогою алгоритму Діффі-Хелмана, після чого ці дані буду відправлені отримувачу та дешифровані за допомогою публічного ти приватного ключа отримувача. Код методу шифрування даних з використанням алгоритму Діффі-Хелмана виглядає наступним чином:

```
public string Encrypt()
{
    using (ECDiffieHellmanCng alice = new ECDiffieHellmanCng())
    {

        alice.KeyDerivationFunction =
ECDiffieHellmanKeyDerivationFunction.Hash;
        alice.HashAlgorithm = CngAlgorithm.Sha256;
        alicePublicKey = alice.PublicKey.ToByteArray();
```

```
        CngKey bobKey = CngKey.Import(bob.bobPublicKey,
CngKeyBlobFormat.EccPublicBlob);
        byte[] aliceKey = alice.DeriveKeyMaterial(bobKey);
        CryptoStream cs = new CryptoStream(ciphertext,
aes.CreateEncryptor(), CryptoStreamMode.Write);
        byte[] plaintextMessage =
Encoding.UTF8.GetBytes(secretMessage);
        cs.Write(plaintextMessage, 0, plaintextMessage.Length);
        cs.Close();
        encryptedMessage = ciphertext.ToArray();

        return encryptedMessage.ToString();
    }
}
```

Далі розглянемо реалізацію деяких методів модулів серверного додатку програмної системи. Для відправки даних клієнту з сервера, необхідно реалізувати метод, який буде проводити маршрутизацію шляхом отримання адреси отримувача з бази даних та встановлення з’єднання саме за адресою останнього активного сеансу. Код такого методу виглядає наступним чином:

```
static void SendDataToClient(long receiverId, string message)
{
    var receiver = _userService.GetById(id);
    try
    {
        using (TcpClient client = new TcpClient(receiver.IP,
receiver.Port))
        using (NetworkStream stream = client.GetStream())
        {
            // Конвертація рядка в байти
            byte[] data = Encoding.UTF8.GetBytes(message);

            // Відправлення даних на клієнтський додаток
            stream.Write(data, 0, data.Length);

            _log.Info("Дані успішно відправлені на клієнтський
додаток.");
        }
    }
    catch (Exception ex)
    {
        _log.Error($"Помилка: {ex.Message}");
    }
}
```

Для автентифікації користувача на стороні серверного додатку необхідно розробити метод, який перевірятиме наявність з'єднання з користувачем за допомогою перевірки наявності в базі даних інформації про сеанс користувача. Дана перевірка відбуватиметься як перед початком процесу передачі даних, так і під час процесу передачі даних при кожній спробі відправити нову транзакцію з даними. Код даного методу виглядає наступним чином:

```
static bool CheckUserConnectionData(string username)
{
    var isUserExists = _userService.ExistsByUsername(username);
    if (!isUserExists)
        throw new UserDoesNotExistException(username);

    var session = _sessionService.GetByUsername(username);
    if (string.IsNullOrWhitespace(session.IP) || session.Port is
default || !session.IsConnected)
        return false;

    return true;
}
```

Для коректного відпрацювання модулю верифікації на стороні серверного додатка необхідно реалізувати метод, який генеруватиме приватний та публічний ключі для користувача під час реєстрації в системі. Ці ключі генеруватимуться на основі персональних даних користувача, таких як логін, пароль та ім'я користувача. Таким чином, ці ключі будуть унікальними, оскільки персональні дані користувача також унікальні. Це дозволить не зберігати ці ключі на сервері, що зменшить ризик витоку даних про пару приватного та публічного ключа користувача, хоч і підвищить відповідальність користувача за збереження цих ключів і їх введення при авторизації. Код даного методу виглядає наступним чином:

```
static KeyPair GenerateKeyPair(string username, string login, string
password)
{
    // Використання алгоритму RSA для генерації ключів
    using (RSACryptoServiceProvider rsa = new
RSACryptoServiceProvider())
    {
        // Отримання інформації для генерації ключів
```

```
        string keyInfo = $"{username}{login}{password}";

        // Перетворення рядка в байти
        byte[] keyBytes = Encoding.UTF8.GetBytes(keyInfo);

        // Генерація ключів
        CspParameters cspParams = new CspParameters
        {
            KeyContainerName =
Convert.ToBase64String(MD5.Create().ComputeHash(keyBytes)),
            Flags = CspProviderFlags.UseMachineKeyStore
        };

        rsa.PersistKeyInCsp = false;
        rsa.ImportParameters(rsa.ExportParameters(true));

        // Повернення публічного та приватного ключів
        return new KeyPair
        {
            PublicKey =
Convert.ToBase64String(rsa.ExportSubjectPublicKeyInfo()),
            PrivateKey =
Convert.ToBase64String(rsa.ExportRSAPrivateKey())
        };
    }
}
```

4.1.3 Реалізація моделі бази даних

Для реалізації моделі бази даних Найзручнішим способом є спосіб керування контекстом бази даних під назвою «codefirst». Таким чином, сутності бази даних – це класи програми, які конвертуються та використовуються окремими бібліотеками для звертань та керувань відповідними сутностями в базі даних.

Розглянемо реалізацію моделей програмної системи детальніше. Оскільки розроблювана програмна система складається з двох основних компонентів, а саме з клієнтського та серверного додатку, тому програмна система міститиме дві окремі моделі бази даних для кожного додатку.

Для клієнтського додатку однією з сутностей моделі бази даних є сутність процесу передачі даних. Для реалізації цієї сутності методом «codefirst» необхідно реалізувати відповідний клас у коді програми. Код даного класу сутності процесу передачі даних виглядає наступним чином:

```
public class DataTransferProcess : EntityBase
{
    public long SenderId { get; set; }
    public long ReceiverId { get; set; }
    public DataType DataType { get; set; }
    public bool IsTransferCompleted { get; set; }
    public DateTime TransferStartTime { get; set; }
    public DateTime TransferLastTransactionTime { get; set; }
    public DateTime? TransferEndTime { get; set; }
}
```

Для серверного додатку однією з сутностей є сутність сесії, або сеансу, користувача. Програмний код класу для даної сутності виглядає наступним чином:

```
public class UserSession : EntityBase
{
    public long UserId { get; set; }
    public DateTime SessionStartTime { get; set; }
    public DateTime LastInterractionTime { get; set; }
    public DateTime? SessionEndTime { get; set; }
    public string IP { get; set; }
}
```

Для підключення бази даних до коду будь-якого додатку необхідно реалізувати конфігураційний метод, який виконуватиме підключення до бази даних лише один раз при запуску додатку, оскільки багаторазове підключення до бази даних при кожному зверненні до БД може призвести до різкого падіння продуктивності системи та зайняти багато ресурсів пристрою. Для зручності розділимо отримання даних для підключення та безпосереднє підключення на два різні процеси. Код отримання даних для підключення до бази даних серверного додатку виглядає наступним чином:

```
private static string GetConnectionString()
{
    try
    {
        var host = Environment.GetEnvironmentVariable(DB_HOST);
        if (string.IsNullOrWhiteSpace(host))
            throw new Exception(DB_HOST);

        var user = Environment.GetEnvironmentVariable(DB_USER);
        if (string.IsNullOrWhiteSpace(user))
```

```
            throw new Exception(DB_USER);

        var pass = Environment.GetEnvironmentVariable(DB_PASS);
        if (string.IsNullOrWhiteSpace(pass))
            throw new Exception(DB_PASS);

        var name = Environment.GetEnvironmentVariable(DB_NAME);
        if (string.IsNullOrWhiteSpace(name))
            throw new Exception(DB_NAME);

        return
$"Host={host};Username={user};Password={pass};Database={name}";
    }
    catch (Exception ex)
    {
        throw new MissingEnvVarException(ex.Message);
    }
}
```

Підключення бази даних відбувається в класі Program. Саме в цьому класі знаходиться конвеєр обробки запитів, який займається конфігурацією, підключенням різноманітних проміжних функцій, налаштуванням залежностей та конфігурацією баз даних. Код методу підключення бази даних, який викликатиметься в класі Program, виглядає наступним чином:

```
public static IServiceCollection AddServerDbContext(
this IServiceCollection services)
{
    var connectionString = GetConnectionString();
    services.AddDbContext<ServerContext>(options =>
    {
        options.UseSql(connectionString, x =>
x.MigrationsHistoryTable("__ServerMigrationsHistory"));
    });

    return services;
}
```

4.1.4 Реалізація методів поліпшення технічних характеристик системи

Для реалізації функціоналу, який безпосередньо імплементує алгоритм удосконалення передачі даних в мережі Інтернет необхідно реалізувати відповідні

класи та методи. Ці класи та методи необхідно реалізувати як на стороні клієнтського додатку, так і на стороні серверного додатку. Розглянемо деякі методи, необхідні для повноцінної імплементації алгоритму покращення передачі даних.

На стороні клієнтського додатку одним з методів, які необхідно реалізувати для імплементації розробленого методу удосконалення передачі даних, є метод для порівняння повідомлення, яке відправник генерував для отримувача з тим, яке йому повернув підписаним отримувач. Для цього також необхідно попередньо провести підписання згенерованого повідомлення самим відправником. Код методу для підписання повідомлення відправником виглядає наступним чином:

```
static string SignMessage(string message, string privateKey)
{
    using (RSACryptoServiceProvider rsa = new
RSACryptoServiceProvider())
    {
        rsa.FromXmlString(privateKey);

        // Підпис повідомлення відправником
        byte[] messageBytes = Encoding.UTF8.GetBytes(message);
        byte[] signatureBytes = rsa.SignData(messageBytes, new
SHA256CryptoServiceProvider());

        // Повернення підпису у вигляді рядка
        return Convert.ToBase64String(signatureBytes);
    }
}
```

Код методу для порівняння підписів відправника та користувача виглядає наступним чином:

```
static void VerifySignature(string message, string signature, string
publicKey)
{
    using (RSACryptoServiceProvider rsa = new
RSACryptoServiceProvider())
    {
        rsa.FromXmlString(publicKey);

        // Перевірка підпису отримувачем
        byte[] messageBytes = Encoding.UTF8.GetBytes(message);
        byte[] signatureBytes = Convert.FromBase64String(signature);
```

```
        bool isSignatureValid = rsa.VerifyData(messageBytes, new
SHA256CryptoServiceProvider(), signatureBytes);

        if (isSignatureValid)
        {
            _log.Info("Підпис валідний. Повідомлення відправлено
вірно.");
        }
        else
        {
            _log.Error("Помилка: Підпис не валідний. Повідомлення може
бути змінено або не відправлено вірно.");
        }
    }
}
```

Для імплементації розробленого алгоритму на стороні сервера необхідно, в тому числі, розробити метод для порівняння IP користувача, отриманого в запиті, з IP користувача, записаним в сутності сесії користувача в базі даних. Таким чином реалізовується автентифікація на стороні сервера. Код даного методу виглядає наступним чином:

```
public bool IsRequestIPValid(string username)
{
    string databaseUserIpAddress =
_sessionService.GetByUsername(username);

    // Отримання IP-адресу клієнта
    IPAddress clientIpAddress =
((IPEndPoint)_client.Client.RemoteEndPoint).Address;

    // Порівняння IP-адресів
    bool isIpMatch =
IPAddress.Parse(ipAddress1).Equals(databaseUserIpAddress,
clientIpAddress.ToString());

    if (isIpMatch)
    {
        _log.Info("IP-адреси співпадають.");
        return true;
    }

    _log.Error("IP-адреси не співпадають.");
    return false;
}
```

4.2 Результати тестування системи та їх аналіз

4.2.1 Вибір методів тестування

Тестування програмної системи є важливим елементом у життєвому циклі розробки програмного забезпечення. Процес тестування виявляє помилки та недоліки програмної системи, дозволяє зрозуміти, які модулі чи компоненти системи потребують оптимізації.

Існує декілька найпопулярніших методів тестування:

– інтеграційне тестування;
– тестування продуктивності;
– модульне тестування.

Серед найпоширеніших методів тестування для розроблюваної системи найбільш корисно буде використати саме метод модульного тестування, оскільки цей метод тестування використовується для точкової перевірки окремих модулів та методів програмної системи, що дозволяє більш детально дослідити надійність та відмовостійкість кожної функції окремо та в цілому.

Оскільки усі компоненти та модулі програмної системи написані з використанням технологій мови програмування C#, для проведення модульного тестування необхідно використовувати фреймворк для модульного тестування, написаний на тій же мові програмування.

Серед найпоширеніших засобів модульного тестування на мові програмування C# є бібліотека xUnit. Особливістю цієї бібліотеки є те, що її розробники проводять постійні оновлення та виправлення функціоналу бібліотеки, додають новий функціонал. В цілому, якщо порівняти бібліотеку xUnit з іншими наявними бібліотеками для проведення модульного тестування, xUnit має значно більше функціоналу та зручніший інтерфейс взаємодії з усіма методами, інтерфейсами та класами бібліотеки.

Основними модулями та методами, які необхідно піддати детальному тестуванню є модулі, відповідальні за функціонал, який імплементує удосконалений метод передачі даних у мережі Інтернет. Таким чином, необхідно тестувати такі

програмні модулі, як модулі верифікації на стороні клієнтського та серверного додатку, а також модулі автентифікації на стороні клієнтського та серверного додатку і їх взаємодію.

Дане модульне тестування буде проведено на серверному та клієнтському додатку окремо та незалежно одне від одного, що дозволить перевірити більше сценаріїв одразу.

4.2.2 Розробка тестових сценаріїв

Для початку, було проведено розробку сценаріїв тестування для клієнтського додатку програмної системи. Під час модульного тестування даного компоненту системи буде проведено тестування коректність роботи та взаємодії методів з модулів автентифікації та верифікації користувачів.

Таким чином, модульне тестування серверного додатку дозволить дослідити надійність та відмовостійкість функцій, необхідний для імплементації удосконаленого методу передачі даних.

Розглянемо детальніше тест-кейси, розроблені для модульного тестування клієнтського додатку (таблиця 4.1).

Таблиця 4.1 – Тест-кейси для модульного тестування клієнтського додатку

№	Модуль/функція	Вихідні дані	Очікуваний результат
1	2	3	4
С-М-1	Модуль автентифікації/валідація введених користувачем персональних даних	Коректні логін, пароль, ім’я користувача	Користувача успішно автентифіковано та надано доступ до усіх функцій системи; відображено наявні контакти та збережені дані історій передач даних за їх наявнсоті
С-М-2	Модуль автентифікації/валідація введених користувачем персональних даних	Некоректні логін, пароль, ім’я користувача	Користувача не автентифіковано, виведено повідомлення про невірно введені якісь із персональних даних користувача

Кінець таблиці 4.1

1	2	3	4
C-M-3	Модуль верифікації/перевірка отриманого підписаного повідомлення	Коректно підписане іншим користувачем повідомлення, повідомлення	Користувача успішно верифіковано; надано дозвіл для продовження процесу передачі даних
C-M-4	Модуль верифікації/перевірка отриманого підписаного повідомлення	Некоректно підписане іншим користувачем повідомлення, повідомлення	Користувача не верифіковано; дозвіл для продовження процесу передачі даних не надано; повернено повідомлення про те, що користувач міг бути скомпрометований

Наступним етапом було проведено розробку сценаріїв тестування для серверного додатку програмної системи. В процесі модульного тестування серверного додатку тестуванню буде піддано модуль автентифікації та модуль верифікації користувачів.

Таким чином, модульне тестування серверного додатку дозволить дослідити надійність та відмовостійкість функцій, необхідний для імплементації удосконаленого методу передачі даних.

Наведемо детальний опис тест-кейсів, розроблених для модульного тестування серверного додатку (таблиця 4.2).

Таблиця 4.2 – Тест-кейси для модульного тестування серверного додатку

№	Модуль/функція	Вихідні дані	Очікуваний результат
1	2	3	4
S-M-1	Модуль автентифікації/валідація IP користувача	Коректна IP-адреса користувача з бази даних, коректна IP-адреса з запиту	Користувача успішно автентифіковано; надано дозвіл для продовження процесу передачі даних
S-M-2	Модуль автентифікації/валідація IP користувача	Коректна IP-адреса користувача з бази даних, некоректна IP-адреса з запиту	Користувача не автентифіковано; дозвіл для продовження процесу передачі даних не надано; повернено повідомлення про ймовірну компрометацію користувача

Кінець таблиці 4.2

1	2	3	4
S-M-3	Модуль верифікації/генерація приватного та публічного ключа	Коректні логін, пароль, ім’я користувача	Персональні дані користувача перевірено успішно; пару приватного та публічного ключа згенеровано успішно
S-M-4	Модуль верифікації/генерація приватного та публічного ключа	Некоректні логін, пароль, ім’я користувача	Персональні дані користувача перевірено з помилкою; пару приватного та публічного ключа не згенеровано; повернено повідомлення про невірно введені персональні дані

4.2.3 Аналіз результатів тестування

Тестування програмної системи відбувалося в такому ж порядку, в якому було розроблено тестові сценарії, тобто спочатку було проведено тестування модулів клієнтського додатку, після чого було проведено тестування модулів серверного додатку, передбачених тестовими сценаріями.

Розроблені тестові сценарії дозволили покрити основний функціонал, необхідний для реалізації удосконаленого методу передачі даних. Наведемо результати тестування описаних раніше тестових сценаріїв (таблиця 4.3).

Таблиця 4.3 – Результати проведеної перевірки тестових сценаріїв

№	Модуль/функція	Вихідні дані	Очікуваний результат	Різниця між реальним та очікуваним результатом
1	2	3	4	5
C-M-1	Модуль автентифікації/валідація введених користувачем персональних даних	Коректні логін, пароль, ім’я користувача	Користувача успішно автентифіковано та надано доступ до усіх функцій системи; відображено наявні контакти та збережені дані історій передач даних за їх наявнсоті	Відсутня

Кінець таблиці 4.3

1	2	3	4	5
C-M-2	Модуль автентифікації/валідація введених користувачем персональних даних	Некоректні логін, пароль, ім’я користувача	Користувача не автентифіковано, виведено повідомлення про невірно введені якісь із персональних даних користувача	Відсутня
C-M-3	Модуль верифікації/перевірка отриманого підписаного повідомлення	Коректно підписане іншим користувачем повідомлення, повідомлення	Користувача успішно верифіковано; надано дозвіл для продовження процесу передачі даних	Відсутня
C-M-4	Модуль верифікації/перевірка отриманого підписаного повідомлення	Некоректно підписане іншим користувачем повідомлення, повідомлення	Користувача не верифіковано; дозвіл для продовження процесу передачі даних не надано; повернено повідомлення про те, що користувач міг бути скомпрометований	Відсутня
S-M-1	Модуль автентифікації/валідація IP користувача	Коректна IP-адреса користувача з бази даних, коректна IP-адреса з запиту	Користувача успішно автентифіковано; надано дозвіл для продовження процесу передачі даних	Відсутня
S-M-2	Модуль автентифікації/валідація IP користувача	Коректна IP-адреса користувача з бази даних, некоректна IP-адреса з запиту	Користувача не автентифіковано; дозвіл для продовження процесу передачі даних не надано; повернено повідомлення про ймовірну компрометацію користувача	Відсутня
S-M-3	Модуль верифікації/генерація приватного та публічного ключа	Коректні логін, пароль, ім’я користувача	Персональні дані користувача перевірено успішно; пару приватного та публічного ключа згенеровано успішно	Відсутня
S-M-4	Модуль верифікації/генерація приватного та публічного ключа	Некоректні логін, пароль, ім’я користувача	Персональні дані користувача перевірено з помилкою; пару приватного та публічного ключа не згенеровано; повернено повідомлення про невірно введені персональні дані	Відсутня

Таким чином, в процесі тестування програмної системи було обрано метод тестування, а також технологію реалізації даного методу тестування. Згідно з обраним методом тестування було розроблено тестові сценарії та реалізовано безпосереднє тестування ПС, відповідно до розроблених тест-кейсів.

Базуючись на отриманих та зведених результатах тестування, можна зробити висновок, що розроблена програмна система є надійною та відмовостійкою, а заявлений функціонал розроблено коректно, згідно зі сформованими вимогами до реалізованої системи.

4.3 Оцінка ефективності моделей та методів вирішення задач

Для того, щоб отримати висновок стосовно ефективності розробленої програмної системи, необхідно провести теоретичну та практичну оцінку реалізованих методів.

Розглянемо детальніше усі зміни, забезпечені реалізацією компонентів системи.

1 Розбиття даних на окремі транзакції. Завдяки зміні традиційного процесу передачі даних шляхом розбиття даних на окремі транзакції безпека та якість передачі даних значно підвищується. У взаємодії з іншими методами удосконалення передачі даних, розбиття даних на окремі транзакції дозволяє відслідковувати підозрілі зміни в адресах користувачів, за якими передаються дані та швидше відреагувати на ймовірне компрометування отримувача. Завдяки тому, що дані надходять отримувачу окремими транзакціями, у разі перехоплення однієї, чи навіть декількох, транзакцій, зловмисники не зможуть отримати повноцінний доступ до передаваних даних. В той момент, коли дані буде перехоплено, це буде відслідковано, а процес передачі даних буде зупинено. Таким чином, даний метод зробить перехоплення даних зловмисниками неефективним.

2 Повторювана автентифікація відправника та отримувача. Даний метод дозволятиме серверу автентифікувати та валідувати IP-адресу або будь-які інші персональні дані як отримувача, так і відправника. Це дозволить швидко відреагувати

на будь-яку спробу зловмисників перехопити передачу даних та спробу отримувати дані користувача на свій пристрій. При виявленні такої загрози процес передачі даних буде зупинено, відправника сповіщено про те, що отримувача було скомпрометовано, а дані останньої транзакції - перехоплено.

3 Повторювана верифікація отримувача. Повторювана верифікація отримувача дозволяє доповнити інші розроблені методи удосконалення передачі даних шляхом верифікації отримувача при кожній передачі даних. Такий метод дозволить виявити зловмисні дії інших користувачів мережі навіть у тому випадку, якщо це не буде виявлено на етапі автентифікації. Невірно підписане повідомлення отримувачем означатиме для відправника, що отримувача було скомпрометовано, а пара публічного та/або приватного ключів невірна, оскільки зловмисники не змогли отримати чи відгадати його з першої спроби.

Під час практичної апробації результатів, отриманих в процесі розробки програмної системи, було виявлено, що впровадження трьох розроблених методів дозволило значно покращити якість та безпеку процесу передачі даних в мережі Інтернет. Кожен з впроваджених методів є важливою складовою розробленого алгоритму, оскільки вони доповнюють та компенсують недоліки одне одного.

4.4 Висновки

Отже, під час виконання розділу визначено та описано структуру і призначення програмних модулів розроблюваної системи, а також описано взаємозв’язок між ними і необхідність реалізації кожного з визначених програмних модулів системи.

В результаті опису програмних модулів було проведено безпосередню програмну реалізацію кожного модуля обох компонентів програмної системи: як клієнтського, так і серверного додатку. Було надано уривки коду реалізації деяких найважливіших функцій розроблюваних програмних модулів.

В процесі реалізації програмних модулів в тому числі було реалізовано моделі бази даних як для серверного, так і для клієнтського додатку, шляхом використання методу керування базами даних «codefirst».

Окрему увагу під час розробки програмних модулів було приділено реалізації методів, які необхідні для імплементації удосконаленого методу передачі даних у мережі Інтернет.

Після реалізації усіх необхідних модулів системи було проведено вибір методу тестування ПС та технології, за допомогою якої обраний метод тестування можна реалізувати програмно.

На основі розроблених тестових сценаріїв було проведено тестування та аналіз результатів тестування. Оцінивши результати тестування було зроблено висновок про високу надійність та відмовостійкість розробленої програмної системи.

Наостанок було проведено оцінку ефективності впроваджених моделей та методів, обґрунтовано їхню ефективність та необхідність реалізації саме в такому вигляді, який було запропоновано удосконаленим алгоритмом передачі даних в процесі написання кваліфікаційної роботи.

ВИСНОВКИ

В результаті виконання кваліфікаційної роботи було розроблено програмну систему для користування удосконаленим методом передачі даних в мережі Інтернет.

В ході написання першого розділу кваліфікаційної роботи було досліджено предметну область, останні дослідження у сфері передачі даних, проаналізовано наявні методи та засоби вдосконалення передачі даних, а також сформовано задачі на кваліфікаційну роботу.

В ході написання другого розділу було розглянуто запропоновані концепції, моделі та методи вирішення поставлених задач. Запропоновані концепції, моделі та методи було описано та обґрунтовано необхідність їх застосування.

В ході написання третього розділу кваліфікаційної роботи було проведено дослідження та аналіз вимог до програмного засобу. Пізніше було розроблено структуру програмного засобу та спроєктовано структуру даних. Також було проаналізовано та обрано засоби, необхідні для програмної реалізації розроблених методів та моделей.

В ході написання четвертого розділу кваліфікаційної роботи було описано структуру та призначення модулів програми, а також взаємозв’язок між ними. Згодом було проведено безпосередню програмну розробку спроєктованих модулів, зокрема реалізовано методи, необхідні для імплементації удосконаленого методу передачі даних в мережі Інтернет. Після проведення програмної реалізації було проведено вибір методів та тестування, розроблено тестові сценарії та проведено тестування програмної системи з використанням обраного методу на основі розроблених тестових сценаріїв.

На основі отриманих результатів було проведено емпіричне дослідження для обґрунтування працездатності та функціональної придатності програмної системи.

Розроблений програмний продукт дозволяє підвищити безпеку та надійність передачі даних в мережі Інтернет та покращити досвід користування мережею Інтернет для користувачів в цілому. Завдяки імплементованому удосконаленому методу передачі даних було досягнуто розбиття даних на окремі транзакції, що значно

підвищило безпеку передачі даних в контексті забезпечення неможливості отримати цілісні дані зловмисниками. Разом з цим, було досягнуто повторної автентифікації та верифікації користувачів під час процесу передачі даних, що разом з розбиттям даних на транзакції дозволило повноцінно забезпечити дані від повноцінного потрапляння в руки зловмисників, а також дозволило одразу помічати підозрілі дії зловмисників, такі як компрометування одного з користувачів, і припиняти процес передачі даних з метою запобігання потрапляння більшої кількості транзакцій даних до зловмисників.

Результати перевірки та тестування програмної системи підтверджують її працездатність. Тому її рекомендовано інтегрувати компаніям, які зацікавлені у підвищенні безпеки передачі даних в мережі Інтернет.

Таким чином, у результаті виконання даної кваліфікаційної роботи було проведено системний аналіз у сфері передачі даних в мережі Інтернет. На основі отриманих даних було спроєктовано та імплементовано концепцію розділеної передачі даних з повторною автентифікацією та верифікацією користувачів у вигляді повнофункціональної програмної системи.

Згідно з результатами досліджень було опубліковано тези доповіді на Всеукраїнській науково-практичній конференції [19].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Історія Інтернету. URL: https://ucloud.ua/istoriya-internetu-vid-arpanet-do-sogodni/ (дата звернення: 06.09.2023).

2. 2,6 мільярда пристроїв в 5G, 75% трафіка буде витрачатися на відео. URL: https://www.epravda.com.ua/news/2019/11/25/654105/ (дата звернення: 07.09.2023).

3. Понад 60% населення світу використовує соціальні мережі. URL: https://interfax.com.ua/news/telecom/924195.html (дата звернення: 09.09.2023).

4. Підраховано кількість користувачів смартфонів у світі. URL: https://lenta.ua/pidrahovano-kilkist-koristuvachiv-smartfoniv-u-sviti-146749/ (дата звернення: 13.09.2023).

5. Візуалізація мобільного трафіку у світі. URL: https://www.proidei.com/mobile-traffic-2021-2812/ (дата звернення: 19.09.2023).

6. 63% людей зараз онлайн. Великий звіт Digital 2022 про користувачів інтернету. URL: https://ain.ua/2022/04/30/zvit-digital-2022/ (дата звернення: 20.09.2023).

7. Близько 78% українців щодня користуються інтернетом. URL: https://www.ukrinform.ua/rubric-technology/3497671-blizko-78-ukrainciv-sodna-koristuutsa-internetom.html (дата звернення: 25.09.2023).

8. Середня вартість збитків від витоку даних від кібератак у світі в 2022 році зросла до $4,4 млн. URL: https://forinsurer.com/news/23/02/20/42397 (дата звернення: 03.10.2023).

9. Що таке кібератака? URL: https://www.microsoft.com/uk-ua/security/business/security-101/what-is-a-cyberattack (дата звернення: 14.10.2023).

10. Найгучніші кібератаки в Україні та світі і способи захисту від них. URL: https://psm7.com/uk/security/kibervijna-ce-suchasna-realnist-najguchnishi-kiberataki-v-ukraïni-ta-sviti-i-sposobi-zaxistu-vid-nix.html (дата звернення: 15.10.2023).

11. Вимоги до захисту інформації в умовах військового стану. URL: https://www.kmu.gov.ua/news/vymohy-do-zakhystu-informatsii-v-informatsiinykh-

systemakh-u-voiennyi-chas-roziasnennia-derzhspetszviazku (дата звернення: 16.10.2023).

12. Вимоги до захисту інформації в інформаційних системах у воєнний час. URL: https://cip.gov.ua/services/cm/api/attachment/download?id=50943 (дата звернення: 20.10.2023).

13. У 2022 році в Україні зареєстрували 2194 кіберінциденти. URL: https://suspilne.media/397220-u-2022-roci-v-ukraini-zareestruvali-2194-kiberincidenti-derzspeczvazku/ (дата звернення: 23.10.2023).

14. У 2022 році кількість зареєстрованих кіберінцидентів виросла майже втричі. URL: https://cip.gov.ua/ua/news/u-2022-roci-kilkist-zareyestrovanikh-kiberincidentiv-virosla-maizhe-vtrichi-zvit (дата звернення: 30.10.2023).

15. Протоколи передавання даних: їх типи та особливості. URL: https://highload.today/uk/protokoli-peredavannya-danih-yih-tipi-ta-osoblivosti/ (дата звернення: 03.11.2023).

16. Шифрування даних: все, про що ви повинні знати, щоб захистити дані. URL: https://www.sim-networks.com/ukr/blog/data-encryption-best-practices (дата звернення: 08.11.2023).

17. Введення в криптографію: симетричне шифрування. URL: https://markup-ua.com/vvedennya-v-kriptografiyu-simetrichne-shifruvannya/ (дата звернення: 14.11.2023).

18. Що таке асиметрична криптографія? URL: https://portalcripto.com.br/uk/dicionario/o-que-e-criptografia-assimetrica/ (дата звернення: 16.11.2023).

19. Слутяк Є.І., Радельчук Г.І., Балицький Б.І. Удосконалення передачі даних у мережі Інтернет з використанням алгоритму верифікації повідомлень // Актуальні проблеми комп’ютерних наук АПКН-2023: Збірник наукових праць за матеріалами XV Всеукраїнської науково-практичної конференції, м. Хмельницький, 17-18 листопада 2023 р. Хмельницький, 2023. С. 274–277.

ДОДАТОК А

(обов’язковий)

ЗАГАЛЬНА ДІАГРАМА ВАРІАНТІВ ВИКОРИСТАННЯ

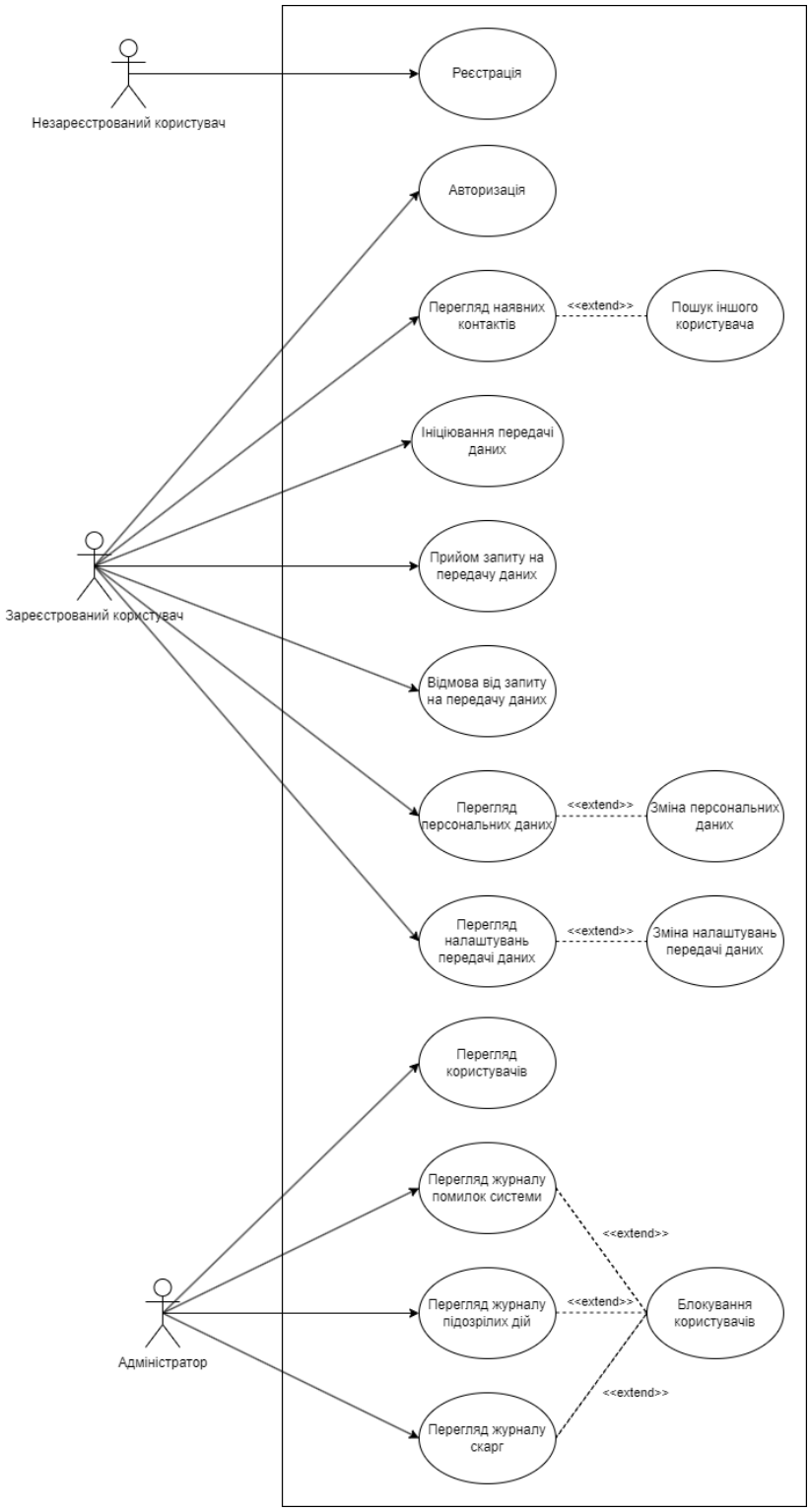


Рисунок А.1 – Діаграма варіантів використання

ДОДАТОК Б

(обов’язковий)

МОДЕЛІ СТРУКТУРИ ДАНИХ

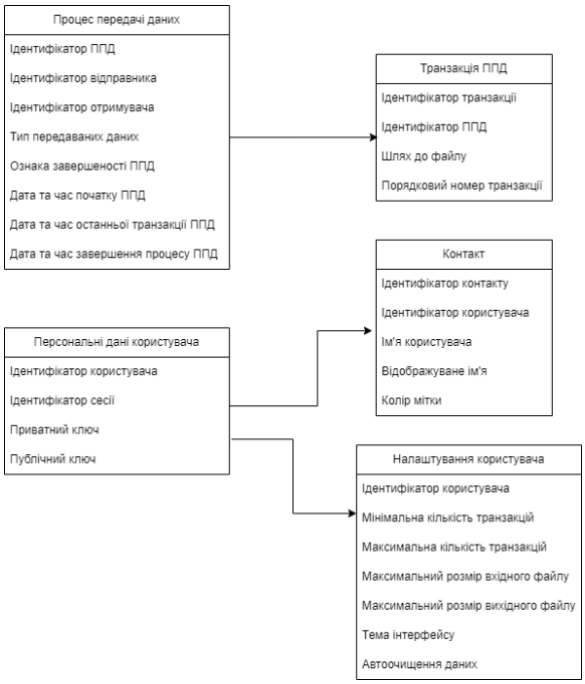


Рисунок Б.1 – Схема БД клієнтського додатку

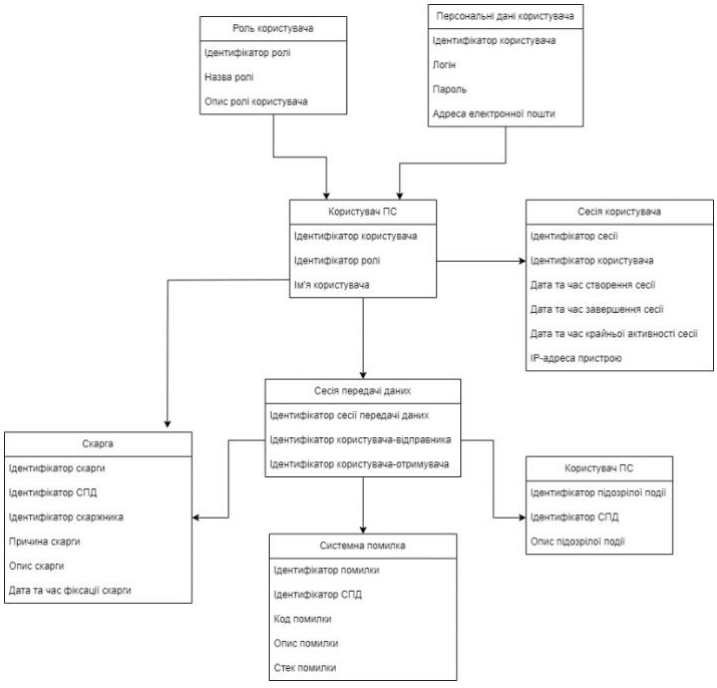


Рисунок Б.2 – Схема БД серверного додатку

ДОДАТОК В

(обов’язковий)

КОПІЯ НАУКОВОЇ ПУБЛІКАЦІЇ

Міністерство освіти і науки України
Хмельницький національний університет

ЗБІРНИК НАУКОВИХ ПРАЦЬ

за матеріалами XV Всеукраїнської науково-практичної конференції
«Актуальні проблеми комп'ютерних наук АПКН-2023»

17-18 листопада 2023

Хмельницький 2023

УДК 004:37:001:62

Збірник наукових праць за матеріалами XV Всеукраїнської науково-практичної конференції «Актуальні проблеми комп'ютерних наук АПКН-2023». Хмельницький. 2023. 345с.

У збірнику наукових праць подані перспективні практичні розробки аспірантів, студентів та здобувачів в області сучасних інформаційних технологій. Розглянуто актуальні проблеми комп'ютерних наук, комп'ютерної інженерії, прикладної математики й інженерії програмного забезпечення, приведено ряд робіт по впровадженню інформаційних технологій у виробництво та управління. Висвітлено перспективні розробки сучасних систем пошуку, обробки й захисту інформації, медійних та комунікаційних системи.

УДК 004:37:001:62

Матеріали конференції відтворені з авторських оригіналів. При макетуванні можливі незначні зміни компоновки контенту авторських оригіналів.

Участь у конференції та складові всіх її етапів (розгляд праць, макетування, публікація збірника наукових праць та видача сертифікатів) є безкоштовними для всіх учасників. Оргкомітет конференції висловлює подяку учасникам конференції та сподівається на подальшу співпрацю.

З питань проведення конференції та подальшого обміну інформацією звертатись на e-mail конференції: *apkt.khnu@gmail.com*

УДК 004.4

Слутяк Є.І., Радельчук Г.І., Балицький Б.І.

Хмельницький національний університет

УДОСКОНАЛЕННЯ ПЕРЕДАЧІ ДАНИХ У МЕРЕЖІ ІНТЕРНЕТ З ВИКОРИСТАННЯМ АЛГОРИТМУ ВЕРИФІКАЦІЇ ПОВІДОМЛЕНЬ

Проведено аналіз предметної області, досліджено наявні реалізації протоколів передачі даних та алгоритмів шифрування. Удосконалено передачу даних, що знизить ймовірність перехоплення даних користувачів під час їх передачі у мережі Інтернет. Запропонована удосконалена модель вирішує усі проблеми, виявлені під час дослідження.

An analysis of the subject area was carried out, existing implementations of data transfer protocols and encryption algorithms were investigated. Data transmission has been improved, which will reduce the probability of interception of user data during its transmission over the Internet. The proposed improved model solves all the problems identified during the research.

Останні дослідження показують, що з кожним роком кількість користувачів мережі Інтернет зростає. Відповідно, разом з цим зростає трафік, який споживається цими користувачами у мережі. Так, відповідно до звіту Digital 2022 Global Statshot [1] близько 63% населення світу користуються мережею Інтернет станом на кінець 2022 року. Також, згідно зі зведеною статистикою приблизний щоденний обсяг інтернет-трафіку протягом останніх десяти років зріс з десятків ексабайт даних до сотень ексабайт даних.

Таке зростання популярності та поширеності мережі Інтернет призвів до значного зростання кіберзловмисників. Відповідно до звіту про роботу системи виявлення вразливостей та реагування на кіберінциденти [2] станом на кінець 2022 року кількість кіберзлочинів зросла у 2.8 разів. Згідно з даними Держспецзв'язку [3] близько 90% кібератак у 2022 році було здійснено хакерами з території Білорусії та Російської Федерації.

Саме тому проблема безпеки передачі даних є актуальною та потребує дослідження, аналізу та знаходження нових методологій та методів вирішення проблем безпеки передачі даних у мережі Інтернет.

Метою роботи є удосконалення методу передачі даних у мережі Інтернет з використанням алгоритму верифікації повідомлень.

Класичний метод передачі даних складається з двох частин, а саме:

– протокол передачі даних;

– алгоритм шифрування даних.

Загальна модель (рисунок 1), яка базується на класичному методі передачі даних, складається з двох частин: клієнта та сервера. Також класична модель передбачає однобічний зв'язок між складовими мережі.

Рисунок 1 – Класична модель передачі даних

Цей метод, з його розподілом та рівнями відповідальності, має значну кількість недоліків. Зокрема, дана модель не дозволяє жодним чином проводити автентифікацію або верифікацію користувачів мережі. Тому такий підхід до передачі даних дозволяє зловмисникам перехоплювати (або навіть компрометувати) одного з користувачів та безперешкодно отримувати дані, які відправник чи кілька відправників надаватимуть отримувачу, не підозрюючи, що його було скомпрометовано та усі дані надходять зловмисникам.

Покращений метод передачі даних передбачає ще одну складову під час передачі даних – шар автентифікації та верифікації користувачів та даних (рисунок 2).

Рисунок 2 – Покращена модель передачі даних

Покращена модель містить стільки ж складових, скільки і класична, проте різницею між ними є те, що покращена модель передбачає двосторонній зв'язок під час передачі даних. Ця модель передбачає декілька кроків передачі даних.

Перший крок передачі даних (рисунок 3) передбачає аутентифікацію відправника та перевірку наявності і стабільності з'єднання з отримувачем.

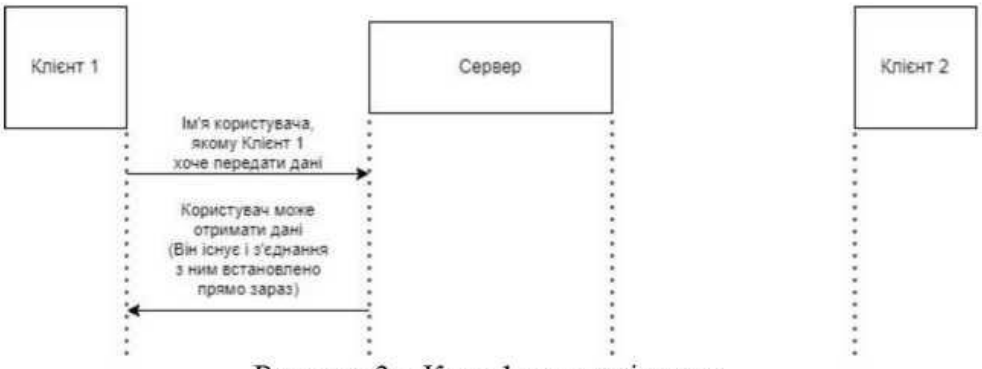


Рисунок 3 – Крок 1 передачі даних

Рисунок 4 – Крок 2 передачі даних

Таким чином, вирішується проблема «сліпої» передачі даних. Завдяки цьому етапу передачі даних сервер-посередник знатиме, що відправник має можливість надсилати дані, а також визначить, чи може отримувач безпечно отримати дані та чи не був отримувач скомпрометований.

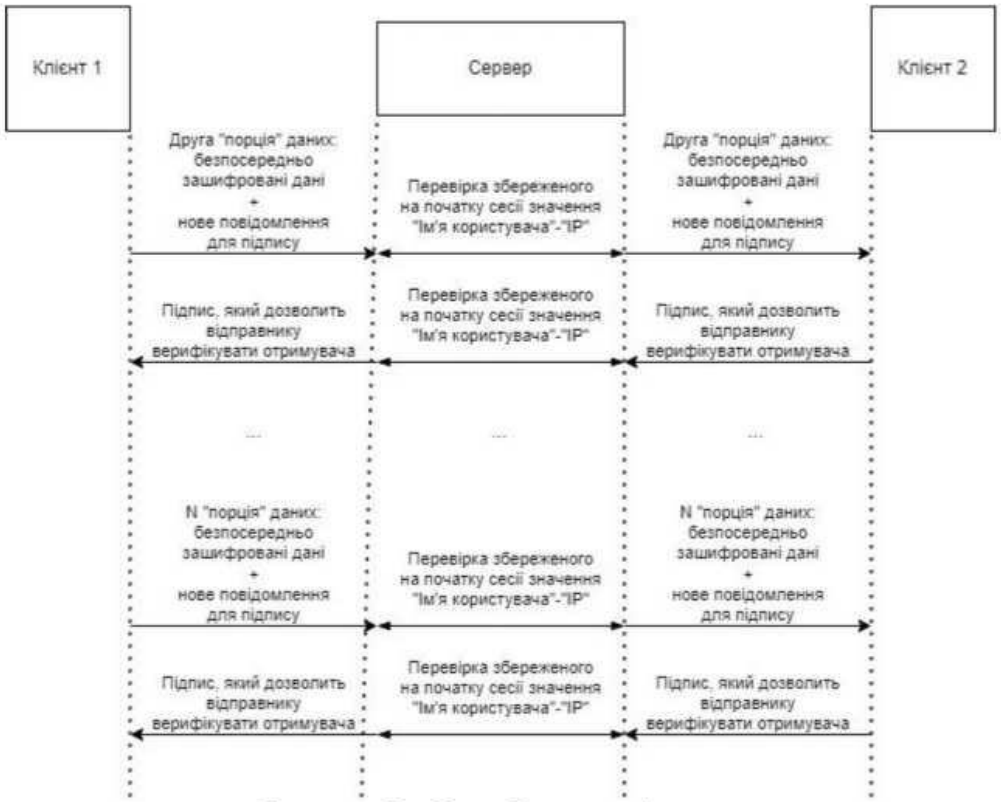


Рисунок 5 – Крок 3 передачі даних

Наступна модель передбачає, що дані передаватимуться отримувачу частинами. Під час виконання другого кроку передачі даних (рисунок 4) виконується передача першої частини даних. Під час передачі даних виконується реалізація алгоритму верифікації повідомлень. Отримувачу надсилається згенероване повідомлення, після чого він його підписує за допомогою власного ключа і повертає відправнику. Під час передачі даних сервер виконує важливу роль, проводячи автентифікацію обох учасників передачі даних.

Під час виконання третього кроку передачі даних (рисунок 5) відбувається повторна передача частин даних, включно з останньою частиною даних. Таким чином, це дає можливість не дати доступу зловмисникам до повного набору даних. Під час передачі кожної з частин повідомлень відбувається верифікація отримувача.

На завершальному етапі передачі даних відбувається відправка повідомлення отримувачу про завершення передачі даних (рисунок 6).

Рисунок 6 – Крок 4 передачі даних

Отже, удосконалений метод передачі даних через мережу Інтернет вирішує виявлені під час дослідження недоліки та суттєво підвищує захищеність процесу передачі інформації.

Подальші дослідження спрямовані на реалізацію розробленого методу удосконалення передачі даних протоколом TCP (Transmission Control Protocol) з використанням алгоритму шифрування Діффі-Хелмана.

Перелік посилань

1. Digital 2022: October global statshot report. URL: https://datareportal.com/reports/digital-2022-october-global-statshot (дата звернення: 24.10.2023).
2. Звіт про роботу системи виявлення вразливостей і реагування на кіберінциденти та кібератаки, URL: https://cip.gov.ua/services/cm/api/attachment/download?id=50943 (дата звернення: 27.10.2023).
3. У 2022 році кількість зареєстрованих кіберінцидентів виросла майже втричі – звіт. URL: https://cip.gov.ua/ua/news/u-2022-roci-kilkist-zareyestrovanikh-kiberincidentiv-virosla-maizhe-vtrichi-zvit (дата звернення: 29.10.2023).

ДОДАТОК Г

(обов’язковий)

ПРЕЗЕНТАЦІЙНІ МАТЕРІАЛИ

Кафедра інженерії програмного забезпечення

Метод удосконалення передачі даних протоколом TCP за допомогою алгоритму верифікації повідомлень та алгоритму Діффі-Хелмана

Виконав:
Студент II курсу, група ІПЗм-22-1
Слутяк Євгеній Іванович

Керівник:
Доцент, кандидат технічних наук
Радельчук Галина Іванівна

Актуальність роботи

- Актуальність роботи полягає у тому, що якість, надійність та безпека передачі даних у мережі Інтернет постійно знаходиться під загрозою. Методи нанесення шкоди даним користувачів та корпорацій постійно розвиваються. Саме тому необхідно розвивати методи для надання вищої якості, надійності та безпеки передачі даних, щоб мати змогу протидіяти зловмисникам.

Об’єкт, предмет і мета дослідження

- Об’єктом дослідження є процес передачі даних у мережі Інтернет.
- Предметом дослідження є методи, алгоритми та практики покращення передачі даних у мережі Інтернет.
- Мета дослідження – удосконалення традиційних методів передачі даних у мережі Інтернет, що понизить ймовірність перехоплення даних користувача під час їх передачі або понизить ймовірність їх розшифрування у разі перехоплення.

Задачі дослідження

Відповідно до мети дослідження, слід вирішити наступні задачі:

- провести аналіз сфери безпеки передачі даних у мережі Інтернет;
- дослідити сучасні методи та практики підвищення безпеки передачі даних;
- удосконалити наявні методи та практики для повного або часткового вирішення виявлених проблем;
- на основі удосконаленого методу передачі даних виконати проєктування програмного забезпечення для реалізації цього методу;
- виконати програмну реалізацію спроєктованого програмного забезпечення;
- виконати тестування та перевірку отриманих результатів;
- проаналізувати результати проведеного дослідження та сформувати висновок щодо доцільності впровадження удосконаленого методу передачі даних у мережі Інтернет.

Аналіз стану проблеми і інших рішень

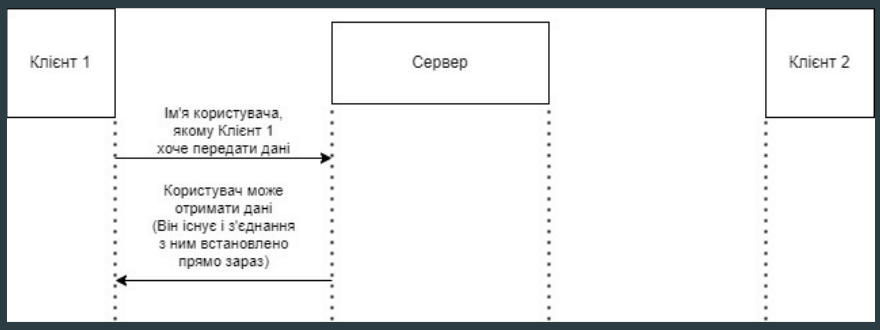
- У ході дослідження наявних методів та моделей передачі даних у мережі Інтернет було виділено основну традиційну методологію передачі даних та відносно нову, модернізовану, модель передачі даних.
- Традиційна модель є простою, з трьома вузлами мережі, зв’язок між якими відбувається односторонньо. Такий підхід є легким в реалізації та впровадженні, не вимагає значних витрат у ресурсах систем як клієнтів, так і сервера, а також є досить швидким, оскільки передача даних відбувається за одну транзакцію, без потреби додаткових перевірок, верифікацій, аутентифікацій чи розділення даних на окремі частини.
- Новіша модернізована модель передбачає верифікацію користувача, проте ця верифікація відбувається лише один раз перед початком передачі даних, що не може гарантувати безпеки під час процесу передачі даних.

Розроблені методи

- Покращена модель, так само як і традиційна, передбачає використання трьох основних вузлів мережі задіяних у процесі передачі даних: два клієнти, між якими відбуватиметься передача даних та сервер. Проте роль кожного з вузлів є значно ширшою, аніж у традиційній моделі. Відмінністю між загальними традиційною та покращеною моделями є двосторонність цього зв’язку, що дозволятиме реалізувати вирішення для кожної з досліджених проблем.

Розроблені методи

- Перший етап передачі даних покращеною моделлю передбачає автентифікацію та попередню перевірку активності та стабільності з’єднання з отримувачем.

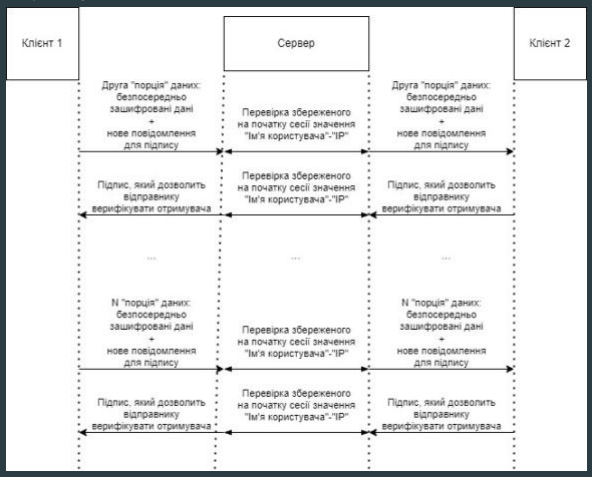


Розроблені методи

- Другий етап передбачає початок процесу передачі даних, зокрема верифікацію та автентифікацію отримувача, передачу інформації про передавані дані.

Розроблені методи

- Третій етап передбачає безпосередню передачу даних, яка супроводжується верифікацією та автентифікацією отримувача впродовж усього процесу передачі даних.

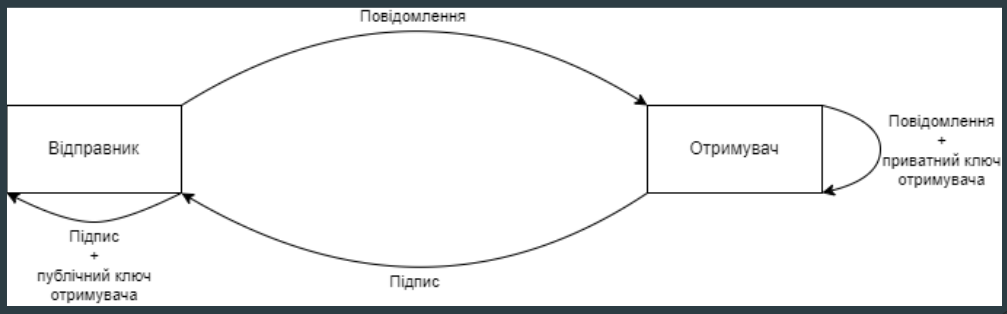


Розроблені методи

- Четвертий етап передбачає завершення процесу передачі даних.

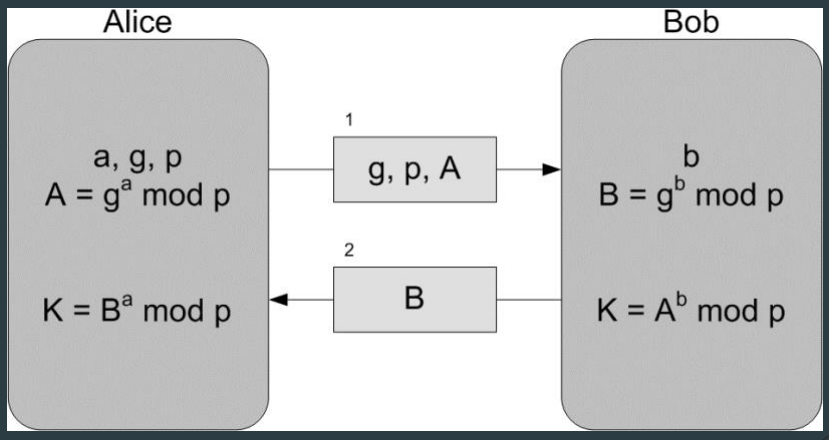
Алгоритми вирішення задач

- Алгоритм верифікації повідомлень



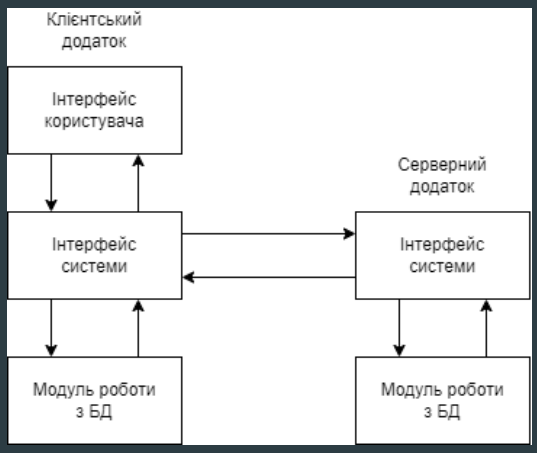
Алгоритми вирішення задач

- Алгоритм Діффі-Хелмана



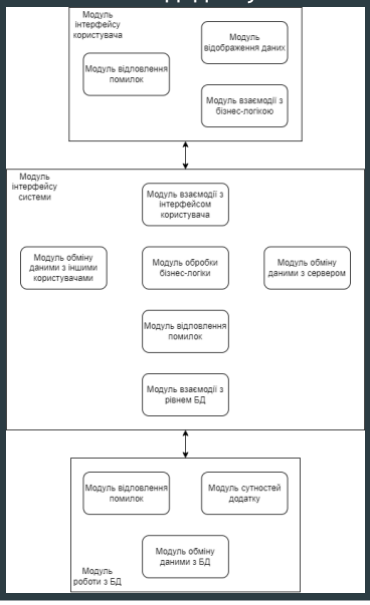
Структура програмної системи

- Взаємодія компонентів програмної системи



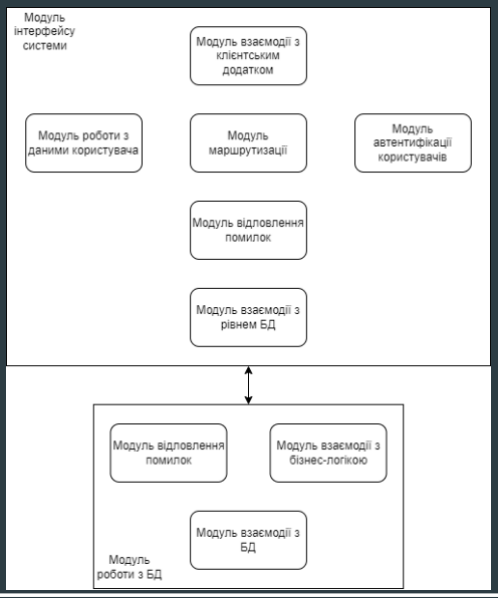
Структура програмної системи

- Взаємодія елементів клієнтського додатку



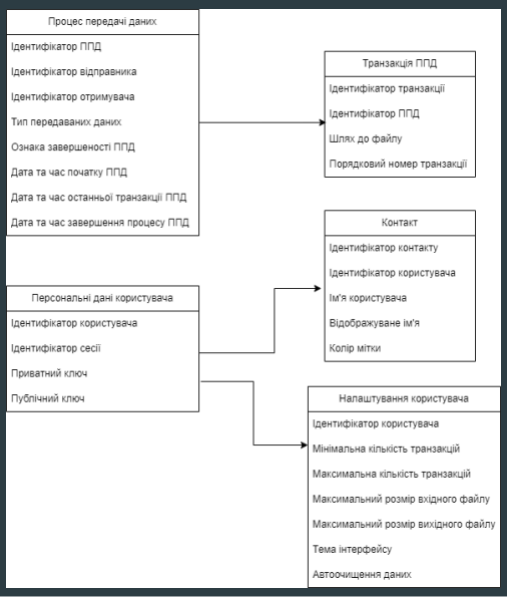
Структура програмної системи

- Взаємодія елементів серверного додатку



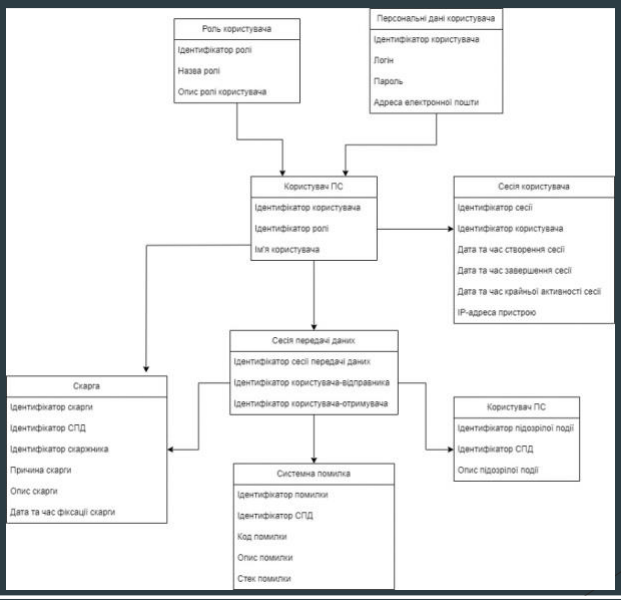
Моделі структури даних

- Схема бази даних клієнтського додатку



Моделі структури даних

- Схема бази даних серверного додатку



Отримані результати

Розроблена програмна система забезпечує:

- покращену передачу даних протоколом TCP;
- покращений спосіб автентифікації відправника та отримувача даних;
- покращений спосіб верифікації отримувача протягом усього процесу передачі даних;
- покращений спосіб передачі даних шляхом розбиття даних на окремі транзакції.

Наукова новизна

- Удосконалено метод передачі даних у мережі Інтернет, який відрізняється розширеними методиками повторюваної автентифікації, повторюваної верифікації та розбиття даних на окремі транзакції, що забезпечує підвищення якості, надійності та безпеки передачі даних у мережі Інтернет.

Практичне значення

- Практична цінність отриманих результатів полягає в успішному дослідженні, виділенні основних проблем передачі даних у мережі Інтернет та покращенні наявних методів, моделей і рішень.
- Завдяки розробці даного рішення було підвищено якість, надійність та безпеку передачі даних у мережі Інтернет, тому удосконалений метод та розроблена програмна система має високі конкурентні шанси на ринку.

Наукова публікація

- Слутяк Є.І., Радельчук Г.І., Балицький Б.І. Удосконалення передачі даних у мережі Інтернет з використанням алгоритму верифікації повідомлень // Актуальні проблеми комп'ютерних наук АПКН-2023: Збірник наукових праць за матеріалами XV Всеукраїнської науково-практичної конференції, м. Хмельницький, 17-18 листопада 2023 р. Хмельницький, 2023. С. 274-277.

Висновки та рекомендації

- В результаті виконання кваліфікаційної роботи було проведено дослідження у сфері передачі даних в мережі Інтернет, визначено недоліки наявних рішень. На основі визначених недоліків було запропоновано методи та рішення, які дозволять підвищити безпеку передачі даних. Базуючись на отриманих методах спроектовано та реалізовано програмну систему, яка довела роботу концепції методу удосконалення передачі даних протоколу TCP з використанням алгоритму верифікації повідомлень та алгоритму Діффі-Хелмана.
- Результати перевірки та тестування програмної системи підтверджують її працездатність. Тому її рекомендовано інтегрувати компаніям, які зацікавлені у підвищенні безпеки передачі даних в мережі Інтернет.

Завідувачу кафедри інженерії програмного
забезпечення проф. Леоніду БЕДРАТЮКУ
здобувача вищої освіти
Євгенія СЛУТЯКА
факультет ІТ, 2 курс, група ІПЗм-22-1

ЗАЯВА

З правилами чинного Положення про систему забезпечення академічної доброчесності в Хмельницькому національному університеті, згідно з яким виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту і застосування заходів дисциплінарної та академічної відповідальності, ознайомлений. Про використання програмно-технічних засобів для перевірки кваліфікаційних робіт здобувачів вищої освіти на плагіат оповіщений та надаю свою згоду на обробку й збереження університетом моєї роботи в інституційному репозитарії університету.

Також надаю університету право на передачу моєї роботи для обробки та збереження в базах даних програмно-технічних засобів (Unicheck та Anti-Plagiarism) та використання роботи для виявлення плагіату в інших роботах, які перевіряються програмно-технічними засобами та користувачами, що мають доступ до цих програмно-технічних засобів, виключно в обмежених цілях для виявлення плагіату в текстах робіт.

Робота для перевірки університетом надається в друкованому та електронному варіанті. Електронна версія моєї роботи збігається (ідентична) з друкованою.

08.12.2023
дата

підпис

Sat Dec 09 11:23:07 EET 2023, Форкун Юрій Вікторович, Хмельницький національний університет, ХНУ

Anti-Plagiarism v-15.257

Максимальне співпадіння з одним документом 2.0%

Словники перевірки: en_US, ru_RU, ua_UA. **Помилок в документах: 8%**

	Документ		Сумарний збіг по Базі Даних	
ID: 122256 Назва: Метод удосконалення передачі даних протоколом TCP за допомогою алгоритму верифікації повідомлень та алгоритму Діффі-Хелмана Додано в БД: 2023-12-09 Автора: Слуутяк Євгеній Керівники: Радельчук Галина Консультанти: Опоненти:	Символи	Лексеми	Символи	Лексеми
	128012	935	3700 (3%)	50 (5%)

Джерело плагіату

ID	Опис	Наявність плагіату в документі	
		Символи	Лексеми

UNICHECK
by Turnitin

Ім'я користувача:
ІПЗ

Дата перевірки:
09.12.2023 12:19:40 EET

Дата звіту:
09.12.2023 12:29:40 EET

ID перевірки:
1015986740

Тип перевірки:
Doc vs Internet + Library

ID користувача:
100012953

Назва документа: Слутяк Антиплагіат

Кількість сторінок: 92 Кількість слів: 18090 Кількість символів: 146016 Розмір файлу: 752.50 KB ID файлу: 1015667908

5.41%
Схожість

Найбільша схожість: 1.91% з джерелом з Бібліотеки (ID файлу: 1005677564)

4.95% Джерела з Інтернету | 607 ..... Сторінка 94

2.76% Джерела з Бібліотеки | 147 ..... Сторінка 97

0.18% Цитат

Цитати | 2 ..... Сторінка 98

Не знайдено жодних посилань

0%
Вилучень

Немає вилучених джерел

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

РЕЦЕНЗІЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ
освітнього ступеня «магістр»

Магістр Слутяк Євгеній Іванович

Тема Метод удосконалення передачі даних протоколом TCP за допомогою алгоритму верифікації повідомлень та алгоритму Діффі-Хелмана

Спеціальність 121 «Інженерія програмного забезпечення»

Обсяг кваліфікаційної роботи:

Кількість сторінок кваліфікаційної роботи 94.

1.Короткий зміст роботи та прийнятих рішень У кваліфікаційній роботі виконано системний аналіз у сфері передачі даних в мережі Інтернет та визначено проблеми і недоліки наявної області. На основі проведеного аналізу було покращено наявний метод передачі даних та реалізовано його програмно. Удосконалений метод підвищує безпеку передачі даних у мережі Інтернет, що дає змогу передавати дані, не турбуючись про їх перехоплення та використання у злочинних цілях зловмисниками.

2. Висновок про відповідність роботи поставленому завданню Кваліфікаційна робота освітнього ступеня «магістр» у повній мірі відповідає поставленому завданню як у теоретичній, так і в практичній її частині.

3. Характеристика виконання кожного розділу роботи, ступінь використання останніх досягнень науки і техніки і передових методів роботи: У вступі обґрунтовується актуальність теми роботи, формулюються мета та завдання дослідження, описується наукова новизна та практична цінність отриманих результатів. У першому розділі охарактеризовано структуру предметної області та існуючих методів і засобів безпечної передачі даних у мережі Інтернет, визначені метологічні підходи до вирішення задачі, виконана розгорнута постановка задачі. У другому розділі досліджено методи і способи вирішення поставлених задач. Традиційні підходи до передачі даних, досліджені на етапі аналізу, були вдосконалені шляхом комбінації традиційних підходів, їх переосмислення та впровадження нових. У третьому розділі обґрунтовано проектні рішення, що дають змогу реалізувати технічні вимоги, забезпечити сумісність та взаємодію різних компонентів системи. У четвертому розділі розглянуто питання, що стосуються реалізації програмного засобу на основі прийнятих рішень, а також її технічні та технологічні характеристики. Також проведено емпіричне дослідження, спрямоване на доведення працездатності розробленого програмного засобу та його функціональної придатності. Обґрунтована ефективність удосконаленого методу передачі даних в мережі Інтернет та розроблено рекомендації з його застосування.

4. Позитивні сторони роботи Кваліфікаційна робота містить низку інноваційних рішень, зокрема, було доведено доцільність удосконалення методу та засобів передачі даних у мережі Інтернет. Запропоновано покращити спосіб верифікації отримувача передаваних даних та здійснено методику одночасної авторизації при верифікації

саме запропонованим способом. Удосконалений алгоритм показав свою ефективність під час апробації.

5. Негативні сторони роботи У роботі використано відразу декілька алгоритмів та засобів покращення передачі даних для вдосконалення розроблюваного методу, через що виникає питання: можливо краще було б вибрати менше алгоритмів, проте зробити це детальніше та з різних аспектів, ніж розфокусовувати дослідження до розв'язання більш широкого спектру проблем.

6. Оцінка графічного оформлення та пояснювальної записки роботи Графічне оформлення виконане відповідно до теми кваліфікаційної роботи з дотриманням вимог стандартів. Пояснювальна записка відповідає вимогам стандартів до її оформлення.

7. Відгук про роботу в цілому В цілому кваліфікаційна робота заслуговує позитивної оцінки. Весь матеріал роботи структурований, чіткий та послідовний. Усі розділи роботи є послідовними та логічними, що дозволяє чітко розуміти викладений матеріал у рамках тематики кваліфікаційної роботи. Графічний матеріал дозволяє наочно побачити доцільність та ефективність рішень, які були прийняті для вирішення поставленої задачі.

8. Інші зауваження

9. Оцінка кваліфікаційної роботи Розглянувши позитивні та негативні сторони представленої кваліфікаційної роботи, можна зробити висновок, що вона заслуговує оцінки «добре».

РЕЦЕНЗЕНТ (прізвище, ім'я, по-батькові, посада, місце роботи) Мартинюк Валерій Володимирович, зав. кафедри автоматизації, комп'ютерно-інтегрованих технологій та робототехніки

«04» грудня 2023 р. (підпис)

РІШЕННЯ ЕКСПЕРНОЇ КОМІСІЇ
КАФЕДРИ ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ
ПРО ДОПУСК КВАЛІФІКАЦІЙНОЇ РОБОТИ ДО ЗАХИСТУ

Підтверджуємо ознайомлення з результатами звіту/звітів подібності щодо роботи, продукованими програмно-технічним засобом(ами) перевірки текстів на плагіат.
Назва: «Метод удосконалення передачі даних протоколом TCP за допомогою алгоритму верифікації повідомлень та алгоритму Діффі-Хелмана»
Автор: Слутяк Євгеній Іванович
Спеціальність: 121 – Інженерія програмного забезпечення
Освітня програма: Освітньо-професійна програма «Інженерія програмного забезпечення»
Науковий керівник: Радельчук Галина Іванівна, кандидат технічних наук, доцент
Після аналізу звіту подібності зроблено такий висновок:

№	Висновок	Позначка про відповідність
1	Запозичення, виявлені в роботі, є законними і не є плагіатом. Робота приймається до захисту.	**відповідає**
2	Виявлені запозичення не є плагіатом, розміщені у розділах, які не описують безпосередньо авторське дослідження, але кількість цитат перевищує обсяг, виправданий поставленою метою роботи. Робота приймається до захисту, але має бути відкоригована. Відкоригований варіант має бути поданий на кафедру за два дні до захисту, разом із заявою щодо самостійності виконання письмової роботи та ідентичності друкованої й електронної версії роботи.	
3	Виявлені запозичення не є плагіатом, але частково розміщені у розділах, які описують безпосередньо авторське дослідження, а кількість цитат перевищує обсяг, виправданий поставленою метою роботи. В зв'язку з цим мета роботи та поставлені завдання не були досягнені. Робота може бути допущена до захисту (наступного року) після того, як буде відкоригована та доопрацьована і успішно пройде повторну перевірку на академічний плагіат.	
4	Робота містить навмисні текстові спотворення, передбачувані спроби укриття запозичень або інші прояви академічного плагіату. Робота містить фабрикацію або фальсифікацію даних. Робота не допускається до захисту.	
5	Інше:	

Підтвердження:
Запозичення, виявлені у роботі, є законними і не є плагіатом, оскільки:
1) у тексті кваліфікаційної роботи системою перевірки на плагіат Unicheck виявлено схожість з деякими документами у частині загальновживаних обов'язкових словосполучень у стандартних бланках (титульний аркуш), у структурі змісту, у переліку скорочень, у назвах розділів/підрозділів, у назвах публікацій переліку джерел посилання;
2) в якості запозичень системою Unicheck було зафіксовано деякі послідовності вихідного коду, які є стандартними мовними конструкціями програмування та не можуть розглядатися як об'єкт авторських прав і, відповідно, їх порушення;
3) запозичення, виявлені в тексті роботи, є фрагментарними або мають належним чином оформленні посилання;
Максимальний обсяг запозичень, визначений системою Anti-Plagiarism, складає **2.0%.** Обсяг запозичень, визначений системою Unicheck виявлення збігів ідентичності/схожості, складає **5.41%** і адресується до **607** джерел з інтернету і **147** джерел з бібліотеки, що, з урахуванням наведених обґрунтувань, відповідає характеру теми і свідчить на користь кваліфікаційної роботи.

09.12.2023 р.

Завідувач кафедри ІПЗ _____ Леонід БЕДРАТЮК
Гарант освітньої програми _____ Оксана ЯШИНА
Керівник кваліфікаційної роботи _____ Галина РАДЕЛЬЧУК