Харківський національний університет імені В.Н. Каразіна
Факультет комп'ютерних наук
Безпека інформаційних систем і технологій

«Допущено до захисту»
Зав.кафедрою БІСТ
Сватовський І.І._____
« » червня 2023р.

Пояснювальна записка
до кваліфікаційної роботи бакалавра
спеціальність: 125 Кібербезпека

на тему: «Розробка протоколів та механізмів для побудови децентралізованого конфіденціального месенджера»

оцінка « » Керівник к.т.н, доцент Нарєжній О.П.
(прізвище та ініціали/підпис)

Голова ЕК Рецензент ст. викл. Осипчук А.В.
(прізвище та ініціали/підпис)

Лемешко О.В._____ Виконавець студентка групи КБ-42

Сира П.А._____
(прізвище та ініціали/підпис)

Харків – 2023

РЕФЕРАТ

Пояснювальна записка містить 66 сторінок, 24 рисунки, 1 таблицю, 1 додаток, 25 джерел.

Метою дипломної роботи є розробка та впровадження застосунку з протоколами та механізмами для побудови децентралізованого конфіденціального месенджера.

Об'єктом дослідження дипломної роботи є протоколи і механізми, які використовуються для забезпечення конфіденційності та децентралізації у месенджерах. Робота також спрямована на вивчення атак, які можуть бути спрямовані на систему, а також на захисні механізми, що допоможуть запобігти шкідливій дії зловмисників та тим самим забезпечити безпеку користувачів.

Предметом розробки є додаток, який надає можливість спілкування через конфіденційний децентралізований месенджер. Це включає в себе розробку архітектури месенджера, розробку протоколів для обміну повідомленнями, аутентифікації користувачів. Також, для забезпечення безпеки використовується функція отримання своєї геолокації за використанням A-GPS. Це включає в себе застосування допоміжного метода, такого як Wi-Fi, у випадку коли за допомогою GPS-приймачів з якихось причин неможливо отримати дані. Це робиться для того, щоб у випадку, якщо процес авторизації пройшов зловмисник, то він не мав доступу до системи, так як його локація не відповідає стандартизованій. Децентралізованість самого месенджеру досягається за рахунок P2P.

Для досягнення поставленої мети та вирішення завдань в дипломній роботі використовуються різні *методи дослідження*. Одним з них є аналіз науково-технічної літератури та вивчення наявних робіт у галузі безпеки та використовувані методи захисту інформації. Цей метод дозволяє отримати необхідну теоретичну базу та ознайомитись зі сучасними розробками та рішеннями.

Результатами проведеної роботи є прототип конфіденційного захищеного месенджера, який використовує розроблені протоколи та механізми. В якості забезпечення захищеності використовується процес авторизації та визначення геолокації. У випадку якщо поточна геопозиція не відповідає “встановленій”, то доступу до листування не надається. Сам застосунок реалізованого на мові програмування Python. Окрім зазначених способів захисту додатково використовуються наступні : обмеження кількості спроб входу, розмежування рівнів доступу, валідація введених даних під час встановлення координат, використання протоколу HTTPS для захищеного з'єднання при передачі даних для визначення кооріднат.

Ключові слова: ДЕЦЕНТРАЛІЗОВАНИЙ МЕСЕНДЖЕР, A-GPS, МЕРЕЖЕВІ ПРОТОКОЛИ, P2P, ОДНОРАНГОВА МЕРЕЖА.

ABSTRACT

The explanatory note contains 66 pages, 24 figures, 1 table, 1 annexe, 25 sources.

The aim of the thesis is the development and implementation an application with protocols and mechanisms for building a decentralized confidential messenger.

The subject matter are the protocols and mechanisms used to ensure confidentiality and decentralization in messengers. The work is also aimed at studying the attacks that can be directed at the system, as well as at the protective mechanisms that will help prevent the malicious action of attackers and thereby ensure the safety of users.

The scope of the study is an application that provides the ability to communicate through a confidential decentralized messenger. This includes the development of the messenger architecture, the development of messaging protocols, and user authentication. Also, to ensure security, the function of obtaining your geolocation using A-GPS is used. This includes using a back-up method, such as Wi-Fi, when GPS receivers cannot receive data for some reason. This is done so that if an attacker passes the authorization process, he does not have access to the system, since his location does not correspond to the standardized one. Decentralization of the messenger itself is achieved due to P2P.

Research methods are used to achieve the set goal and solve the problems in the thesis. One of them is the analysis of scientific and technical literature and the study of existing works in the field of security and the used methods of information protection. This method allows you to get the necessary theoretical basis and get acquainted with modern developments and solutions.

The results of the work is a prototype of a confidential secure messenger that uses the developed protocols and mechanisms. The process of authorization and geolocation is used to ensure security. If the current geolocation does not correspond to the "set", then access to correspondence is not provided. The application itself is implemented in the Python programming language. In addition to the specified methods of protection, the following are additionally used: limiting the number of login attempts,

distinguishing access levels, validating the entered data when setting coordinates, using the HTTPS protocol for a secure connection when transferring data to determine coordinates.

Keywords: DECENTRALIZED MESSENGER, A-GPS, NETWORK PROTOCOLS, P2P, PEER NETWORK.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ ТА СИМВОЛІВ

P2P	-	Peer-to-peer
GPS	-	Global Positioning System
DDoS	-	Denial of Service Attack
RFeye CRFS	-	Назва комерційного продукту, який розробляється компанією CRFS.
TCP	-	Transmission Control Protocol
HTTP	-	Hypertext Transfer Protocol
TLS	-	Transport Layer Security
SSL	-	Secure Sockets Layer
HTTPS	-	Hypertext Transfer Protocol Secure
AES	-	Advanced Encryption Standard
gpsd	-	Global Positioning System Daemon

ВСТУП

У сучасному світі все більше і більше людей прагнуть до приватності і безпеки у своїй комунікації. Традиційні месенджери, які базуються на централізованих серверах, не завжди здатні забезпечити належний рівень конфіденційності, оскільки дані користувачів можуть бути вразливими перед атаками зловмисників або зловживанням з боку провайдерів послуг та в випадку успішної атаки, що спрямована на сервер, ставить під загрозу можливість спілкування поміж користувачами.

У зв'язку з цим, розробка децентралізованого конфіденційного месенджера стає актуальною задачею. Такий месенджер забезпечує збереження особистих даних користувачів і використання безпечних протоколів комунікації, що дозволяє їм спілкуватися без ризику порушення конфіденційності.

Мета дипломної роботи - створення децентралізованого конфіденційного месенджера, зокрема розробка й застосування протоколів та механізмів, що дозволяють забезпечити ці властивості.

Об'єкт дослідження полягає у розгляданні потенційних атаки на систему й які захисні механізми використовуються для запобігання зловмисній дії й забезпечення безпеки користувачів.

Предмет розробки — додаток, який дозволяє спілкуватися через децентралізований та конфіденційний месенджер. Даний застосунок включає в себе впровадження в механізми захисту A-GPS, який використовується під час авторизації користувачів. Сам месенджер є децентралізованим за рахунок використання технології P2P.

1 ТИПОВІ АРХІТЕКТУРИ МЕРЕЖ ТА ПРИНЦИПИ РОБОТИ GPS

Спершу слід почати з того, що у традиційних моделях таких як клієнт-сервер, характерних для багатьох систем, інформація знаходиться на центральних серверах і передається до клієнтських комп'ютерів через мережі. Клієнтські комп'ютери в цьому випадку виступають переважно в якості пристроїв інтерфейсу користувача. Така централізована архітектура є дуже зручною з точки зору управління правами доступу та забезпеченням безпеки в централізованих системах.

Проте, централізована топологія систем може призвести до неефективності через наявність “вузьких” місць та надмірною витратою різноманітних ресурсів. Навіть, у випадку якщо поліпшити продуктивність апаратного забезпечення і знизити вартість, то встановлення та обслуговування централізованих сховищ є доволі дорогими завданнями. Крім того, для того щоб зберегти актуальну інформацію в цих системах, потрібне втручання людини, що в свою чергу значить що в таких системах необхідний постійний нагляд та підтримка з точки зору забезпечення актуальності інформації.

Одним із варіантів, що дозволить здолати зазначені вище обмеження як раз і стає децентралізована модель, в основі якої покладено P2P (Peer-to-peer) [1].

1.1 Архітектура “Клієнт-сервер”

Мережа вигляду “клієнт-сервер” являється такою у випадку, якщо вона складається з однієї системи, що є високопродуктивною — сервера і кількох інших систем, які переважно являються менш продуктивними — клієнти.

В даній системі єдиним постачальником різноманітних послуг та центральним постачальником — є сервер, в той час як клієнти відправляють запити на отримання вмісту, або виконання певних послуг, не передаючи при цьому жодних власних ресурсів (рис. 1.1) [2].

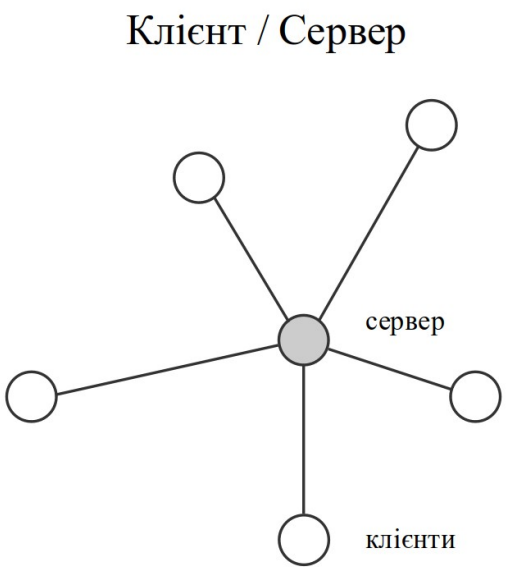

Рисунок 1.1 — Архітектура “Клієнт-сервер”

1.1.1 Переваги “Клієнт-сервер”

В якості позитивних аспектів цієї моделі варто відзначити наступне:

- Управління даними є достатньо простим, оскільки в одному місці зберігаються всі файли, що в свою чергу сприяє швидкому резервному копіюванню та ефективному керуванню помилками.
- Серверне обладнання спеціально розроблене для швидкого оброблення запитів від клієнтів. Всі дані обробляються на сервері, а вже результати передаються клієнту. За рахунок цього зменшується обсяг мережевого трафіку поміж сервером та клієнтською машиною, тим самим пропорційно поліпшується продуктивність мережі.
- “Тонкі” клієнтські архітектури дозволяють в свою чергу достатньо швидко заміняти клієнтів, у випадку такої необхідності, через те що на сервері знаходяться всі програми та дані [2].

1.1.2 Недоліки “Клієнт-сервер”

До основних мінусів відносять :

- Серйозним недоліком даної архітектури є ціна за обслуговування таких систем.
- При виникненні збою на сервері уся система в кращому випадку зазнає величезних затримок, а в гіршому — виходить повністю із строю, що в свою чергу, потенційно, блокує роботу великої кількості користувачів та відповідних додатків та даних [2].

Отже, після розгляду архітектури “клієнт-сервер”, переваг, недоліків, саме час зосередитися на розгляді альтернативного варіанту побудови системи, а саме на P2P.

1.2 P2P

“Peer-to-peer”, або P2P — це мережевий протокол, в основі якого покладено розподілення ресурсів таким чином, що вони являються децентралізованими. В даному контексті месенджерів це означає що спілкування між учасниками або, інакше кажучи вузлів, відбувається без прив’язки до центральних серверів, а напряму поміж учасниками. (рис. 1.2) [1, 2, 3].

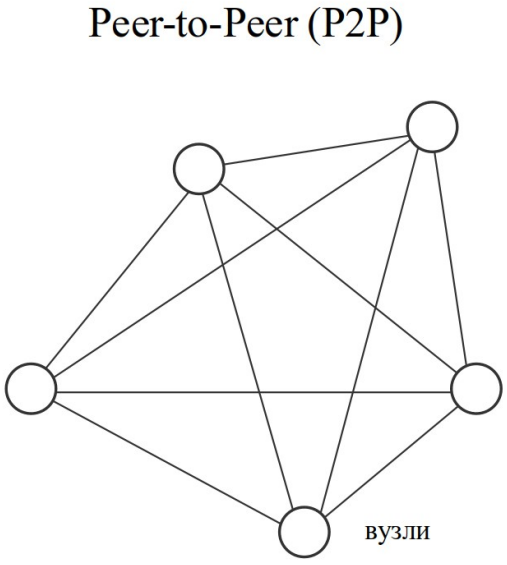

Рисунок 1.2 — Архітектура P2P

Слід зазначити, що абсолютно кожна з технологій має ряд свої переваг разом із недоліками, що і розглянуто нижче в контексті P2P.

1.2.1 Переваги P2P

До переваг даної моделі відносять наступне :

- Через свій принцип роботи у такій моделі відсутня єдина точка відмови, так як навіть у випадку проблем з одним вузлом та його “вибуттям” система все одно продовжує працювати та відбувається взаємодія між іншими вузлами.
- На відміну від клієнт-серверної архітектури даний одноранговий зв’язок передбачає також розподілення як ресурсів, так і навантажень в системі разом з витратами.
- Саме через доцільний розподіл за допомогою того ж однорангового зв’язку та дотримання балансу уникаються “вузькі” місця, такі як перенавантаження трафіку.
- Так як централізований контроль в даній системі відсутній і пристрої взаємодіють між собою напряму, то і присутня краща масштабованості системи [2, 3].

1.2.2 Недоліки P2P

Наряду із перевагами завжди йдуть недоліки. В даному випадку вони наступні :

- Через відкритий характер однорангової мережі може бути високий ризик виникнення вразливостей до атак, шпигунства та випадкових помилок.
- Обмеження в високій пропускній здатності, через те що зв’язок між одноранговими вузлами такого не передбачає
- У порівнянні з централізованими системами, однорангові мережі не мають такої швидкості та ефективності для пошуку інформації, через необхідність взаємодії з різними вузлами, тоді як за використанням центральної бази даних відбувається це робиться вельми краще.

Отже, клієнт-серверна модель передбачає чітке дотримання і витримку ролей в тому сенсі, що декілька учасників взаємодіють з одним сервером, натомість коли в P2P вузли працюють між собою напряму [1, 2, 3].

1.3 Визначення GPS та принцип його роботи

Так як розробляємому додатку, задля забезпечення безпеки використовується механізми GPS (Global Positioning System), то доцільно буде розглянути що це собою уявляє та яким чином працює.

Отже, слід почати з визначення що ж таке GPS. GPS — це система супутникової навігації, що була розроблена Міністерством оборони США, в першу чергу для військових застосувань. Через деякий час даний проект став доступний і для цивільного застосування та використання. Існування почалося лише з 24 активних супутників у 1970-х років [4].

Для кращого розуміння, слід зазначити, що глобальна система позиціонування працює на основі принципу трилатерації. Принцип даного методу заснований на вимірюванні часу, необхідний для передачі сигналу від супутника до приймача, а згодом він використовується для розрахунку відстані між приймачем і супутником. Варто зауважити той факт, що приймач отримує дані мінімум три або більше супутників і використовує їх для обчислення відстані до кожного з них (рис 1.3). Взагал,і чим більше супутників з яких отримуються сигнали, тим точніше можна визначити місцезнаходження приймача.

Це виконується за наступною схемою : декілька супутників насилають радіосигнали, в яких міститься інформація про їх положення та точний час відправлення сигналу. В свою чергу приймач одночасно отримує ці сигнали, які він аналізує та розраховує період, який пройшов від відправлення сигналу супутником до його прийому. Саме знання про час відправлення сигналу та момент прийому дозволяє приймачу обчислити відстань до кожного навігаційного апарату. Так як для точного визначення місцеположення необхідно знати конкретну годину, то отримується сигнал з ще одного супутника. Це робиться для того, щоб відбулася синхронізація внутрішнього годинника з глобальним часом GPS [4, 5, 6].

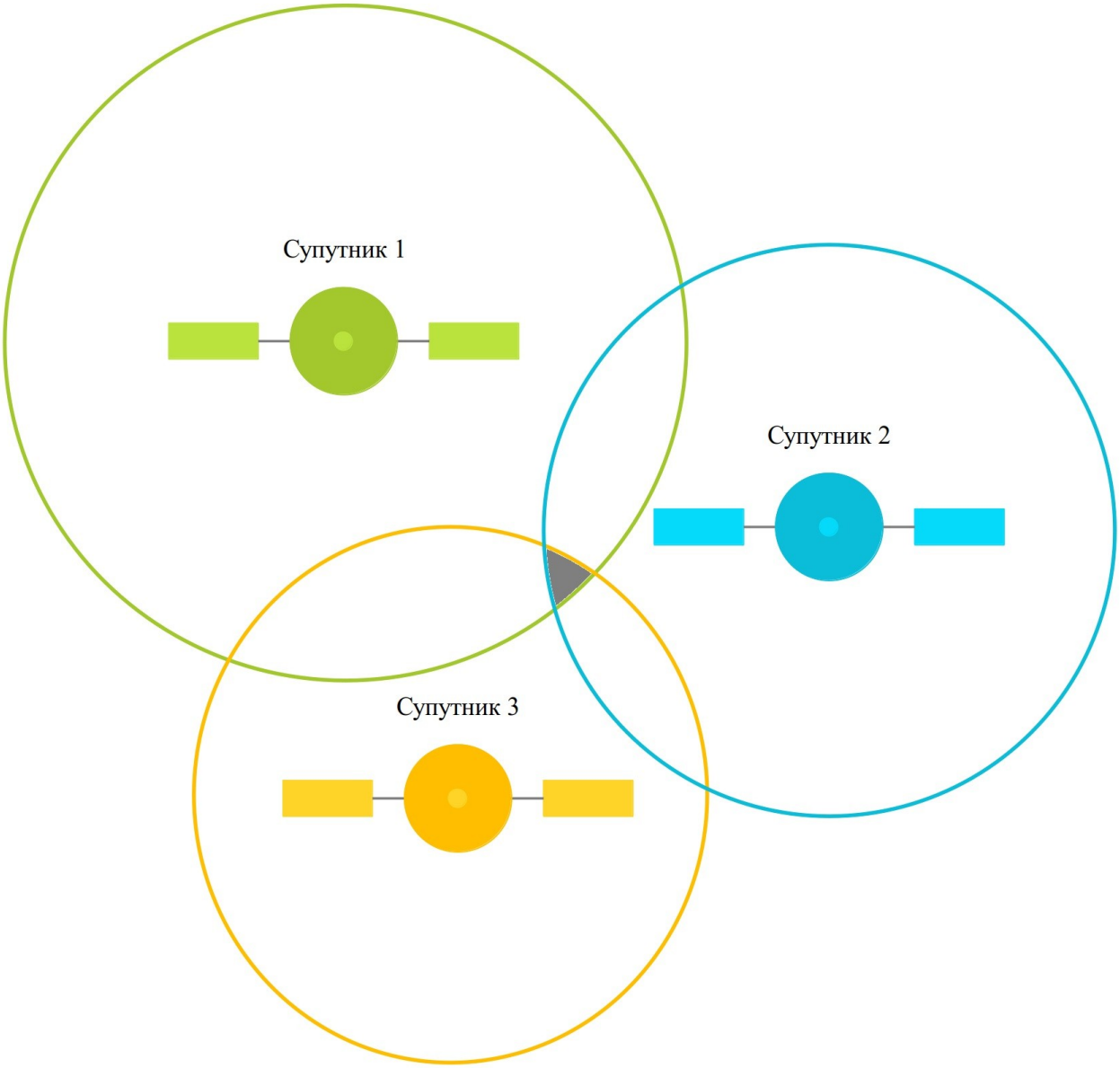

Рисунок 1.3 — Принцип трилатерації

Важливо зауважити, що природні перешкоди у вигляді гір та будівель можуть стати істотною проблемою та заблокувати слабкі сигнали.

2 ОГЛЯД ВИДІВ АТАК, ЇХ РІЗНОВИД І ЯК ЦЬОМУ ПРОТИСТОЯТИ

2.1 Види атак та заходи протистояння

В сучасному цифровому світі зростає загроза кібератак, що можуть нанести шкоду інформаційним системам та мережам. У цьому розділі будуть розглянуто кілька видів атак, які широко використовуються зловмисниками : спуфінг атаки, джамінг атаки та DDoS (Denial of Service Attack). Разом з принципом дії буде також розглянуто способи протидії, з метою уникнення або хоча б мінімізації шкідливих наслідків.

2.1.1 Спуфінг атаки

Спуфінг — це тип мережевих атак, метою яких є обман використовуваних технічних засобів шляхом передачі таких зловмисних сигналів, які хоча і нагадують реальні, але не являються їми.

Якщо розглянути даний вид атак з точки зору GPS-приймачів, то необхідно зазначити, що особливістю підробленого сигналу є те, що він має бути сильніший за потужністю, аніж реальний. Не менш важливою умовою є те, що зловмисник повинен бути в непосередній близькості від користувача та мати інформацію про його точне місцезнаходження. Таким чином, при надсиланні підроблених даних з плином часу вони будуть замінені з фактичних на підроблені [7].

Схематичний принцип даної атаки на прикладі GPS зображено нижче (рис. 2.1).

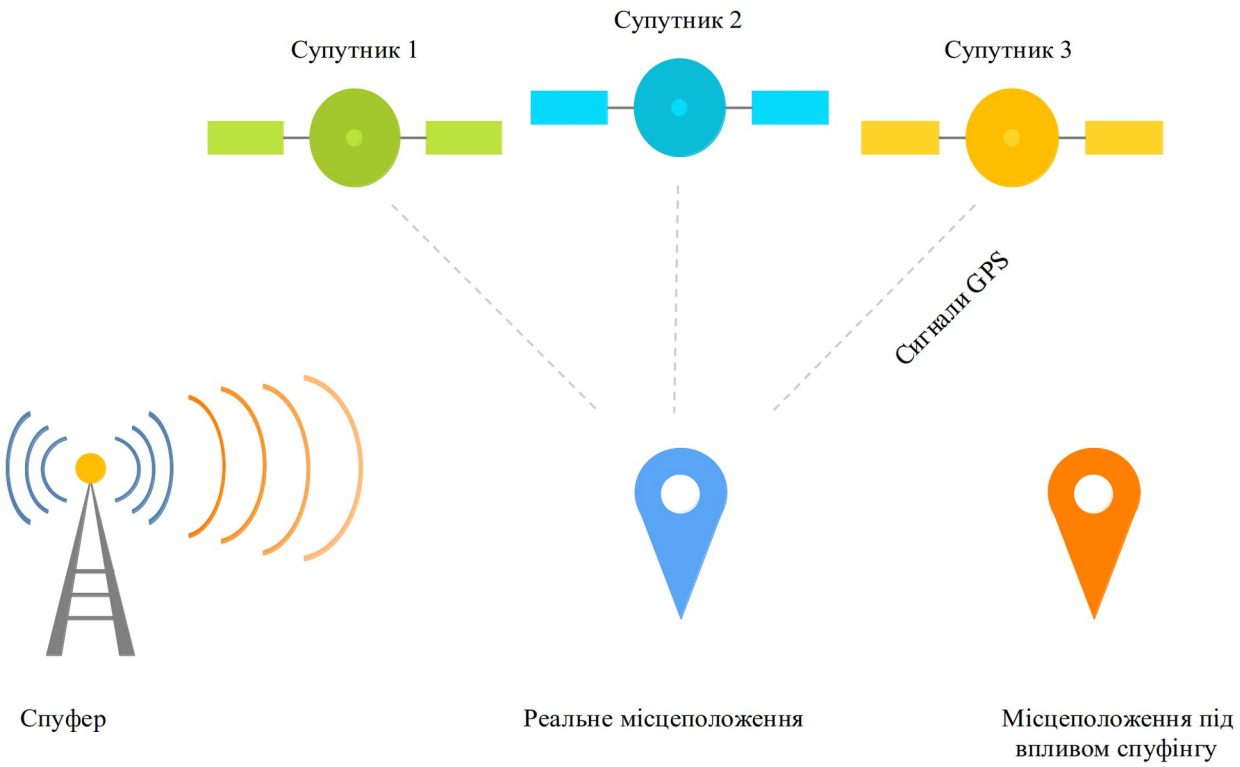

Рисунок 2.1 — Спуфінгова атака

Якщо продовжити розглядання методів боротьби зі спуфінгом на прикладі GPS-приймачів, то в якості запобіжних варіантів — можна виконувати наступні кроки :

- Використання аутентифікації : для забезпечення безпеки можна використовувати аутентифікацію сигналів. Наприклад, це можна зробити за допомогою криптографічних методів, таких як цифровий підпис, що дозволить перевірити легітимність сигналу супутників.
- Виявлення спотворення : сюди можна віднести аналіз параметрів сигналів, або можливість знаходження аномалій за допомогою алгоритмів машинного навчання. Таким чином, при детектуванні GPS приймач може відхилити такі сигнали і використовувати лише достовірні дані.
- Використання порівняння : можливо застосування різноманітних методів порівняння з очікуваними шаблонами, та в випадку виявленні відмінностей відбувається відхилення як спотвореного.

- Використання криптографічних протоколів : до них можна віднести цифровий підпис. В свою чергу це дає можливість перевірки чи був сигнал надісланий від довіреного джерела або ні. При використанні такого метода також збільшується захист від несанкціонованого доступу, що знов таки приводить до зменшення фальсифікованих сигналів.
- Синхронізація з декількома джерелами : якщо використовувати не тільки один сигнал GPS, а й інші різні незалежні джерела, такі як сторонні системи позиціонування або визначення за допомогою Wi-Fi, то ризик спотворення сигналу суттєво зменшується та, в свою чергу, підвищується надійність визначення часу.
- Моніторинг : за умови проведення відстежування та своєчасної перевірки — можна виявити спотворення до того моменту, коли наслідки стануть необоротними. Саме своєчасне прийняття заходів залог успішного запобігання вторгненню й забезпечення захисту систем навігації [8].

2.1.2 Джамінг атаки

Джамінг, або перешкоджання сигналу — це атака, що полягає в блокуванні або зниженні якості сигналу шляхом створення таких радіочастот, які призводять до глушіння передачі сигналів. Різновидами такої атаки можуть бути : збільшення потужності сигналу на одній частоті для зниження якості на інших частотах, або використання спеціальних пристроїв, що створюють сильні електромагнітні поля для перешкоджання сигналу. Важливо зазначити, що такі бар’єри можуть бути не тільки через зловмисну дію, а й ненавмисними.

Отже, боротьба з джамінг атаками може полягати в наступному :

- Виявлення завад : одним із важливих кроків як раз і є виявлення наявності перешкоди в отримуваному сигналі. Для цього можуть використовуватися спеціальні системи моніторингу спектру, які здатні аналізувати частоти та завчасно виявляти незвичайні зміни або інтерференцію, що в свою чергу може свідчити про наявність даного типу атаки. В деяких випадках такі системи можуть автоматично надсилати сигнал, чи повідомлення при

виявленні перепони. В якості прикладу можна навести аналіз частотних спектрів, який надає змогу встановити тривалість перешкод та тип сигналу, що допомагає відрізнити навмисну дію від випадкової.

- Локалізація джерела перешкод : після того, як було виявлено зловмисну дію наступним важливим шагом є визначення її місцеположення. Для цього використовуються мобільні системи пеленгації, які дозволяють визначати напрямок і силу сигналу, а це допомагає локалізувати джерело злочинної дії. Сюди можна віднести застосування мобільних систем пеленгації — це дозволяє визначити, з яких саме напрямків надходять сигнали, що заважають нормальній роботі системи. В результаті це дозволяє прийняти подальші заходи для нейтралізації атаки та за оперативної дії, мінімізувати наслідки шкоди.
- Резервні механізми : для того щоб забезпечити безперервність пристроїв у разі виникнення будь-яких наслідків через атаку — використовуються резервні механізми. Одним з яких являється комунікація через довірені вузли, у випадку якщо основні сервери месенджера стають недоступні.
- Криптографічний захист : важливим аспектом в протистоянні джамінг атакам є використання криптографічних методів захисту. Шифрування та аутентифікація повідомлень дозволяє перевірити їх відповідність і тим самим вчасно запобігти підробленню, або, навіть, заміні отриманих надходжень., що допомогає забезпечити їх цілісність та достовірність.
- Заходи щодо виявлення та реагування : в випадку коли на систему зафіксовано атаку, є дуже важливо мати систему реагування. Це може включати автоматичне сповіщення відповідних служб та своєчасні подальші заходи для усунення заподіяної правопорушником дії. Таким чином, через автоматичне реагування швидкість буде вище, а значить завдана шкода — менше. Також, можуть використовуватися системи моніторингу та аналізу, які дозволяють відстежувати і аналізувати активність перешкод [3, 9].

2.1.3 DDoS атаки

DdoS-атака, або інакше кажучи розподілена атака з обмеженням доступу — це вид кібератаки, в якій зловмисники намагаються перевантажити систему шляхом штучного створення великої кількості запитів і трафіку. Особливістю DDoS-атак є їх розподілений характер. Це означає, що зловмисна дія відбувається з різних джерел, таких як наприклад комп'ютери-зомбі або компрометовані пристрої, що і робить їх виявлення та блокування вельми складним завданням.

К даному типу атак вразливі GPS-приймачі, так як системи на основі P2P, через свою розподілену структуру, не мають такої залежності від одного конкретного сервера[10].

Згідно до джерелу [11] DDoS-атаки можуть бути класифіковані на дві основні категорії : атаки затоплення та атаки на вразливість.

Атаки затоплення — вони споживають ресурси мережі, та водночас переповнюючи її великим обсягом трафіку, що веде до перешкоджання нормальному функціонуванню та обробці легітимних запитів.

Атаки на вразливості — використовують вразливості в протоколах, таких як TCP (Transmission Control Protocol) і HTTP (Hypertext Transfer Protocol), або інших слабких місцях системи, з метою злому та встановленням контролю над нею.

Слід зазначити, що повністю зупинити DdoS-атаки просто неможливо. Однак, їх вплив може бути мінімізовано шляхом використання таких методів, як відмовостійкість або підвищення якості обслуговування. Зазвичай, використовуються ці атаки не “самостійно”, а в поєднанні з IP-спуфінгом. Це робиться задля приховання своєї особи.

Отже, методами та засобами, що запобігають зловмисній дії є :

- Фільтрація : головна ідея полягає в запобіганні атакам і перешкодам для непомітного проникнення до системи зловмисником. Суттєвою частиною цього процесу є як раз використання методів фільтрації на рівні маршрутизаторів. В свою чергу це забезпечує, що лише законний трафік має доступ до системи.

- Ханіпот , або горщик для меду : основна ідея та мета полягає в тому, щоб змусити зловмисника думати що він атакував реальну систему, в той час як ханіпот лише є її імітацією. Інакше кажучи являє собою приманку для хакерів. Слід зазначити, що у випадку шкідливої дії зберігаються записи про зловмисника : тип атаки, яке програмне забезпечення було використано. Тобто, зберігається вся інформація про порушника, що може стати у нагоді при виявленні злочинника та отриманні інформації про нього.
- Безпечна накладка : ідея полягає у побудові накладної мережі поверх IP. Справа в тому, що вона є точкою входу для зовнішньої мережі, щоб встановився зв'язок з захищеною. Передбачається, що ізоляція може бути досягнена у випадку, якщо захищена мережа приховує свої IP-адреси або використовує розподілений брандмауер. Таким чином гарантується, що до такої забезпеченою безпекою системи може потрапити лише довірений трафік від вузлів накладної мережі. Хоча цей механізм забезпечує захист від атак DDoS, застосування можливе лише до приватної структури і не є відповідним для публічного сервера.
- Балансування навантаження : метою цього підходу є розподіл навантаження між різними системами таким чином, щоб жодна з систем не виходила за межі своїх можливостей. В результаті таким чином можна досягнути оптимальну продуктивність. У таких випадках, коли сервер стикається з DDoS атакою, саме балансування й забезпечує стійкість, що робиться шляхом перенаправлення трафіку на інші активні і не піддані атакам сервери. Слід вказати що для того, щоб дійсно забезпечити максимальне балансування, то необхідне виконання такої умови як збільшення пропускної здатності на всіх критичних з'єднаннях. Також потрібно забезпечити достатню кількість реплікованих пристроїв та центрів обробки даних, що робиться для уникнення точки відмови. В свою чергу це також допомагає зменшити площу атаки та достатньо ускладнює вичерпання існуючих ресурсів.

- Проведення профілактичних заходів : під цим мається на увазі регулярне оновлення заходів безпеки, та проведення своєчасного аналізу системи, задля виявлення заражених компонентів [10, 11].

3 ВИЯВЛЕННЯ ВРАЗЛИВОСТЕЙ

Виявлення вразливостей є дуже важливим аспектом забезпечення безпеки будь-якої системи, у тому числі і месенджерів, що з кожним днем стають все більше популярні, а в деяких сферах навіть надзвичайно важливими для стабільної роботи багатьох людей та компаній.

Однак, з пропорційний розвитком технологій в такому ж ключі зростають й атаки на них. Саме недоліки можуть бути використані зловмисниками для того, щоб заподіяти шкоду різноманітним системам, що в свою чергу призводить до досить небезпечних ситуацій.

Виявлення недосконалостей є дуже важливим кроком задля протидії ураженню. Мається на увазі систематичний підхід до ідентифікації та вирішення потенційних проблем безпеки. Даний процес включає в себе аналіз протоколів та алгоритмів, що використовуються в системах, пошук слабких місць у програмному та апаратному забезпеченні, а також перевірку наявності захисту від різних видів атак.

3.1 Загальна процедура визначення вразливостей

Перш за все слід визначити послідовність загальної методології, яку розглянуто в підпунктах нижче [12].

3.1.1 Визначення критичних та найпривабливіших елементів

Задля виконання першого кроку відбувається повний аналіз системи. Ідея полягає в ідентифікуванні ключової мережі, комплексів, які й являються основою працездатності, а також різноманітних механізмів, що в подальшому використовують дані для своєї діяльності.

Для того, щоб визначити привабливість елементів з погляду потенційних зловмисників, необхідно розглянути можливі мотиви та яка потенційна шкода може бути завдана [12].

3.1.2 Проведення оцінки вразливості

Надалі, другий етап передбачає активне сканування абсолютно всієї мережі чи системи за допомогою автоматизованих інструментів, для виявлення проблем безпеки та слабкостей. Такі елементи, які можуть бути віднесені до категорії “критичні” або “привабливі” повинні бути визначені як “цілі”, для яких в першу чергу і потрібно провести подальший аналіз та тестування з реальними сценаріями, з метою знаходження та градації виявлених небезпек.

При оцінці слід також врахувати попередження про вразливості від постачальників програмних засобів, недоліків баз даних та інформація про загрози, яка може бути зазначена в звітах відповідних організацій. Саме ця інформація допомагає зрозуміти потенційні загрози та ризики, що стосуються безпеки месенджера, і таким чином і являються підґрунтям для подальшого розроблення заходів з підвищення охорони.

Якщо, є місце випадку, коли загальна ефективність відповідає встановленим вимогам, то тоді даний етап вважається успішно закінченим. Інакше, слід перейти до наступного пункту [12].

3.1.3 Аналіз вразливості та оцінка ризику

Під час третього етапу розгляду вразливостей, проводиться аналіз джерела та встановлення основної причини виявленої на попередньому етапі — слабкості безпеки. Отже, цей етап надає систематичний підхід до вирішення проблеми. Для кожного виявленого з недоліків встановлюється рейтинг “важкості” або рангу, з урахуванням наступних факторів :

- Які дані перебувають під загрозою.
- Яка система або її частина піддається впливу.
- Наскільки “серйозною” та “інтенсивною” може бути можлива атака.
- Потенційна шкода, яка можлива в результаті ураження [12].

3.1.4 Усунення проблем

Четвертий етап полягає в ефективному вирішенні виявлених незахищеностей з метою покращення та підвищення рівня безпеки. Для кожної з

проблем встановлюється самий оптимальний шлях зниження ризику. Наприклад, заходи, які можуть бути виконані для усунення вразливостей, включають:

- Оновлення всіх налаштувань, або внесення змін у роботу системи.
- Розробка та впровадження елементів, які можуть виправити раніше виявлені недоліки.
- Впровадження нових заходів безпеки, процедур, або використання спеціальних інструментів для підвищення рівня захисту.

Дані заходи спрямовані на запобігання можливим атакам та сприяють зниженню ймовірності виникнення проблем з безпекою у майбутньому. Важливо зазначити, що абсолютно кожен з недоліків потребує саме індивідуального підходу та встановлення відповідних заходів для забезпечення надійності і покращення стану надійності комплексу [12].

3.1.5 Переоцінка системи з урахуванням вдосконалень

Після виконання заходів усунення існуючих проблем та прогалин в безпеці, п’ятим етапом необхідно провести повторний аналіз комплексу, але вже з тими змінами та покращеннями, що були запропоновані. Ця фаза включає в себе визначення нових оцінок таких як : ймовірності нейтралізації, перерви та ефективності системи

Під час повторного розгляду системи необхідно врахувати вплив попередньо внесених змін на загальний рівень захисту та функціональність розглянутої структури. Тобто, досліджуються нові експертизи, які відображають наступне : рівень ймовірності виявлення та усунення можливих небезпек, вірогідність виникнення переривання у роботі системи та взагалі досягнення бажаної ефективності.

Сам процес аналізування повторюється циклічно доти, поки всі виявлені проблеми не будуть повністю подолані, а загальна ефективність не буде підвищена. Цей етап є необхідним для перевірки результативності вжитих заходів та впевненості в перевіряємому механізмі перед подальшим застосуванням та використанням [12].

3.1.6 Звітність про результати

Шостим, та водночас заключним етапом методології експертизи на вразливість безпеки — є підготовка звіту про отримані результати. Головна мета полягає в наданні чіткої та точної інформації щодо ефективності та можливих рекомендацій, стосовно рішень які мають бути прийняті, у випадку якщо поточний рівень захисту виявився недостатнім.

Не менш важливо вказати, що інформація про проведені етапи аналізу вразливостей, виявлені проблеми, які заходи були застосовані для їх усунення та оцінка ефективності вжитих процедур — має бути надана у звіті. Також варто вказати детальну діагностику загроз, виявлених під час усього процесу та їх потенційний вплив на систему.

Звіт повинен містити конкретні рекомендації та заходи для поліпшення захисту системи, у випадку такої необхідності.

Даним кроком не слід нехтувати під час процесу оцінки вразливостей безпеки, оскільки це в результаті дозволяє організації отримати доволі чітке уявлення про поточний стан безпеки й які саме кроки мають бути вжиті для забезпечення надійності та захищеності продукту перед його початковим впровадженням, або подальшою експлуатацією [12].

3.2 Методи виявлення небезпек

Отже, алгоритм подій визначено вище, наступним кроком слід зазначити яким саме чином можна провести виявлення вразливостей. Найвідомішими та найпопулярнішими є нижче наведені варіанти.

3.2.1 Тестування на проникнення

Одним із варіантів тестування на вразливості є тестування на проникнення, в основі якого лежить імітація реальних атак. Це робиться шляхом вторгнення в систему з ціллю виявлення потенційних недоліків та незахищеностей. Мається на увазі що відбувається дійсна атака з використанням тих інструментів та засобів, які зазвичай використовуються зловмисниками. Прикладами може бути багато

чого, починаючи зі злому паролю, чи вбудовуванням шкідливого коду та закінчуючи перехопленням інформації та багато чим іншим.

За допомогою пентестінга може бути визначено наступне :

- Наскільки дійсно буде перенесено використовуваним засобом, в реальній ситуації вже існуючи атаки.
- Який можливий рівень “складності” може бути необхідний зловмиснику для здійснення успішної дії.
- Які заходи зменшать загрозу, та які вразливості мають бути виправлені.
- Чи є належним рівень реагування системи та співробітників в разі виникнення подібної, але вже дійсної ситуації.

Дуже важливим аспектом є той факт, що під час проведення таких заходів тестування безпеки існує великий ризик того, що перевіряємий засіб буде безповоротньо пошкоджено. Таким чином, в обмін на реальні знання та варіанти протидії, ціною може стати неможливість повністю усунути нанесений урон. Саме через це, тестування має проводитися лише після його ретельного розгляду та планування.

Перед атакуванням системи слід провести її перевірку. Дану фазу можна умовно поділити на дві частини.

Перша — полягає у фактичному початку випробування та включає збір і сканування інформації.

Наступною, другою частиною етапу виявлення — є дослідження вразливостей. Під час цього процесу проходить порівняння сканованих хостів, їх служб та програм з базами даних недоліків. Для підвищення швидкості це може бути зроблено автоматизовано за допомогою сканерів, які використовують вбудовані бази даних. Для ручного виявлення прогалин в безпеці використовуються власні знання тестувальників та загальнодоступні бази даних. З іншого боку, аналіз вручну може допомогти виявити, окрім вже існуючих, нові або непередбачувані ризики, які автоматизовані сканери в силу своєї обмеженості можуть пропустити, але це є більш затратно за часом та ресурсами.

І лише тепер, власно, можна перейти до фази атаки, яка і є основною частиною тестування на проникнення. Даний крок включає спроби використання раніше виявлених недоліків. У разі успішного зламу перевіряється недосконаліть та визначаються заходи безпеки для зменшення ризиків.

У контексті месенджера, що забезпечує захищеність користувачів, пентестер може використовувати різні методи і техніки, щоб виявити вразливості в програмному забезпеченні. Наприклад, це може включати в себе перехоплення комунікації, зміна вхідних даних або використання різноманітних інших технік для встановлення наявностей вад і отримання загального уявлення реакції відповідного пристрою на такі вторгнення . [13].

3.2.2 Реверс інжиніринг

Іншим варіантом виявлення недоліків є зворотній інжиніринг. Реверс-інжиніринг або зворотній інжиніринг — це метод дослідження вже існуючої програми з метою розуміння її принципів роботи та внутрішньої структури. Такий підхід передбачає аналіз вихідного коду, або виконуваного файлу програми з метою отримання інформації про його функціональність, алгоритми та дані, що використовуються.

Під час реверс-інжинірингу можуть застосовуватися різні методи, такі як декомпіляція, аналіз динамічної поведінки, розбір виконуваних файлів тощо.

Декомпіляція дозволяє перетворити діючий файл у вихідний код програми, що сприяє полегшенню розуміння її логіки та алгоритмів.

Аналіз динамічної поведінки включає спостереження за взаємодією програми з оточуючим середовищем та введеними даними, що в свою чергу допомагає розібратися у її роботі.

В той же час, розбір виконуваних файлів дає можливість дослідити структуру та компоненти програми, такі як наприклад функції, класи або бібліотеки.

Декомпіляція програмного забезпечення, що застосовується для аналізу месенджера, надає дозвіл для отримання вихідного коду програми з файлу, що є в

експлуатації. Таким чином забезпечується можливість проведення докладної перевірки алгоритмів, обробки даних та механізмів безпеки.

Вивчення протоколів включає в себе аналіз системних даних застосунку. В свою чергу це дозволяє виявити можливі проблеми, такі як наприклад недостатня аутентифікація або потенційно небезпечні команди, що можуть бути використані для потенційних атак.

Отже, використання даного виду інжинірингу дозволяє своєчасно виявити можливі вразливості, помилки або недоліки в їх функціональності, що можуть вплинути на точність та безпеку їх роботи [14].

3.2.3 Атака з бічного каналу

Ще одним методом тестування за допомогою атакування є атака з бічного каналу — метод якої полягає в використанні отримання інформації про пристрій на основі витоку інформації через "бічний канал". Тобто, перевірка на стійкість не спрямована ні на сам алгоритм, ні на програмне забезпечення. Натомість, використовуються незаплановані канали, під якими мається на увазі такі як електромагнітне випромінювання, споживання енергії або часові затримки.

У випадку децентралізованого месенджера, такі атаки можуть бути спрямовані на отримання конфіденційної інформації, включаючи текстові повідомлення, метадані, ключі шифрування та інші. Основні типи атак з бічного каналу включають :

- Електромагнітне випромінювання — використовувані пристрої під час своєї роботи можуть створювати електромагнітні сигнали, які виявляються та перехоплюються зовнішніми спостерігачами. За допомогою такого аналізу можна отримати інформацію про обчислювальні операції, які виконуються, і в подальшому ці відомості можуть бути використані для визначення конфіденційних даних, такі як ключі шифрування або обробка сигналу.
- Аналіз споживання енергії — коли на пристрої виконуються різні операції, то в свою чергу відбувається певні витрати, що в свою чергу потенційно може призвести до втрати інформації. Наприклад, шляхом аналізу енергії

можна допомогти виявити виконання певних дій, або обчислювальних шаблонів, що можуть розкрити конфіденційну інформацію, таку як ключі шифрування або внутрішню структуру алгоритмів.

- Аналіз часових затримок — будь-які пристрої мають певні часові затримки при виконанні операцій. За допомогою моніторингу і спостереженню цих затримок, можна отримати інформацію про внутрішні процеси технічного засобу. Наприклад, невеликі затримки можуть вказувати на виконання певних дій, що відрізняються від звичайних або можуть бути пов'язані з конфіденційною інформацією.
- Аналіз енергетичних витоків — інколи девайси, що є у експлуатації, можуть мати енергетичні витоки, які стають виявними за допомогою відповідного дослідження. Це може бути пов'язано зі змінами електричного струму, або напруги під час виконання процесів. Аналіз може розкрити інформацію про внутрішні процеси пристрою і сприяти отриманню конфіденційних даних [15].

4 АНАЛІЗ МЕТОДІВ УДОСКОНАЛЕННЯ БЕЗПЕКИ WEB-РЕСУРСІВ

Месенджери складають невід'ємну частину нашого життя. Ми використовуємо їх кожного дня для того, щоб поспілкуватися з близькими, вирішити питання по роботі та багато чого іншого. Однак, неможливо не розуміти що з кожним днем пропорційно з тим зростає і кількість атак, тому так важливо активно застосовувати методи покращення безпеки.

4.1 Фільтрація трафіку

Одним із способів яким можна покращити безпеку використовуваних ресурсів та систем є фільтрація трафіку. Воно спрямоване на те, щоб захистити від зловмисників та гарантувати що лише законний потік даних попаде до системи. Сам процес зображено на схемі нижче (рис. 4.1), а в підпунктах далі буде розглянуто різні методи фільтрування [10].

Рисунок 4.1 — Блок-схема процесу фільтрації трафіку

4.1.1 Фільтрація вхідного/вихідного трафіку

Даний метод є дуже поширеним та відомим. Суть даної техніки перешкоджає трафіку з підробленими IP-адресами потраплянню в захищену мережу. В основному, у випадку з вхідним IP-адресом — відсіюється шкідливий трафік, призначений для локальної мережі, а при сортуванні вихідного, відкидається шкідливий трафік, що покидає локальну мережу.

Таким чином фільтрація вхідного трафіку дозволяє пройти в мережу лише тому потоку даних, який відповідає визначеному заздалегідь діапазону доменних префіксів. Відповідно, при використанні підробленої IP-адреси, яка не відповідає нормам, вона буде відкидатися на маршрутизаторах. В результаті забезпечується

запобігання значній кількості DDoS-атак, де використовується підроблена IP-адреса.

Проте даний механізм не є ефективним, коли під час атаки використовуються дійсні IP-адреси ботнетів. Також, наскільки успішними будуть ці відсіювання залежить від знання діапазону очікуваних адрес для порту, що не завжди можливо досягнути для складних топологій, що використовуються в різних мережах. Крім того, якщо зловмисник використовує деякі підроблені мережеві ідентифікатори, які потрапляють у діапазон дійсних адрес, фільтри на маршрутизаторах просто не в змозі виявити зловмисну дію у таких випадках.

Також, слід зазначити, що використання відсіювання вхідного/вихідного трафіку вимагає наявності інструментів тунелювання для користувачів мобільних IP. Це робиться з метою щоб їх трафік був пропущений на маршрутизаторах.

Як приклад реалізації можна навести, що у веб-додатку GPS присутня форма авторизації користувачів, яку можна заповнити для входу в особистий кабінет. Якщо форма авторизації не захищена від атак, куди зловмисники можуть вставляти шкідливий код у поле для введення паролю, то може статися що результатом стане витік даних, або, навіть, повна компрометація веб-додатку [10].

4.1.2 Фільтрація марсіанських (або мартіанських) адрес і перевірка джерела адреси

Фільтрація марсіанських адрес застосовується для того щоб відкинути підроблені IP-адреси, що генеруються з обмеженого набору. Це відсіювання забезпечує, що маршрутизатор не передасть жоден пакет з неправильною початковою або кінцевою IP-адресою. Також гарантується, що будь-який пакет вигляду 255.255.255.255/32 не буде пропущений на маршрутизаторі.

Під час фільтруванні початкова адреса порівнюється з логічним інтерфейсом, на якому він був отриманий. Якщо існує не відповідність тому, через який пакет повинен був бути направлений інтерфейс для досягнення зазначеної початкової адреси, то знов таки відбудеться відкидання маршрутизатором. Саме таким чином і відбувається процес відсіювання з підробленими початковими IP-адресами.

Однак, важливо вказати, що випадки неправильної обробки можуть бути досить великими для маршрутів в асиметричних мережах Інтернету. Мається на увазі що немає ніяких гарантій стовосно збігу інтерфейсів при отриманні чи поверненні пакета з певної початкової адреси. Через це присутній ризик відкидання частини трафіку, що є абсолютно законним [10].

4.1.3 Фільтрація пакетів на основі маршруту

Метод фільтрації пакетів на основі маршруту являється одним із варіантів покращення безпеки децентралізованих месенджерів. Його суть полягає у сортуванні пакетів на основі інформації про шлях, яку отримують основні маршрутизатори. Тобто, в основу закладено наступний принцип, згідно з яким ретранслятори приймають трафік лише з обмеженого набору вихідних IP-адрес. Таким чином, у випадку відмінності адрес від зазначених, відбувається їх відкидання, тому що в такому випадку вони вважаються підробленими. Інакше кажучи, даний метод потребує інформацію про топологію маршрутизації. Підсумовуючи, загальна ефективність залежить від того, чи достатня кількість маршрутизаторів використовує дану техніку сортування.

Слід зробити важливе зауваження, що у випадку, якщо не відбувається підтримка оновленої інформації, то техніка може спричинити відкидання легітимного трафіку через якісь небажані зміни. Крім того, зловмисники також можуть використовувати зламані сеанси протоколу прикордонного шлюзу для обману “кореневої” інформації та “обходу” правил фільтрації. Також присутня вірогідність того, що зловмисником будуть обрані IP-адреси, які не будуть схожі на підроблені та виглядатимуть як легітимні, що зробить цей метод безпеки в такій ситуації неефективним.

Знов таки, повертаючись до прикладу де компанія має власний месенджер, який надає послуги для визначення місцеположення. Для запобігання можливим атакам може використовуватися фільтрація пакетів на основі маршруту. Тобто, кожен пакет, що надходить до додатку, на основі того маршруту, по якому він пройшов перевіряється на відповідність шляху, яким саме й повинен був йти. У випадку якщо IP-адреса вихідного пакета не відповідає очікуваному напрямку, то

в такому випадку він відхиляється системою, і тим самим не пропускається далі до використаному ресурсу. В результаті й уникається проникнення пакетів з підробленими, або недійсними адресами, які можуть бути пов'язані зі зловмисниками, або атаками на ресурс [10].

4.1.4 Протокол перевірки дійсності адреси джерела

Протокол перевірки дійсності адреси джерела є покращеною версією фільтрації на основі маршруту. В даному випадку кожному з маршрутизаторів, що є підключеними до джерела надходить повідомлення, що в собі містять оновлену інформацію про дане джерело. На основі цієї інформації в відповідних таблицях з даними, оновлюється поточна інформація для фільтрування. Цей метод дозволяє подолати проблему асиметрії та динамічності Інтернету, що й виникає у фільтрації на основі маршруту. Однак для того, щоб даний протокол був реалізован, необхідно внести зміни у вже існуючий протокол шляху, що є досить складним процесом. Важливо зазначити, що часткове впровадження протоколу не дає гарантії повного успіху фільтрації зловмисних IP-адрес [10].

4.1.5 Фільтрування кількості переходів

Фільтрування кількості переходів або стрибків — є методом фільтрації пакетів, який використовується, знов таки, для запобігання підробленню IP-адрес і забезпечення безпеки веб-ресурсів.

Суть в тому, що кількість переходів IP-пакетів, під час переміщення від джерела до пункту призначення, неможливо змінити. Тобто, концепція полягає в тому, що використовується підрахунок кількості переходів.

Справа в тому, що дане значення кількості переходів записується в таблиці відображення для кожної адреси джерела. Після того, як пакет надійшов, то вираховується очікуване значення кількості переходів та порівнюється з таблицею відображень, що необхідна для цього конкретного випадку.

Якщо виявляється невідповідність, то визначається факт підробленості і відбувається відкидання. У випадку, коли пакет йде не один, а їх цілий потік, то задля перешкоді атаці всі вони відкидаються.

Слід, зазначити, що правопорушник може підробити дійсну кількість переходів у своїх пакетах, що в свою чергу допоможе пройти фільтр. При потоці таких пакетів система може бути уражена, через не здатність виконання порівнянь та обчислень [10].

4.1.6 Фільтрування на основі історії

Фільтрування на основі історії, це ще один метод фільтрації, де використовується історія маркування пакетів звичайного трафіку для розрізнення зловмисної дії. Під час цього методу адресатом атаки підтримується база даних IP-адрес, яка містить в собі такі, які зазвичай зустрічаються в пунктах призначення. У випадку, якщо система стикається з атакою на пропускну здатність, вона допускає лише ті IP, які знаходяться у їхній базі даних, і таким чином всі інші пакети відкидаються.

Однак, знову, цей метод фільтрації має свої обмеження. Наприклад, якщо зловмисник здатен зробити імітацію свого трафіку атаки як звичайного, то в такому випадку ця методологія відсіювання не здатна успішно впоратись зі шкідливим потоком [10].

4.2 HTTPS

HTTPS (Hypertext Transfer Protocol Secure) є захищеним варіантом протоколу HTTP в поєднанні з TLS (Transport Layer Security), що робиться для безпечної передачі даних між клієнтом та сервером. Стандартний HTTP використовується для передачі даних через інтернет і не забезпечує безпеку на рівні шифрування. Це означає, що дані, які передаються через HTTP, можуть бути перехоплені і прочитані третіми особами. Для кращого розуміння чому так необхідно використання TLS, в таблиці 4.1 нижче наведено основні відмінності та порівняння HTTP з HTTPS з точки зору стандартів та протоколів [16, 17].

Таблиця 4.1 — Порівняння HTTP та HTTPS

Параметри порівняння	HTTP	HTTPS
Стандарти	HTTP/1.1 (RFC 7230)	HTTP/1.1 (RFC 7230) TLS (версія 1.2 або 1.3)
Протоколи	TCP	TCP, SSL(Secure Sockets Layer) або (переважно) TLS
Шифрування	Відсутнє	Присутнє (використовується криптографічні алгоритми)
Аутентифікація	Відсутня	Присутня(використовуються цифрові сертифікати)
Захист від прослуховування	Відсутній	Захист від прослуховування (шляхом шифрування надсилаємих даних)
Захист від модифікації даних	Відсутній	Захист від модифікації (завдяки перевірки цілісності даних)

Отже, так як вже було з’ясовано наскільки важливим є використання протоколу захисту, наступним кроком слід вказати який протокол є найчастіше використовуваним — це TLS 1.3. Даний протокол є останньою версією, яка використовується для забезпечення безпеки HTTPS. Його основні етапи включають наступне :

- Протокол привітання являється початковим етапом взаємодії між клієнтом та сервером, які взаємодіють один з одним з метою встановлення захищеного з’єднання. Даний етап включає в себе наступні кроки : надсилання клієнтом до серверу “привітання” і список підтримуваних

криптографічних алгоритмів, з яких сервер обирає зі списку найбезпечніший і відповідає клієнту. Після цього як користувач разом с сервером обмінюються даними, щоб узгодити параметри шифрування і згенерувати секретні ключі.

- Узгодження параметрів шифрування : на даному етапі клієнт разом з сервером виконують узгодження параметрів шифрування, таких як алгоритми шифрування, хеш-функції та ключі. В свою чергу це гарантує, що обидві сторони розуміють, яким саме чином слід зашифровувати та розшифрувати дані.
- Аутентифікація : це робиться для того, щоб обидві сторони переконалися, що вони є дійсними ідентифікаторами. Для цього використовуються цифрові сертифікати, що підтверджують ідентичність сторін.
- Встановлення захищеного з'єднання : після того як було успішно виконано протокол привітання, відбулося узгодження параметрів шифрування і аутентифікації, TLS 1.3 встановлює захищене з'єднання між клієнтом і сервером. Саме це з'єднання забезпечує шифрування даних, що передаються між ними, та тим самим гарантує їх цілісність.

Тут важливо зазначити, які ж поновлення та особливості є в даному протоколі, яких не було в попередніх версіях :

- Використання сучасних криптографічних алгоритмів : TLS 1.3 використовує сучасні криптографічні алгоритми, такі як наприклад криві Едвардса, для підвищення рівня безпеки.
- Зменшена латентність : під цим мається на увазі використання оптимізованих алгоритмів, що дозволяють зменшити кількість раунд-трипів, необхідних для встановлення з'єднання, і зменшити латентність з'єднання.
- Виключення застарілих алгоритмів : усі застарілі алгоритми, які вважаються небезпечними або неефективними виключаються з застосування [18].

Отже, TLS 1.3 рекомендується використовувати для забезпечення безпеки комунікації та для запобігання можливим атакам на дані.

4.3 Шифрування

Різноманітні методи шифрування застосовуються для захисту конфіденційної інформації, що зберігається на сервері або в базі даних. Так як мова йде про GPS-ресурси, то в такому випадку зашифроване зберігання може стосуватися таких даних, як геолокаційні дані користувачів, маршрути або інша приватна інформація. Використання шифрування забезпечує додатковий рівень безпеки та перешкоджає доступу до цих даних неправомірним особам.

Двома основними видами шифрування являються симетричне та асиметричне, що розглянуто нижче на прикладах AES та RSA відповідно.

4.3.1 AES

AES (Advanced Encryption Standard) — симетричний алгоритм шифрування, який використовується для захищеної передачі та зберігання даних.

Його основні характеристики в себе включають:

- Ключі : використовуються ключі фіксованої довжини, які можуть бути 128, 192 або 256 бітів.
- Блочний шифр : дані розбиваються по блоках, які є фіксованого розміру, що складаються з 128 бітів.
- Перетинання та заміна : в основі AES лежить перетинання та заміна (substitution-permutation) мережа, що забезпечує надійне шифрування даних.
- Криптографічна стійкість : даний алгоритм вважається стійким до різних атак, в тому числі і до криптоаналітичних методів.

До переваг відносять високу швидкість обробки, криптографічну стійкість і широке застосування в різних сферах. Алгоритм також має просту реалізацію та ефективно працює на різних пристроях, що робить його доволі популярним у застосуванні.

До недоліків відносять : обмеження розмірами блоків та ключів, що може інколи робити застосування даного алгоритму досить складним. Деякі атаки, що

використовують фізичні властивості пристрою або відновлення ключа, також можуть мати вплив на стійкість AES [19].

4.3.2 RSA

RSA є асиметричним криптографічним алгоритмом, що використовується для шифрування та цифрового підпису. Основна ідея алгоритму полягає у складності факторизації великих цілих чисел.

Опис алгоритму RSA:

- Генерація ключів:

Обираються два різних простих числа, наприклад, p і q.

Обчислюється їх добуток, який становить модуль для шифрування та розшифрування (формула 4.1).

$$n = p * q \tag{4.1}$$

де p та q - це прості числа.

Обчислюється значення функції Ейлера для n, яке відоме як φ(n) (формула 4.2):

$$\varphi(n) = (p-1) * (q-1) \tag{4.2}$$

де p та q - це прості числа.

Обирається таке ціле число e, яке буде взаємно простим з φ(n), але являється меншим за нього. Це і становить публічний ключ для шифрування.

Надалі знаходиться обернене значення для e модуля φ(n), що позначається як d. Це приватний ключ для розшифрування та підпису.

Складання ключів : пара (e, n) — складає публічний ключ, а (d, n) — приватний ключ.

- Шифрування:

Якщо є повідомлення M, яке потрібно зашифрувати, то за допомогою публічного ключа (e, n) обчислюється зашифроване повідомлення C (за формулою 4.3.) :

$$C = M^e \bmod n \tag{4.3}$$

- Розшифрування:

Отримувач в свою чергу може розшифрувати повідомлення C за допомогою приватного ключа (d, n).

Розшифроване повідомлення M обчислюється наступним чином (за формулою 4.4):

$$M = C^d \bmod n \tag{4.4}$$

Отже, після того як було розглянуто загальний алгоритм, тепер слід розглянути слабкі та сильні сторони. До сильних відносяться :

- Безпека, що забезпечується на складності факторизації великих чисел, що суттєво ускладнює злам алгоритму.
- Асиметрична структура — вона полягає в тому, що використовуються дві пари ключів : публічний та приватний.
- Широке застосування в багатьох системах.

До слабких сторін, інакше кажучи недоліків, можна віднести :

- Обчислювальна складність. Так як в RSA використовується факторизація великих чисел, то процес як шифрування, так і розшифрування, вимагає значних обчислювальних ресурсів, що в свою чергу робить усю роботу відносно повільною.
- Ключова довжина : так як для того, щоб забезпечити належний рівень безпеки використовується достатньо довгі ключі, то чим більше ключ, тим відповідно і більші обчислювальні витрати.

- Приватний ключ має бути належно захищений, так як його втрата ставить хрест на захисті системи [20, 21, 22, 23].

5 РОЗРОБКА ЗАСТОСУНКУ ДЛЯ КОНФІДЕНЦІЙНОГО ДЕЦЕНТРАЛІЗОВАНОГО МЕСЕНДЖЕРУ

У сучасному цифровому світі, де зростає значущість конфіденційності та захисту особистих даних, розробка застосунків для конфіденційного децентралізованого месенджера стає вельми актуальною. Застосунок такого типу, про який йтиме мова, впроваджує низку заходів для забезпечення безпеки, контролю доступу та автентифікації користувачів. Його використання можливе наприклад в межі одного офісу корпорації, де співробітникам необхідно спілкування поміж один одним.

В даному випадку новітність роботи полягає в тому, в даному застосунку поєднано поєднанні механізми авторизації для користувачів з додатковим визначення геолокації, за використанням A-GPS. У випадку виконання успішної авторизації та відповідності, попередньо визначеною організацією, локації розташування отримується доступ до листування у децентралізованому чаті.

Тут слід визначити, що A-GPS використовується тому, що в випадку якщо неможливо визначити поточне місцезнаходження за допомогою GPS, через те що сигнал з якихось причин недоступний або слабкий, то для цього застосовуються додаткові сторонні методи, такі як визначення за допомогою Wi-Fi [24].

5.1 Реалізація застосунку

В результаті реалізації було розроблено додаток для конфіденційного децентралізованого месенджеру який працює наступним чином.

Один із основних аспектів цього застосунку — це перевірка аутентифікації, що передує отриманню доступу до системи. Для того щоб запобігти несанкціонованому доступу до системи використовується авторизація користувачів, в якій обмежено кількість спроб до 5, що в свою чергу дозволяє унеможливити атаку на систему шляхом перебору паролів або імен користувачів.

Після успішної авторизації, за допомогою валідних облікових даних, застосунок проводить визначення геолокації. Це робиться для того, щоб за умови отримання несанкціонованого доступу до системи зловмисником, то потрібно ще і додаткове співпадіння локації користувача з тою, що визначеною організацією.

Сам процес місцезнаходження реалізується шляхом використання технології GPS (Global Positioning System). Проте, у випадку, якщо така можливість, з якихось причин відсутня, то застосунок використовує альтернативний метод — визначення за допомогою Wi-Fi. При цьому, для визначення геопозиції, користувачеві потрібно ввести дві MAC-адреси та сили сигналів з двох пристроїв. Слід зазначити, що застосовується валідація введених даних для визначення локації, та використання HTTPS для захищеного з’єднання під час відправки на сервер інформації для отримання координат.

Для втілення такого ресурсу використовується мова програмування Python та ряд бібліотек і фреймворків, а фрагменти самого коду даної програми наведені у Додатку А. Сама децентралізованість системи основана на використанні P2P.

Отже, при запуску програми першим що бачать користувачі — це вікно з авторизацією, де вони можуть увійти за допомогою свого імені та пароля. Після натискання кнопки "Увійти" виконується метод, який перевіряє введені дані та реагує відповідно. Якщо введені параметри відповідають даним робітника, програма перевіряє, чиїм є даний обліковий запис. Даний процес вікна зі входом показано нижче (рис. 5.1).

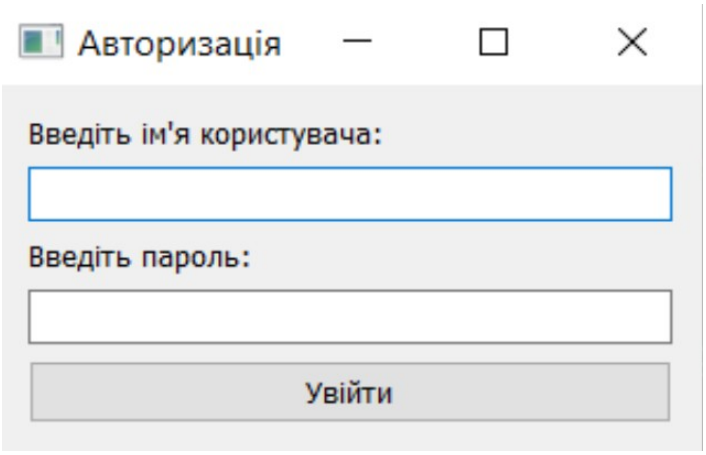

Рисунок 5.1 — Вікно авторизації

Важливо зауважити, що у випадку якщо було введено неправильне ім’я, або пароль, то з’являється спеціальне вікно з повідомленням (рис. 5.2). Для того, щоб виключити випадки проникнення до системи, встановлено обмеження на кількість спроб, яке становить 5 разів, після чого програма завершується (рис. 5.3).

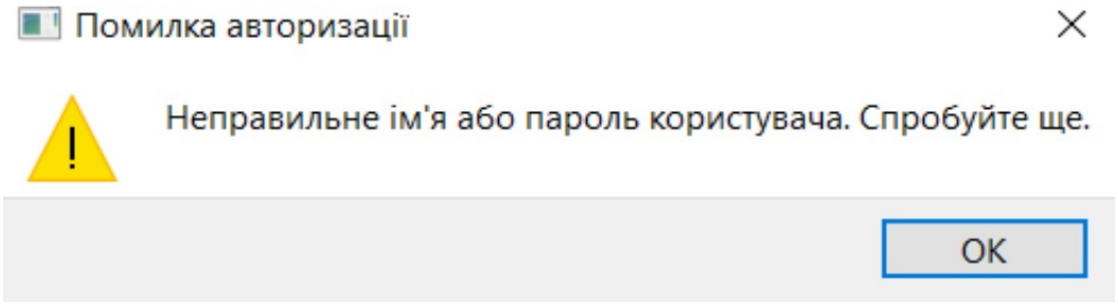

Рисунок 5.2 — Помилка авторизації

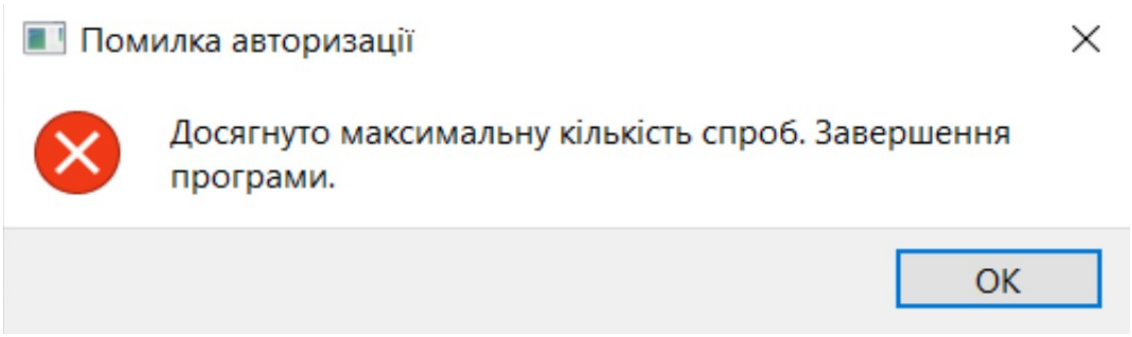

Рисунок 5.3 — Обмеження кількості спроб авторизації

Як було зазначено вище в даному ресурсі здійснюється розмежування доступу. Під цим мається на увазі, що після того, як адміністратор успішного ввійшов до системи (рис. 5.4) він має доступ до усіх даних користувачів, що з’являються у відповідному вікні (рис. 5.5). В той же час після того, як до системи отримуються доступ звичайні клієнти (рис. 5.6), то вони лише переходять до наступного етапу — перевірку своєї локації для подальшого спілкування за допомогою децентралізованого месенджера.

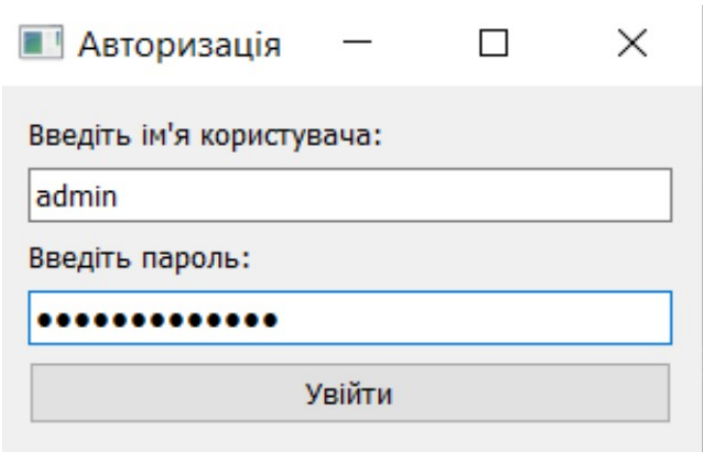

Рисунок 5.4 — Авторизація адміністратора

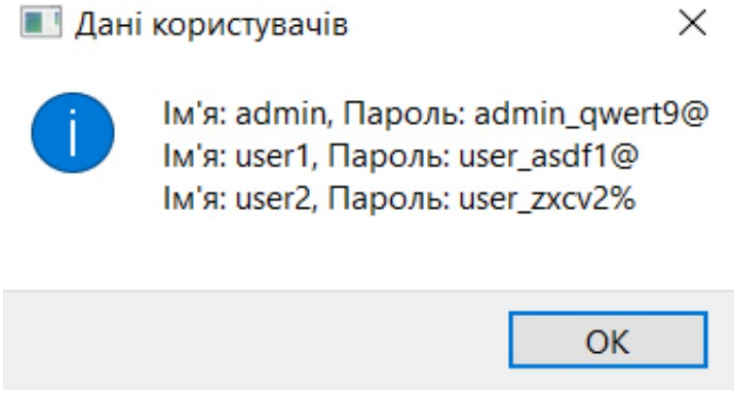

Рисунок 5.5 — Вікно з даними

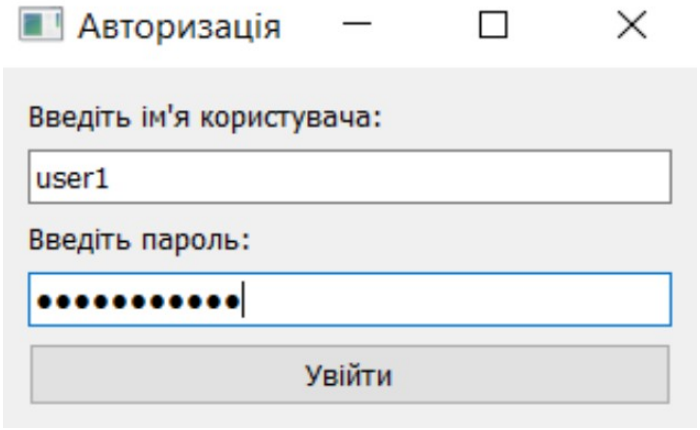

Рисунок 5.6 — Авторизація звичайного клієнта

Далі, все починається зі спроби здобуття даних за допомогою GPS. Першим чином відбувається підключення до локального демона gpsd (Global Positioning

System Daemon). Gpsd — це програма, яка дозволяє отримувати дані з приймачів GPS та надає доступ до цих даних для інших програм.

Тут необхідно вказати, що для коректної роботи необхіден також спеціальний приймач GPS, такий як наприклад, спеціально налаштований Raspberry Pi. В інакшому випадку система не спрацює і поточне місцезнаходження не буде встановлено.

У разі унеможливлення виконання необхідного процесу з якихось причин, викликаються функції наступного етапу отримання координат за допомогою Wi-Fi.

Слід зазначити, для автоматичного визначення за допомогою Wi-Fi своїх поточних координат, потрібні спеціальні дозволи від провайдерів, тому для того щоб зробити цей процес універсальним — було прийнято рішення про використання MAC-адрес та сили сигналів, для здобуття локації. В даному випадку необхідно ввести дані з двох пристроїв, так як тоді можна отримати більше інформації про сигнал Wi-Fi в конкретній області та знизити ризик що зловмисник зможе зробити самостійне підключення до системи одному, без залучення до цього процесу колег.

Слід визначити, що в даному застосунку не розглядається варіант недоброчесного використання робітником, а саме передачу конфіденційної інформації, такої як MAC-адреси та сили сигналу, зловмиснику.

Отже, перевірка за допомогою Wi-Fi відбувається автоматично, якщо програмою не вдалося встановити сигнал GPS та тим самим визначити поточне знаходження. В цьому разі юзер бачить вікно, в якому запропоновано ввести MAC-адреси та сили сигналу Wi-Fi двох пристроїв (рис. 5.7).

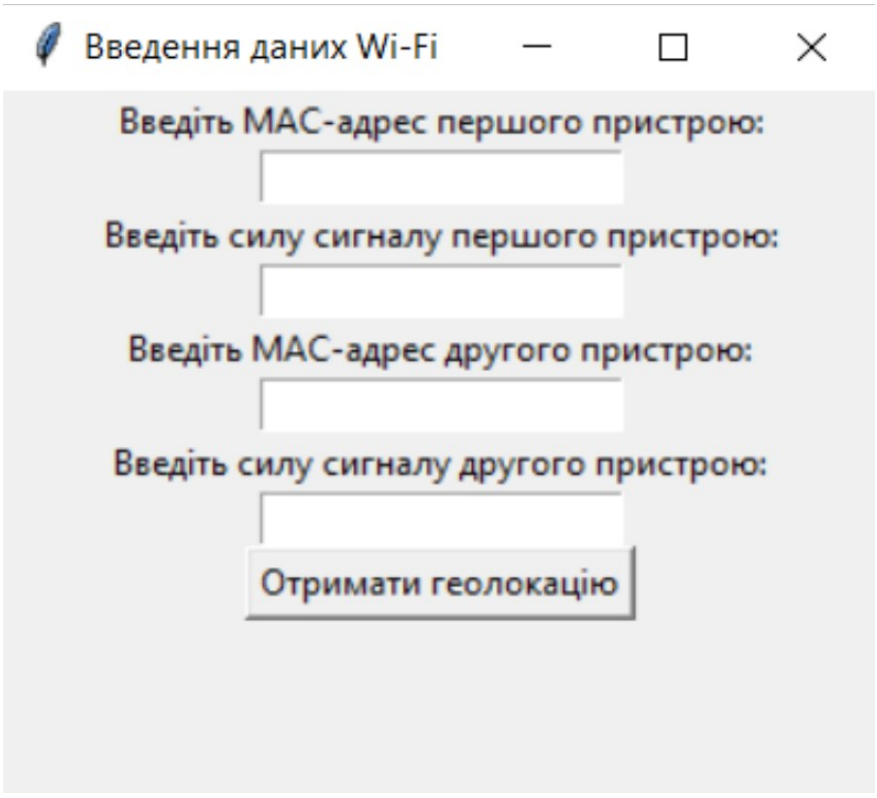

Рисунок 5.7 — Введення даних Wi-Fi

Якщо користувач натискає кнопку до того, як всі поля будуть заповнені, то він отримує відповідне повідомлення про відсутність даних в тому, чи іншому полі (рис. 5.8 та рис. 5.9).

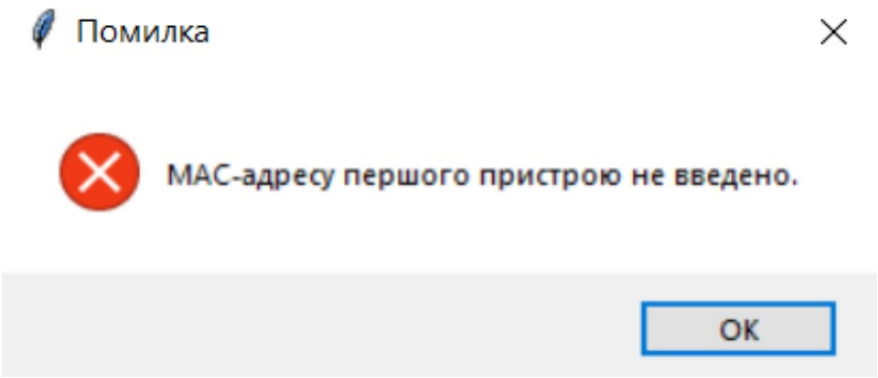

Рисунок 5.8 — Помилка у разі відсутності даних MAC-адреси

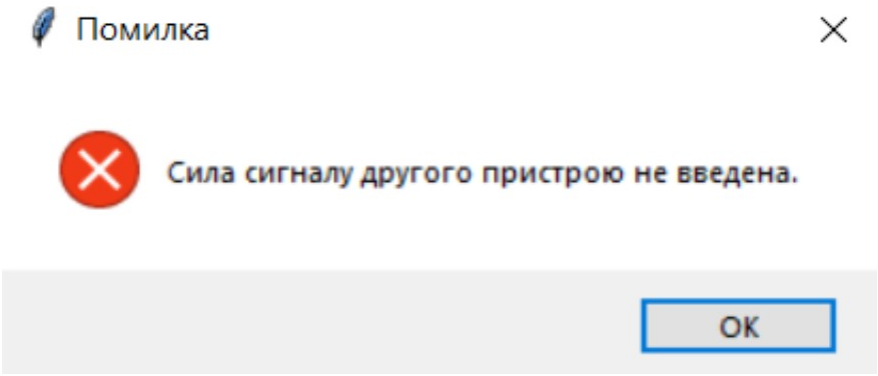

Рисунок 5.9 — Помилка у разі відсутності даних сили сигналу

Після того як всі поля будуть заповнені наступає етап валідації даних. Перевіряється кожна з адрес, шляхом перевірки відповідності введеного значення певному шаблону. Для перевірки чи має введена MAC-адреса правильний формат, використовується регулярний вираз (формула 5.1) [25].

$$r'^([0\text{-}9A\text{-}Fa\text{-}f]\{2\}[:-])\{5\}([0\text{-}9A\text{-}Fa\text{-}f]\{2\})\$' \quad (5.1)$$

де

- ^ — Початок рядка
- [0-9A-Fa-f] — Символи, які допускаються в MAC-адресі.
- 0-9 — Відповідає цифрам від 0 до 9, A-F відповідає літерам від A до F (великими літерами), а a-f відповідає літерам від a до f (малими літерами). Це означає, що допускаються як цифри, так і літери в верхньому, або нижньому регістрі.
- {2} — Два попередні символи повинні бути присутніми. Тобто, [0-9A-Fa-f] {2}, означає два символи, які можуть бути цифрами або літерами
- [:-] — Дозволені символи між парами груп. [:-] - означає, що можливі як двокрапка (:), так і дефіс (-).
- ( ) — Всі символи між такими дужками утворюють одну групу.
- {5} — Значить п'ять повторень попередньої групи. Мається на увазі,що ([0-9A-Fa-f]{2}[:-]){5} означає п'ять груп з двоцифровими символами, розділеними двокрапкою або дефісом

- [0-9A-Fa-f]{2} — Остання двоцифрова група
- $ — Кінець рядка.

Також перевіряється сила сигналу, чи знаходиться він в межах допустимого : від 0 до 100.

У випадку, якщо будь-яке значення не задовільняє вимогам, то будуть з’являтися вікна з відповідними попередженнями (рис. 5.10 та 5.11), до тих пір, поки все не буде виправлено належним чином (рис. 5.12).

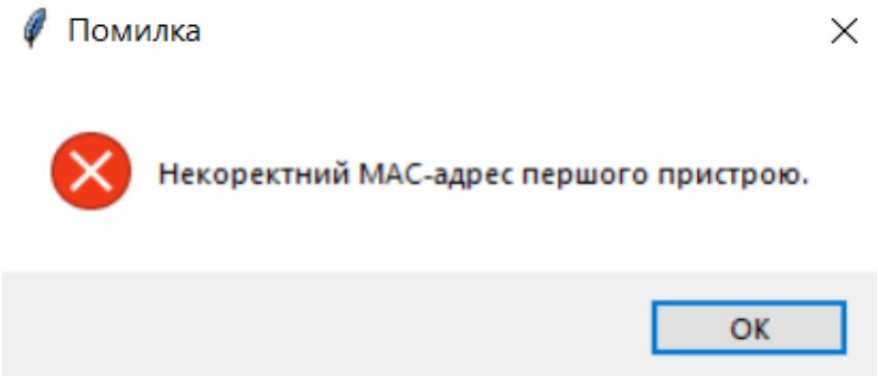

Рисунок 5.10 — Попередження про некоректну MAC-адресу

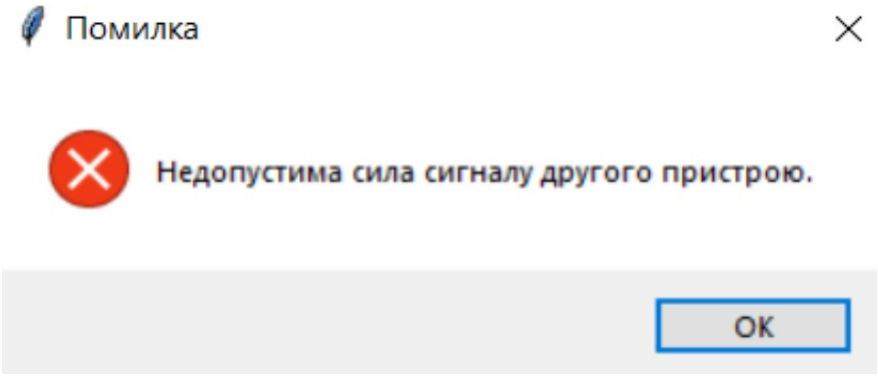

Рисунок 5.11 — Попередження про недопустиму силу сигналу

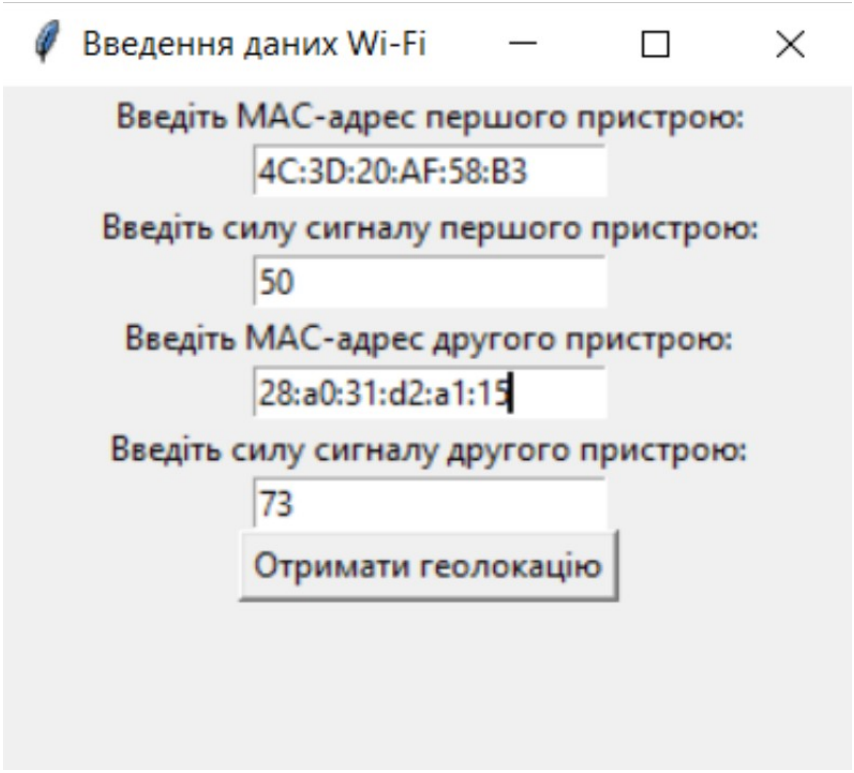

Рисунок 5.12 — Приклад коректного введення даних

Після успішної валідації даних далі формується словник з зібраною інформацією та відправляється POST-запит на сервер Mozilla Location Service, використовуючи при цьому захищений протокол HTTPS, замість HTTP, який не являється безпечним. У відповідь надсилається координати, які містять широту та довготу знайденої локації. Результат отриманих даних з’являється в відповідному вікні (рис. 5.13). У випадку якщо отримано відповідь з локацією, що не відповідає “стандартизованій”, то з’являється відповідне повідомлення та програма автоматично завершується (рис. 5.14).

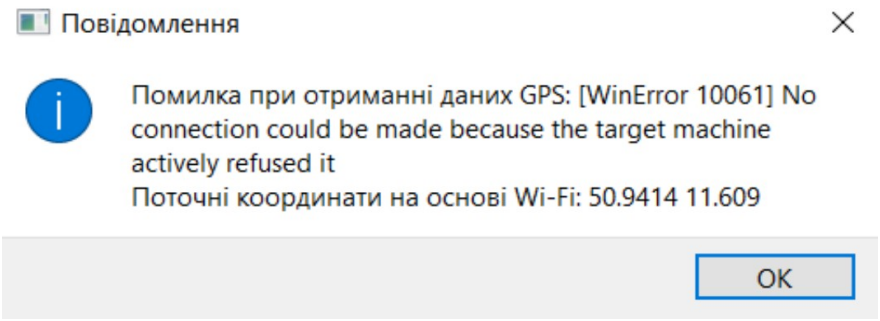

Рисунок 5.13 — Отримання координат за допомогою Wi-Fi

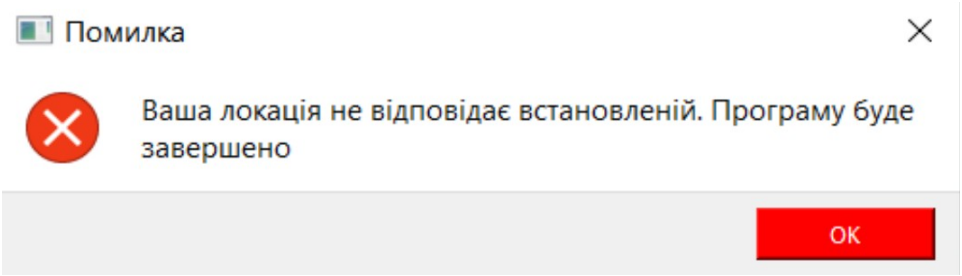

Рисунок 5.14 — Повідомлення про невідповідність місця розташування

Після успішного виконання успішних дій обидва користувачі надалі встановлюють один з одним з'єднання та переписуються поміж собою. Механізм наступний : після запуску програми, клієнт створюється з випадковим номером порту, в межах від 1000 до 5000, і спробує підключитися до сервера, відправивши йому повідомлення "ready", на що чекається відповідь. Після того як підключається другий користувач, то й відбувається етап листування друг з другом у консолі (рис. 5.15 та 5.16).

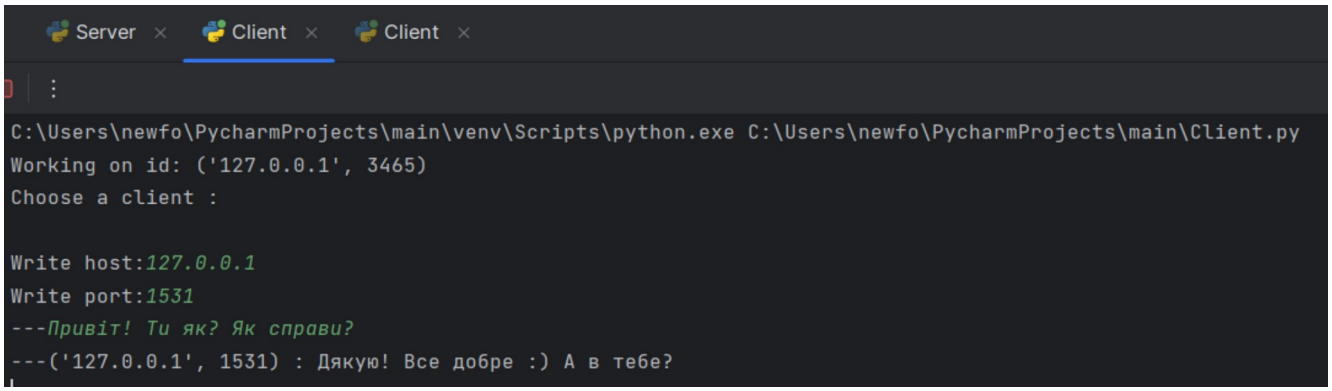

Рисунок 5.15 — Запуск месенджера після успішної перевірки для користувача 1

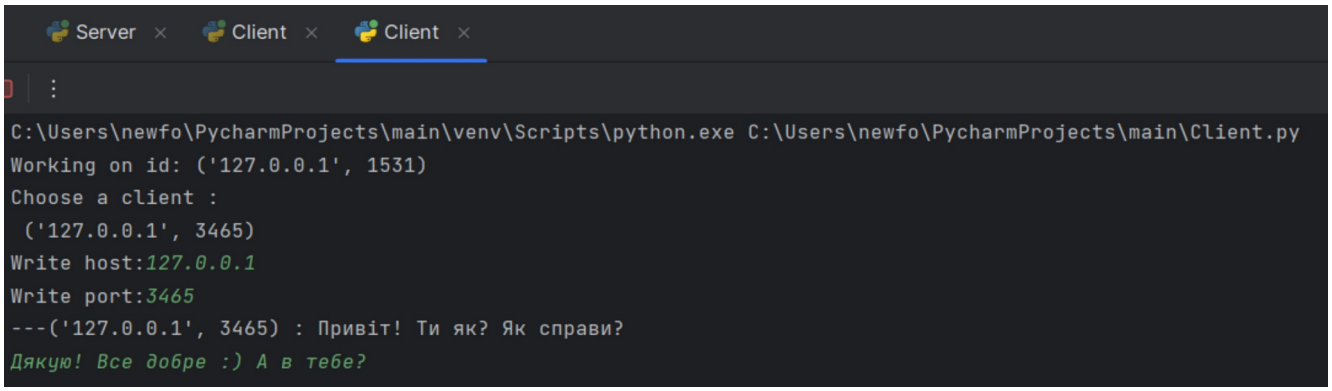

Рисунок 5.16 — Запуск месенджера після успішної перевірки для користувача 2

На двух скріншотах вище можна побачити що також використовується сервер і може здатися що дана програма уявляє собою звичайну архітектуру “клієнт-сервер”. Але в даному випадку сервер був потрібен лише для встановлення з’єднання поміж друг другом та після його закриття (рис. 5.17) програма не завершує свою роботу, а продовжує працювати і клієнти вже листуються напряму (рис. 5.18 та рис. 5.19), що в свою чергу свідчить про реалізацію архітектури P2P.

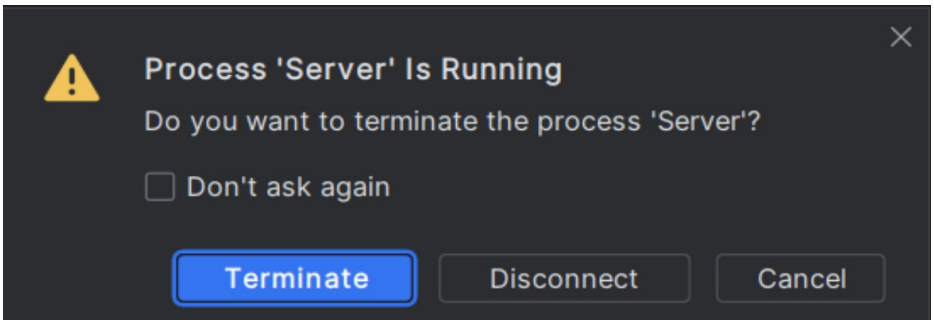

Рисунок 5.17 — Припинення роботи сервера

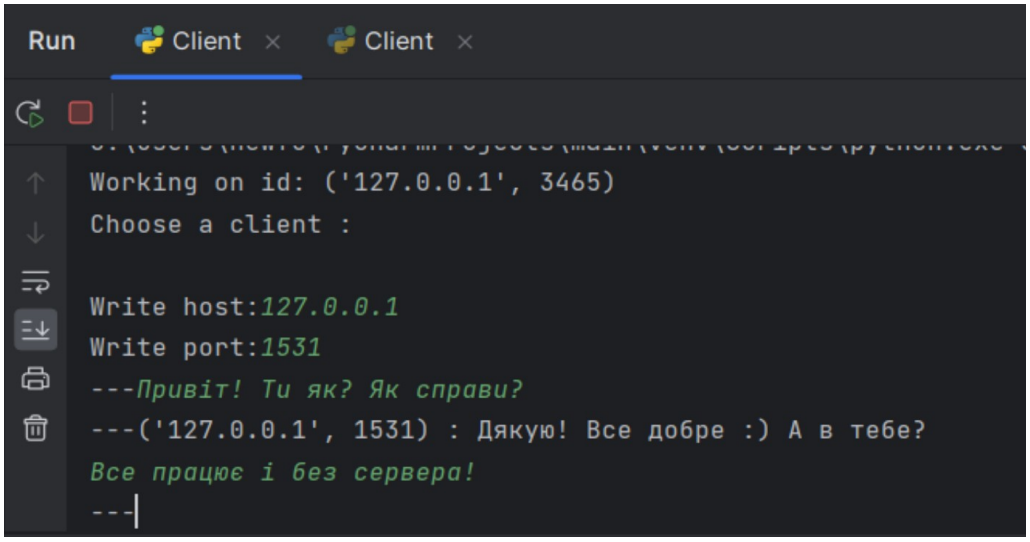

Рисунок 5.18 — Пряме з’єднання між користувачами

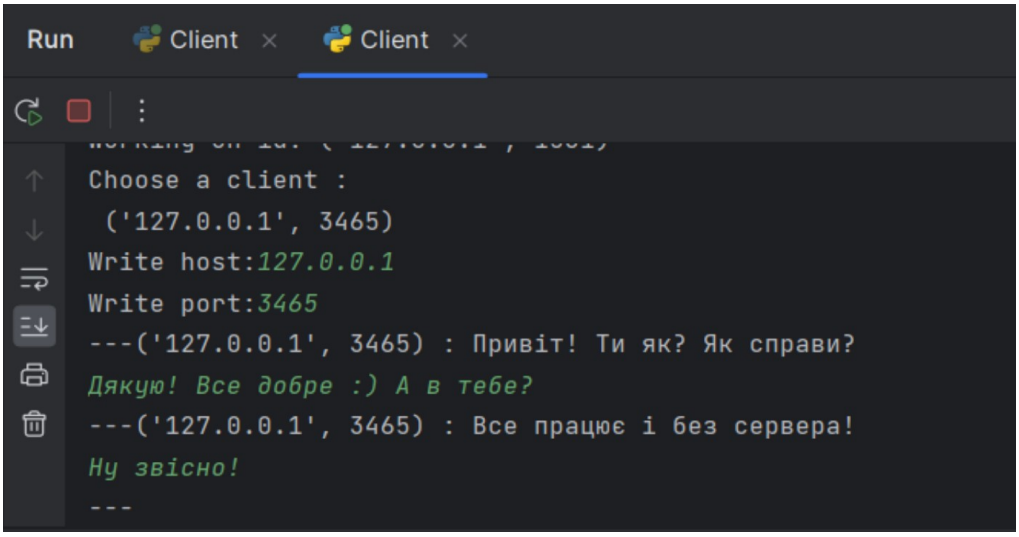

Рисунок 5.19 — Продовження листування між користувачами і без сервера

5.2 Варіанти подальшої розробки та можливі покращення

Варіанти подальшої розробки та можливі покращення цього застосунку є безмежними, пропонуючи широкі можливості для розширення та вдосконалення функціоналу. Додаткові можливості розвитку включають наступні підпункти.

5.2.1 Розширення функцій реєстрації

Сюди можна віднести :

- Додавання нових користувачів, тобто процес реєстрації.
- Збереження введеної інформації про користувача в базі даних з використанням безпеки.

5.2.2 Розширення функціональності месенджера

Може включати наступне :

- Листування у більш звичному інтерфейсі через вікна, замість консолі.
- Можливість надсилати медіафайлів, таких як наприклад фотографій.
- Застосування механізму збереження даних для повідомлень.
- Забезпечення надійного збереження цих даних. Це може бути реалізовано наприклад шляхом резервного копіювання.

5.2.3 Покращення заходів безпеки

Це можна реалізувати шляхом :

- Використання алгоритмів шифрування повідомлень.
- Також можна застосувати різні методи фільтрації, щоб виключити злам системи на етапі введення паролю. Мається на увазі шлях вставки зловмисного коду у поле з паролем.

ВИСНОВКИ

У сучасному світі, де приватність та безпека під час комунікації один з одним, розробка децентралізованого конфіденційного месенджера є дуже актуальною задачею. В даному випадку для того, щоб розробити децентралізований месенджер, окрім стандартної авторизації користувачів також застосовується процес визначення локації, та у випадку відмінності, закриття роботи програми та тим самим недопуск до децентралізованого месенджеру.

У цієй роботі були розглянуті типові архітектури мереж, зокрема "Клієнт-сервер" та P2P, а також принцип роботи GPS та його вразливості до різних атак. Було досліджено основні види атак такі як спуфінг, джамінг та DDoS атаки, а також запропоновані заходи для їх протидії.

Також було розглянуто методи виявлення вразливостей та покращення безпеки додатків, зокрема використання HTTPS для безпечної передачі даних на сервер.

В результаті практичної частини роботи був розроблений застосунок для конфіденційного децентралізованого месенджера. Додаток окрім стандартного процесу авторизації, також використовує GPS, або у випадку якщо не вдається отримати геолокацію за допомогою приймачів Wi-Fi, для точного визначення місцеположення клієнтів, та подальшого допуску, або не допуску до конфіденційного месенджера. Окрім авторизації, були застосовані додаткові заходи безпеки, такі як розмежування доступу, обмеження кількості спроб та використання HTTPS для захищеного з'єднання з сервером під час передачі даних для визначення геолокації.

В результаті після реалізації застосунку визначено наступний потенційний розвиток та вдосконалення включаючи функції реєстрації нових користувачів, розширення функціональності месенджера та покращення заходів безпеки.

Для подальшого розвитку децентралізованого месенджера можна також розглянути додаткові механізми шифрування повідомлень під час їх передачі поміж користувачами даного застосунку.

Усі ці заходи в результаті допоможуть покращити приватність, конфіденційність та безпеку децентралізованого месенджера.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Peer-to-Peer Computing / D. Milojicic та ін. *ResearchGate*. 2002. С. 52. URL: https://www.researchgate.net/publication/2494470_Peer-to-Peer_Computing (дата звернення: 23.03.2023).
2. Maly R. Comparison of Centralized (Client-Server) and Decentralized (Peer-to-Peer) Networking. URL: https://pub.tik.ee.ethz.ch/students/2002-2003-Wi/SA-2003-16.pdf (дата звернення: 23.03.2023).
3. Опорний конспект лекцій з дисципліни : “Технології блокчейн”, ФКН, ХНУ імені В. Н. Каразіна (дата звернення: 24.03.2023).
4. Ershad A. Global Positioning System (GPS): Definition, Principles, Errors, Applications & DGPS. *ResearchGate*. URL: https://www.researchgate.net/publication/340514635_Global_Positioning_System_GPS_Definition_Principles_Errors_Applications_DGPS (дата звернення: 25.03.2023).
5. Global Navigation Satellite Systems: Signal, Theory and Applications. BoD – Books on Demand, 2012. 426 p. URL: https://books.google.de/books?id=RmmQDwAAQBAJ&printsec=frontcover&hl=de (дата звернення: 27.03.2023).
6. Safei S. INDOOR POSITION DETECTION USING WIFI AND TRILATERATION TECHNIQUE. *ResearchGate*. URL: https://www.researchgate.net/publication/230771403_INDOOR_POSITION_DETECTION_USING_WIFI_AND_TRILATERATION_TECHNIQUE (дата звернення: 27.03.2023).
7. Моделі еталонів лінгвістичних змінних для систем виявлення email-спуфінг-атак / А. О. Корченко та ін. *Безпека інформації*. 2018. Т. 24, № 2. URL: https://doi.org/10.18372/2225-5036.24.13062 (дата звернення: 10.04.2023).

8. GPS Vulnerability to Spoofing Threats and a Review of Antispoofing Techniques / A. Jafarnia-Jahromi та ін. *International Journal of Navigation and Observation*. 2012. 127072. https://doi.org/10.1155/2012/127072 (дата звернення: 12.04.2023).
9. Champion M. How to deal with GPS jamming and spoofing. *CRFS*. URL: https://pages.crfs.com/blog/how-to-deal-with-gps-jamming-and-spoofing (дата звернення: 15.04.2023).
10. A survey of distributed denial-of-service attack, prevention, and mitigation techniques / T. Mahjabin та ін. *International Journal of Distributed Sensor Networks*. 2017. Т. 13, № 12. С. 155014771774146. URL: https://doi.org/10.1177/1550147717741463 (дата звернення: 17.04.2023).
11. Mahajan D., Sachdeva M. DDoS Attack Prevention and Mitigation Techniques - A Review. *International Journal of Computer Applications*. 2013. Т. 67, № 19. С. 21–24. URL: https://doi.org/10.5120/11504-7221 (дата звернення: 23.04.2023).
12. Chinnasamy V. Explore Vulnerability Assessment Types and Methodology | Indusface Blog. *Indusface*. URL: https://www.indusface.com/blog/explore-vulnerability-assessment-types-and-methodology/ (дата звернення: 29.04.2023).
13. Recommendations of the National Institute of Standards and Technology / K. Scarfone та ін. National Institute of Standards and Technology, 2008. 80 с. URL: https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-115.pdf (дата звернення: 30.04.2023).
14. “Kaspersky” Encyclopedia. *Kaspersky IT Encyclopedia*. URL: https://encyclopedia.kaspersky.com/ (дата звернення: 30.04.2023).
15. Zhou Y., Feng D. Side-Channel Attacks: Ten Years After Its Publication and the Impacts on Cryptographic Module Security Testing. *Cryptology ePrint Archive*. URL: https://eprint.iacr.org/2005/388.pdf (дата звернення: 01.05.2023).
16. Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing / ред.: R. Fielding, J. Reschke. RFC Editor, 2014. URL: https://doi.org/10.17487/rfc7230 (дата звернення: 03.05.2023).
17. Rescorla E. HTTP Over TLS. RFC Editor, 2000. URL: https://doi.org/10.17487/rfc2818 (дата звернення: 03.05.2023).

18. Rescorla E. The Transport Layer Security (TLS) Protocol Version 1.3. RFC Editor, 2018. URL: https://doi.org/10.17487/rfc8446 (дата звернення: 03.05.2023).
19. Advanced encryption standard (AES). Gaithersburg, MD : National Institute of Standards and Technology, 2001. URL: https://doi.org/10.6028/nist.fips.197 (дата звернення: 05.05.2023).
20. Опорний конспект лекцій з дисципліни : “Прикладна криптологія”, ФКН, ХНУ імені В. Н. Каразіна (дата звернення: 05.05.2023).
21. Jonsson J., Kaliski B. Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1. RFC Editor, 2003. URL: https://doi.org/10.17487/rfc3447 (дата звернення: 05.05.2023).
22. Barker E. B. Recommendation for obtaining assurances for digital signature applications. Gaithersburg, MD : National Institute of Standards and Technology, 2006. URL: https://doi.org/10.6028/nist.sp.800-89 (дата звернення: 08.05.2023).
23. Barker E. Guideline for using cryptographic standards in the federal government:. Gaithersburg, MD : National Institute of Standards and Technology, 2020. URL: https://doi.org/10.6028/nist.sp.800-175br1 (дата звернення: 08.05.2023).
24. Frank Stephen Tromp Van Diggelen. A-GPS: Assisted GPS, GNSS, and SBAS. Boston : Artech House, 2009. 380 с. URL: https://books.google.de/books?id=stTSHdFhrFUC&printsec=frontcover&hl=de (дата звернення: 10.05.2023).
25. Srivastava P. How to validate MAC address using Regular Expression - GeeksforGeeks. *GeeksforGeeks*. URL: https://www.geeksforgeeks.org/how-to-validate-mac-address-using-regular-expression/ (дата звернення: 10.05.2023).

ДОДАТОК А

Фрагменти коду додатку конфіденційного децентралізованого месенджера:

```
# Дані користувачів
class User:
    def __init__(self, username, password):
        self.username = username
        self.password = password

# Листування
def __init__(self, host, port):
    if host == "localhost":
        host = "127.0.0.1"

    self.id = (host, port)
    self.address = None
    self.server = '127.0.0.1', 7777
    print("Working on id:", self.id)

def startProtocol(self):
    self.transport.write("ready".encode("utf-8"), self.server)


def datagramReceived(self, datagram, addr):
    datagram = datagram.decode('utf-8')

    if addr == self.server:
        print("Оберіть клієнта :", datagram)
        self.address = input("Введіть host:"), int(input("Write port:"))
        reactor.callInThread(self.send_message)
    else:
        print(addr, ":", datagram)

# Вікно авторизації
class LoginWindow(QWidget):
    def __init__(self):
```

```
        super().__init__()
        self.setWindowTitle("Авторизація")
        self.resize(300, 150)

        layout = QVBoxLayout()

        self.username_label = QLabel("Введіть ім'я користувача:")
        self.username_input = QLineEdit()
        layout.addWidget(self.username_label)
        layout.addWidget(self.username_input)

        self.password_label = QLabel("Введіть пароль:")
        self.password_input = QLineEdit()
        self.password_input.setEchoMode(QLineEdit.Password)
        layout.addWidget(self.password_label)
        layout.addWidget(self.password_input)

        self.login_button = QPushButton("Увійти")
        layout.addWidget(self.login_button)

        self.login_button.clicked.connect(self.login)

        self.setLayout(layout)

        self.attempts = 0

# Ім"я та пароль
username = self.username_input.text()
password = self.password_input.text()

# Обмеження кількості спроб (5)
self.attempts += 1
if self.attempts >= 5:
  QMessageBox.critical(self, "Помилка авторизації",
                       "Досягнуто максимальну кількість спроб. Завершення програми.")
  self.close()  # Закриття головного вікна програми
else:  # Повідомлення про помилку
  QMessageBox.warning(self, "Помилка авторизації",
                      "Неправильне ім'я або пароль користувача. Спробуйте ще.")
```

```
# Збереження повідомлень
messages = []

# Функція для додавання повідомлення до списку
def log_message(message):
    print(message)
    messages.append(message)

# Валідація MAC-адреса
def validate_mac_address(mac_address):
    pattern = r'^([0-9A-Fa-f]{2}[:-]){5}([0-9A-Fa-f]{2})$'
    return re.match(pattern, mac_address)

# Валідація сили сигналу
def validate_signal_strength(signal_strength):
    return 0 <= signal_strength <= 100

# Отримання координат за допомогою GPS
def get_location_by_gps():
    try:
        # Підключення до локального демона gpsd
        gpsd.connect()

        # Отримання даних про локацію
        packet = gpsd.get_current()

        # Перевірка на отримання даних про геопозицію
        if packet.mode >= 2:
            latitude = packet.position()[0]
            longitude = packet.position()[1]
            log_message("Поточні координати на основі GPS: {} {}".format(latitude,
longitude))

        # Відправлення POST-запиту на сервер з даними Wi-Fi
        response = requests.post(url, json=wifi_data)

        # Обробка відповіді від сервера
        if response.status_code == 200:
```

```
            location = response.json()
            latitude = location['location']['lat']
            longitude = location['location']['lng']
            log_message("Поточні координати на основі Wi-Fi: {} {}".format(latitude,
longitude))
            show_map([latitude, longitude])
        else:
            messagebox.showerror("Помилка", "Не вдалося отримати координати.")


# Створення елементів для вводу даних
label_mac_1 = tk.Label(window, text="Введіть MAC-адрес першого пристрою:")
label_mac_1.pack()
entry_mac_1 = tk.Entry(window)
entry_mac_1.pack()

label_signal_1 = tk.Label(window, text="Введіть силу сигналу першого пристрою:")
label_signal_1.pack()
entry_signal_1 = tk.Entry(window)
entry_signal_1.pack()


# Відображення повідомлень у вспливаючому вікні
if messages:
    log_text = "\n".join(messages)
    app = QApplication([])
    msg_box = QMessageBox()
    msg_box.setIcon(QMessageBox.Information)
    msg_box.setText(log_text)
    msg_box.setWindowTitle("Повідомлення")
    msg_box.exec_()

        else:
            log_message("Дані про позицію за GPS недоступні.")
            return False

    except Exception as e:
        log_message("Помилка при отриманні даних GPS: {}".format(str(e)))
        return False
```

```
# Отримання координат за допомогою Wi-Fi з полв введення
    mac_address_1 = entry_mac_1.get()
    signal_strength_1 = entry_signal_1.get()
    mac_address_2 = entry_mac_2.get()
    signal_strength_2 = entry_signal_2.get()

    # Перевірка на порожність
    if not mac_address_1:
        messagebox.showerror("Помилка", "MAC-адресу першого пристрою не
введено.")
        return
    if not signal_strength_1:
        messagebox.showerror("Помилка", "Сила сигналу першого пристрою не
введена.")
        return
    if not mac_address_2:
        messagebox.showerror("Помилка", "MAC-адресу другого пристрою не
введено.")
        return
    if not signal_strength_2:
        messagebox.showerror("Помилка", "Сила сигналу другого пристрою не
введена.")
        return

    # Конвертація сил сигналу у цілочисельний тип
    try:
        signal_strength_1 = int(signal_strength_1)
        signal_strength_2 = int(signal_strength_2)
    except ValueError:
        messagebox.showerror("Помилка", "Недопустиме значення сили сигналу.")
        return

    # Валідація введених даних
    if not validate_mac_address(mac_address_1):
        messagebox.showerror("Помилка", "Некоректний MAC-адрес першого
пристрою.")
        return
    if not validate_signal_strength(signal_strength_1):
```

```
            messagebox.showerror("Помилка", "Недопустима сила сигналу першого
пристрою.")
            return
        if not validate_mac_address(mac_address_2):
            messagebox.showerror("Помилка", "Некоректний MAC-адрес другого
пристрою.")
            return
        if not validate_signal_strength(signal_strength_2):
            messagebox.showerror("Помилка", "Недопустима сила сигналу другого
пристрою.")
            return

        # Формування даних WiFi
        wifi_data = {
            "wifiAccessPoints": [
                {"macAddress": mac_address_1, "signalStrength": signal_strength_1},
                {"macAddress": mac_address_2, "signalStrength": signal_strength_2}
            ]
        }

        # Відправлення POST-запиту на сервер з даними Wi-Fi
        response = requests.post(url, json=wifi_data)

        # Обробка відповіді від сервера
        if response.status_code == 200:
            location = response.json()
            latitude = location['location']['lat']
            longitude = location['location']['lng']
            log_message("Поточні координати на основі Wi-Fi: {} {}".format(latitude,
longitude))
            show_map([latitude, longitude])
        else:
            messagebox.showerror("Помилка", "Не вдалося отримати координати.")


# Створення елементів для вводу даних
label_mac_1 = tk.Label(window, text="Введіть MAC-адрес першого пристрою:")
label_mac_1.pack()
entry_mac_1 = tk.Entry(window)
```

```
entry_mac_1.pack()

label_signal_1 = tk.Label(window, text="Введіть силу сигналу першого пристрою:")
label_signal_1.pack()
entry_signal_1 = tk.Entry(window)
entry_signal_1.pack()

# Адреса сервера для відправки запитів щодо геолокації
url = 'https://location.services.mozilla.com/v1/geolocate?key=test'
```

2`