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

БАКАЛАВРСЬКА ДИПЛОПНА РОБОТА

на тему:

«Розробка програми для аналізу, виявлення та захисту від атаки типу Man-in-the-Middle (MITM) у бездротових мережах, включаючи підроблені SSL-сертифікати, аномальні ARP-запити-відповіді та зміни в мережевому трафіку»

Виконав: студент 4 курсу, гр. 1КІТС-20Б
спеціальності 125 «Кібербезпека»
ОП «Кібербезпека інформаційних технологій та систем»
Матяш-Рибачук О. С.
Керівник: д.т.н., проф.
Яремчук Ю. Є.
«6» червня 2024 р.
Рецензент: к.т.н., доцент, доцент каф. ОТ
Крупельницький Л. В.
«6» червня 2024 р.

Допущено до захисту
Голова секції УБ кафедри МБІС
д.т.н., проф. Яремчук Ю. Є.
«6» червня 2024 р.

Вінниця ВНТУ – 2024

Вінницький національний технічний університет

Факультет менеджменту та інформаційної безпеки

Кафедра менеджменту та безпеки інформаційних систем

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

ЗАТВЕРДЖУЮ

Голова секції УБ кафедри МБІС

д.т.н., проф. Яремчук Ю. Є.

« 22 » березня 2024 р.

З А В Д А Н Н Я

на бакалаврську дипломну роботу студенту

Матяш-Рибачук Олександрі Сергіївні

1. Тема роботи: «Розробка програми для аналізу, виявлення та захисту від атаки типу Man-in-the-Middle (MITM) у бездротових мережах, включаючи підроблені SSL-сертифікати, аномальні ARP-запити-відповіді та зміни в мережевому трафіку».
Керівник роботи: д.т.н. проф. Яремчук Ю. Є.
затверджені наказом ректора закладу вищої освіти від «11» березня 2024 року № 80.
2. Строк подання студентом роботи 10.06.2024 р.
3. Вихідні дані до роботи: Середовище розробки: Visual Studio Code, Мови розробки – C# та .Net. Операційна система – Windows 11.
4. Зміст розрахунково-пояснювальної записки:
Для досягнення мети було поставлено наступні задачі: дослідити актуальність обраної теми; здійснити аналіз методів захисту інформації; проаналізувати принцип роботи методів захисту; здійснити вдосконалення алгоритму роботи захисту від MITM атаки; реалізувати програмний модуль; здійснити тестування розробленого додатку.
5. Перелік графічного матеріалу: Презентація у форматі PowerPoint з ілюстраціями архітектури розробленої програми, проектування інтерфейсу та результатами тестування

6. Консультанти розділів бакалаврської дипломної роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Розділ I	Яремчук Ю. Є. д.т.н., проф		
Розділ II	Яремчук Ю. Є. д.т.н., проф		
Розділ III	Яремчук Ю. Є. д.т.н., проф		

7. Дата видачі завдання «22» березня 2024 р

КАЛЕНДАРНИЙ ПЛАН

№ етапу	Назва етапів бакалаврської дипломної роботи	Термін виконання		Примітка
		початок	закінчення	
1	Визначення напрямку бакалаврської дипломної роботи, формулювання теми	22.03.2024	25.03.2024	Виконано
2	Аналіз предметної області обраної теми	26.03.2024	29.03.2024	Виконано
3	Постановка задач розробки програмного додатка	30.03.2024	3.04.2024	Виконано
4	Аналіз варіантів роботи програми	5.04.2024	15.04.2024	Виконано
5	Розробка алгоритму роботи	16.04.2024	28.04.2024	Виконано
6	Реалізація програмного модулю	29.04.2024	7.05.2024	Виконано
7	Тустування програмного модулю	8.05.2024	11.05.2024	Виконано
8	Оформлення метеріалів до захисту БДР	12.05.2024	22.05.2024	Виконано
9	Переддипломний захист			
10	Захист бакалаврської дипломної роботи			

Студент (підпис) **Матяш-Рибачук О. С.**

Керівник бакалаврської дипломної роботи (підпис) **Яремчук Ю. Є.**

АНОТАЦІЯ

Бакалаврська дипломна робота складається з 98 сторінок формату А4, на яких є 32 рисунка, 3 таблиці, список використаних джерел містить 30 найменувань.

Бакалаврська робота присвячена розробці програми для аналізу, виявлення та захисту від атаки типу Man-in-the-Middle (MITM) у бездротових мережах, включаючи підроблені SSL-сертифікати, аномальні ARP-запити-відповіді та зміни в мережевому трафіку. Проаналізовано аналоги та зроблено висновок, що дана розробка є актуальною. Програмне забезпечення буде надавати користувачам ефективні інструменти для захисту від даної атаки.

Було розроблено алгоритми та блок-схеми для кожного функціоналу, а також розроблено структуру програми.

Програму було написано на мові програмування C# та використано паттерн MVVM. У якості середовища розробки було обрано Visual Studio Code. Кінцевий програмний продукт відповідає завданням та меті.

Ключові слова: MITM, Man-in-the-Middle, «людина посередині», бездротові мережі, аномальні ARP запити, підроблені SSL-сертифікати, мережевий трафік.

ABSTRACT

The bachelor thesis consists of 98 pages of A4 format, on which there are 32 figures, 3 tables, the list of used sources contains 30 names.

The bachelor's thesis is devoted to the development of an application for the analysis, detection and protection against Man-in-the-Middle (MITM) attacks in wireless networks, including forged SSL certificates, anomalous ARP request-responses and changes in network traffic. Analogues were analyzed and it was concluded that this development is relevant. The software will provide users with effective tools to protect against this attack.

Algorithms and flowcharts were developed for each functionality, as well as the structure of the program was developed.

The program was written in the C# programming language and used the MVVM pattern. Visual Studio Code was chosen as the development environment. The final software product meets the tasks and purpose.

Keywords: MITM, Man-in-the-Middle, wireless networks, anomalous ARP requests, fake SSL certificates, network traffic.

ЗМІСТ

ВСТУП

У сучасному світі інформаційних технологій бездротові мережі набули надзвичайної популярності завдяки своїй зручності та мобільності. Проте, з розвитком технологій, зростає й кількість загроз, пов’язаних з безпекою цих мереж. Однією з найнебезпечніших атак є атака типу Man-in-the-Middle (MITM), яка дозволяє зловмиснику перехоплювати та модифікувати мережевий трафік, що може призвести до викрадення конфіденційної інформації.

Актуальність теми дослідження обумовлена зростанням кількості таких атак та їх постійною еволюцією. Зокрема, використання підроблених SSL-сертифікатів, аномальні ARP-запити-відповіді та зміни в мережевому трафіку ставлять під загрозу безпеку не лише окремих користувачів, а й цілих організацій. Важливим є розробка ефективних методів та інструментів для виявлення та запобігання MITM-атакам, що забезпечить захищеність бездротових мереж.

Головна мета роботи полягає у розробці програми для аналізу, виявлення та захисту від атаки типу MITM у бездротових мережах. Це включає розробку моделей та алгоритмів для виявлення підроблених SSL-сертифікатів, аномальних ARP-запитів-відповідей, моніторинг та аналіз мережевого трафіку для своєчасного виявлення загроз та шифрування/дешифрування файлів.

Для досягнення поставленої мети буде проведено комплексний аналіз існуючих методів захисту, виявлено їх недоліки та розроблено новий підхід, який дозволить більш ефективно захищати бездротові мережі від MITM-атак. У роботі також буде детально описано процес розробки програмного забезпечення, його тестування та впровадження, що дозволить забезпечити його ефективне використання на практиці.

Отже, дана робота є важливим внеском у сферу інформаційної безпеки бездротових мереж, пропонуючи інноваційні рішення для захисту від однієї з найпоширеніших та найнебезпечніших кібератак.

1 ТЕОРЕТИЧНІ ОСНОВИ ЗАХИСТУ БЕЗДРОТОВИХ МЕРЕЖ ВІД MITM-АТАК

У наш час, де бездротові мережі стали невід'ємною частиною нашого життя, питання їхньої безпеки набуває все більшої актуальності. Зростання популярності Wi-Fi, мобільних мереж та інших бездротових технологій відкриває нові шляхи для кіберзловмисників, які прагнуть перехопити та маніпулювати даними, що передаються по цих каналах. Одним із найнебезпечніших типів атак, що ставлять під загрозу конфіденційність та цілісність інформації, є атаки типу "людина посередині" (Man-in-the-Middle).

1.1 Актуальність та значимість дослідження MITM-атак

Саме тому, дослідження MITM-атак є надзвичайно актуальним у нашому світі, де інформаційні технології відіграють ключову роль у повсякденному житті. В умовах стрімкого розвитку цифрових технологій та зростання кількості онлайн-сервісів, таких як онлайн-банкінг, шопінг, соціальні мережі та інші, безпека передачі даних стає критично важливою задачею.

По-перше, зростання кількості та масштабів MITM-атак відзначається постійним вдосконаленням методів та інструментів зловмисників, що робить ці атаки більш складними та небезпечними. Використання доступних онлайн-інструментів та програмного забезпечення робить MITM-атаки більш доступними навіть для непрофесійних зловмисників. Крім того, як вже було сказано - зростання популярності різних онлайн сервісів підвищує вразливість користувачів до MITM-атак.

По-друге, значимість дослідження цих атак зумовлена підвищенням цінності даних, що передаються по бездротових мережах. До таких даних належать особиста інформація, фінансові дані, комерційна таємниця тощо. Крадіжка такої інформації може призвести до серйозних фінансових втрат, шкоди репутації та інших негативних наслідків для користувачів та організацій. Зловмисники можуть

використовувати крадені дані для шантажу, вимагання, фішингу та інших злочинних цілей.

По-третє, бездротові мережі мають певні особливості, які роблять їх більш схильними до MITM-атак. Передача даних по бездротових мережах здійснюється через відкритий ефір, що робить їх доступними для перехоплення зловмисниками. Відсутність фізичного захисту, наприклад, кабелів, також підвищує їх вразливість до несанкціонованого доступу. Використання протоколів автентифікації та шифрування, які не завжди є стійкими до MITM-атак, дозволяє зловмисникам обходити ці протоколи, роблячи бездротові мережі ще більш вразливими.

Нарешті, небезпечні наслідки MITM-атак включають крадіжку особистих даних користувачів, таких як імена, адреси, номери телефонів, номери кредитних карток тощо. Це може призвести до фінансових втрат, оскільки зловмисники можуть використовувати крадені фінансові дані для здійснення шахрайських операцій, таких як зняття коштів з банківських рахунків або здійснення покупок в Інтернеті. Крім того, крадіжка та розголошення конфіденційної інформації може серйозно пошкодити репутацію організацій. Даний вид атаки на критично важливі системи, такі як банківські системи, системи електронного уряду та системи охорони здоров'я, може призвести до серйозних збоїв у роботі та значних фінансових втрат [1].

Таким чином, дослідження MITM-атак є надзвичайно актуальним та значущим завданням, яке має важливе значення для забезпечення кібербезпеки бездротових мереж та захисту даних користувачів та організацій. Використання бездротових мереж стає все більш поширеним, тому розробка ефективних методів протидії MITM-атакам є критично важливою для захисту нашої онлайн-активності.

1.2 Сутність та класифікація MITM-атак

Розглянемо наступний сценарій: дві сторони, Сторона A і Сторона B, домовилися про зустріч для обговорення конфіденційної інформації. Під час зустрічі з'являється третя сторона, Сторона C, яка стверджує, що знайома з

Стороною В і може передати їй повідомлення від Сторони А. Не підозрюючи обману, Сторона А передає Стороні С повідомлення, призначене Стороні В.

Насправді, Сторона С - це зловмисник, який використовує MITM-атаку, щоб перехопити повідомлення. Зловмисник може прочитати, змінити або навіть видати себе за Сторону В, щоб отримати від Сторони А додаткову інформацію [2].

Ця аналогія ілюструє принцип MITM-атаки, де зловмисник таємно втручається в канал зв'язку між двома легітимними сторонами.

Також, такі атаки можна класифікувати за різними критеріями, наприклад, за способом перехоплення зв'язку, за типом маніпулювання даними або за типом цілі.

1) За способом перехоплення зв'язку:

а) атаки на Wi-Fi: зловмисник може використовувати програмне забезпечення для перехоплення Wi-Fi-сигналів, щоб перехопити дані, які передаються між пристроями та точками доступу;

б) атаки на веб-сайти: зловмисник може створити фейковий веб-сайт, який виглядає ідентично легітимному сайту. Коли користувач вводить свої дані на фейковому сайті, зловмисник може їх перехопити;

в) атаки на мережі: зловмисник може використовувати шкідливе програмне забезпечення, щоб заразити комп'ютер або мобільний пристрій і перехопити дані, які передаються через мережу [3].

2) За типом маніпулювання даними:

а) атаки типу "людина посередині з перенаправленням": зловмисник перенаправляє трафік з легітимного сайту на фейковий сайт, де може перехопити дані користувача;

б) атаки типу "людина посередині з модифікацією": зловмисник змінює дані, які передаються між двома сторонами, наприклад, щоб перевести гроші з одного рахунку на інший;

в) атаки типу "людина посередині з підслуховуванням": зловмисник просто підслуховує розмови між двома сторонами .

3) За типом цілі:

а) атаки на особистих користувачів: зловмисник може націлитися на

особистих користувачів, щоб вкрасти їхні особисті дані або фінансову інформацію;

б) атаки на організації: зловмисник може націлитися на організації, щоб вкрасти комерційну таємницю або інші конфіденційні дані;

в) атаки на критично важливі інфраструктури: зловмисник може націлитися на критично важливі інфраструктури, такі як електромережі, транспортні системи та системи охорони здоров'я, щоб завдати шкоди або вивести їх з ладу [4].

Важливо зазначити, що Man-in-the-Middle-атаки постійно розвиваються, тому важливо бути в курсі нових методів та інструментів, які використовують зловмисниками, та вживати відповідні заходи для захисту.

1.3 Специфіка MITM-атак у бездротових мережах

MITM-атаки можуть бути здійснені як у бездротових, так і дротових мережах. Кожен тип мережі має свої особливості, які роблять його більш або менш сприятливим для MITM-атак.

У цьому підрозділі ми порівняємо MITM-атаки в бездротових та дротових мережах, а також зробимо висновки щодо того, який варіант є більш поширеним і чому.

Специфіка MITM-атак у бездротових мережах:

1) Відкритість середовища:

Бездротові мережі по своїй суті є більш відкритими, ніж дротові. Сигнали Wi-Fi можуть перехоплюватися будь-ким, хто знаходиться в радіусі дії точки доступу, навіть якщо він не є авторизованим користувачем.

Це робить бездротові мережі більш сприятливими для MITM-атак, оскільки зловмисникам легше перехопити та маніпулювати трафіком.

2) Відсутність фізичного захисту:

Дротові мережі часто мають фізичний захист, наприклад, кабелі та маршрутизатори, які ускладнюють зловмисникам доступ до мережі.

Бездротові мережі, з іншого боку, не мають такого фізичного захисту, що робить їх більш вразливими для MITM-атак.

3) Використання протоколів:

Бездротові мережі часто використовують протоколи, такі як WEP і WPA, які є більш вразливими до MITM-атак, ніж протоколи, які використовуються в дротових мережах.

Зловмисники можуть використовувати слабкі місця цих протоколів для перехоплення та маніпулювання трафіком.

– Мобільність користувачів:

Користувачі бездротових мереж часто є мобільними, що робить їх більш сприятливими для MITM-атак.

Зловмисники можуть використовувати мобільність користувачів для зміни точок доступу, до яких вони підключаються, що може призвести до перехоплення трафіку.

– Соціальна інженерія:

Зловмисники можуть використовувати методи соціальної інженерії, щоб змусити користувачів бездротових мереж підключатися до фейкових точок доступу, які вони контролюють.

Це може дати зловмисникам доступ до трафіку користувачів і дозволити їм здійснювати MITM-атаки [5].

Специфіка MITM-атак у дротових мережах:

– Фізичний доступ:

Зловмисникам, як правило, потрібен фізичний доступ до дротової мережі, щоб здійснити MITM-атаку. Це може бути зроблено шляхом підключення до мережевого обладнання, такого як маршрутизатори або комутатори, або шляхом прокладання кабелю для перехоплення трафіку.

Це робить MITM-атаки в дротових мережах більш складними, ніж у бездротових, але все ж таки можливими.

– Протоколи та шифрування:

Дротові мережі зазвичай використовують більш стійкі протоколи та

шифрування, ніж бездротові. Це робить їх більш стійкими до MITM-атак, але не повністю небезпечними.

Зловмисники можуть використовувати слабкі місця в протоколах або шифруванні для перехоплення та маніпулювання трафіком.

– Внутрішні загрози:

MITM-атаки в дротових мережах частіше здійснюються зловмисниками, які вже мають доступ до мережі. Це можуть бути співробітники, підрядники або навіть шкідливе програмне забезпечення.

Ці зловмисники можуть використовувати свій доступ до мережі для перехоплення трафіку та здійснення MITM-атак.

– Соціальна інженерія:

Зловмисники можуть використовувати методи соціальної інженерії, щоб змусити користувачів дротових мереж надати їм доступ до мережевого обладнання.

Це може дати зловмисникам можливість перехоплювати та маніпулювати трафіком [6].

Тепер сформуємо таблицю, яка наочно покаже порівняння MITM-атак у бездротових та дротових мережах (табл. 1.1)

Таблиця 1.1 — Порівняння MITM-атак у бездротових та дротових мережах

Характеристика	**Бездротові мережі**	**Дротові мережі**
Відкритість середовища	Відкриті	Закриті
Фізичний захист	Відсутній	Наявний
Протоколи	WEP, WPA, WPA2	TCP/IP, Ethernet
Мобільність користувачів	Висока	Низька

Методи атак	Перехоплення сигналу Wi-Fi, фейкові точки доступу	Фізичний доступ до мережевого обладнання, шкідливе програмне забезпечення
Методи захисту	Шифрування Wi-Fi, VPN, обережність з публічними Wi-Fi	Надійні паролі, оновлення прошивки, сегментація мережі, VPN

Отримуємо висновок, що MITM-атаки в бездротових мережах є більш поширеними, ніж у дротових, через відкритість середовища та відсутність фізичного захисту. Проте, MITM-атаки в дротових мережах можуть бути більш складно виявити та зупинити, оскільки зловмисникам потрібен фізичний доступ до мережі.

1.4 Методи та інструменти реалізації MITM-атак

Зловмисники використовують різні методи та інструменти для здійснення MITM-атак, вибір яких залежить від типу мережі, протоколів, що використовуються, та наявності вразливостей. У цьому розділі ми розглянемо деякі з найпоширеніших методів та інструментів, які використовуються зловмисниками для здійснення MITM-атак.

Знання цих методів та інструментів може допомогти краще захистити себе від MITM-атак.

За методом перехоплення пакетів можна виділити кілька ключових підходів. Аналізуючи випадок атаки MITM, аналітики спочатку намагаються визначити, як шахрай отримав пакети, надіслані на пристрій жертви та з нього.

Отже, шахраї використовують три основні методи:

1) IP Спуфінг

Зловмисник знаходиться безпосередньо між сервером і користувачем і може бачити пакети, надіслані з сайту користувачеві і навпаки. Однак він не має ключа

дешифрування для читання цих пакетів.

Щоб зробити це можливим і запобігти попередженням, шахраї підробляють оригінальний IP-пакет, з якого клієнтський пристрій отримує відкритий ключ.

Потім зловмисник використовує цей ключ, щоб зібрати всі клієнтські запити, прочитати їх і надіслати на сервер.

І жертва, і сервер навряд чи підозрюють про існування третьої особи [7]. На зображенні (рис. 1.1) представлена схема, що ілюструє принцип роботи IP-спуфінгу.

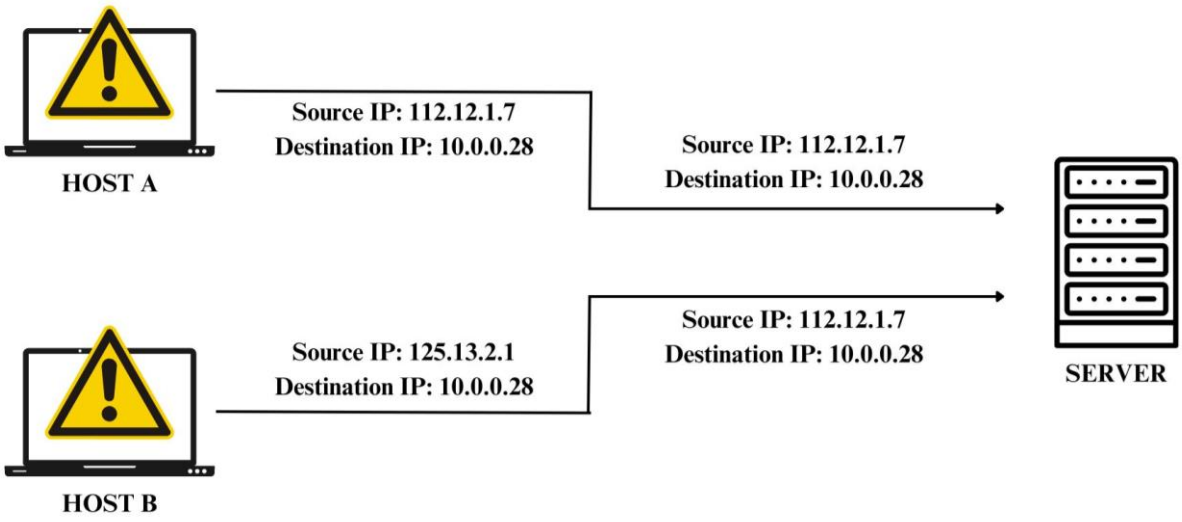

Рисунок 1.1 – Принцип роботи IP-спуфінгу

На схемі зображено двох хостів (Host A та Host B), що намагаються підключитися до сервера.

Host A:

– використовує джерельну IP-адресу 112.12.1.7 і надсилає дані на адресу 10.0.0.28;

– його трафік відображається на сервері з тією ж джерельною IP-адресою (112.12.1.7) і тією ж цільовою IP-адресою (10.0.0.28).

Host B:

– використовує джерельну IP-адресу 125.13.2.1 і надсилає дані на адресу 10.0.0.28;

– його трафік відображається на сервері з джерельною IP-адресою

112.12.1.7 і цільовою IP-адресою 10.0.0.28.

Вона відображає, що обидва хости намагаються підключитися до сервера, але хост B підробляє свою IP-адресу (125.13.2.1) як IP-адресу хоста A (112.12.1.7). Тобто хост B намагається видати себе за хост A, щоб обійти деякі перевірки або системи безпеки на сервері.

Отже, схема чітко ілюструє принцип роботи IP-спуфінгу та показує, як зловмисники можуть використовувати цю техніку для атак на комп'ютери та мережі.

2) Спуфінг DNS

Більш складні трюки, які швидше викликають підозру. Щоб підключитися до веб-сайту, ваш комп’ютер має знайти його в системі доменних імен (DNS).

Через контроль маршрутизатора зловмисники можуть застосувати фальшивий DNS до вибраних сайтів, які ведуть до адрес за вибором хакера.

Коли жертви намагаються підключитися до цього веб-сайту, вони негайно перенаправляються на підроблену сторінку. Саме те, що потрібно шахраю. Ця сторінка часто є фішинговим веб-сайтом, який імітує дизайн сторінок входу з популярних соціальних мереж або веб-сайтів онлайн-банкінгу [8].

На зображенні (рис. 1.2) представлена схема, що ілюструє принцип роботи Спуфінг DNS.

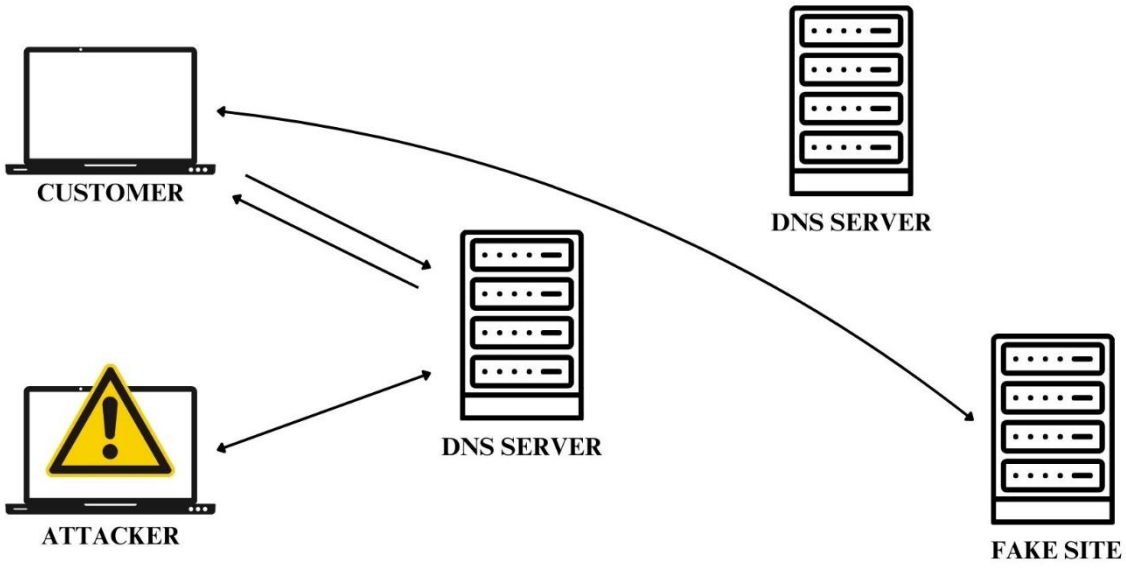

Рисунок 1.2 – Принцип роботи Спуфінг DNS

Ось що зображено на схемі:

– CUSTOMER: це звичайний користувач, який хоче підключитися до якогось вебсайту через DNS-сервер;

– DNS SERVER: це сервер, який обробляє запити на перетворення доменних імен (наприклад, www.example.com) на IP-адреси;

– ATTACKER: це зловмисник, який намагається отруїти кеш DNS-сервера, щоб перенаправити користувача на підроблений сайт;

– FAKE SITE: це підроблений сайт, створений зловмисником, на який він хоче перенаправити користувача.

Опис процесу атаки:

– CUSTOMER відправляє запит до DNS SERVER для отримання IP-адреси сайту;

– ATTACKER також відправляє запит до DNS SERVER, але зловмисник надсилає додаткову підроблену інформацію;

– DNS SERVER, через маніпуляції з боку зловмисника, зберігає підроблену IP-адресу в своєму кеші;

– коли CUSTOMER отримує відповідь від DNS SERVER, він отримує підроблену IP-адресу;

– в результаті CUSTOMER перенаправляється на FAKE SITE замість справжнього вебсайту.

Ця схема демонструє, як зловмисник може використати вразливості в системі DNS для обману користувачів, перенаправляючи їх на фальшиві вебсайти з метою крадіжки даних або інших шкідливих дій.

3) Спуфінг ARP

Цю тактику використовують рідко, оскільки вона працює лише в локальних мережах. Але це все одно може бути ефективним.

Якщо шахраї використовують ваш комп’ютер для роботи у вашій корпоративній мережі, вони можуть заволодіти ним.

Зловмисник постійно надсилає в мережу модифіковані повідомлення ARP. Такі повідомлення використовуються для ідентифікації пристрою іншими

учасниками мережі.

Таким чином усі комп’ютери в мережі змушують вважати, що цей комп’ютер є іншим комп’ютером або сервером. Коли жертви намагаються підключитися до цієї адреси, вони підключаються до комп’ютера під контролем хакера.

Також, подібно до описаного вище методу, він може читати всі мережеві пакети, надіслані жертвою. Щоб підтримувати видимість нормального з’єднання, шахрай надсилає пакети до початкової кінцевої точки та пересилає відповіді жертві [9].

На зображенні (рис. 1.3) представлена діаграма, що ілюструє принцип роботи Спуфінг ARP.

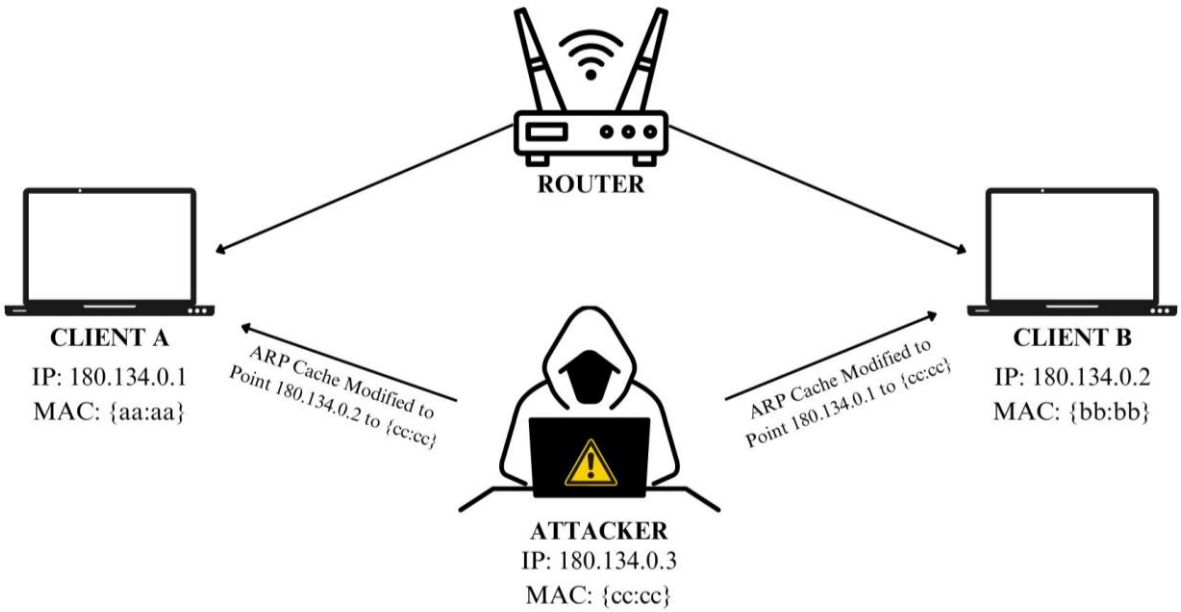

Рисунок 1.3 – Принцип роботи Спуфінг ARP

На ньому зображено чотири пристрої:

– CLIENT 1: цей комп'ютер зображений зліва і підписаний як " CLIENT 1". Він намагається зв'язатися з CLIENT 2, який зображений праворуч;

– ATTACKER: цей комп'ютер зображений посередині і підписаний як "Зловмисник". Він перехоплює трафік між CLIENT 1 та CLIENT 2;

– ROUTER: цей маршрутизатор зображений посередині і підписаний як

"Маршрутизатор". Він перенаправляє трафік між CLIENT 1, зловмисним комп'ютером та CLIENT 2;

– CLIENT 2: цей комп'ютер зображений праворуч і підписаний як " CLIENT 2". Він є кінцевою точкою, з якою намагається зв'язатися CLIENT 1.

Стрілки на схема показують напрямок руху трафіку:

– CLIENT 1 → ROUTER: ця стрілка показує, що CLIENT 1 надсилає запит маршрутизатору;

– ROUTER → ATTACKER: ця стрілка показує, що маршрутизатор перенаправляє запит client 1 зловмисному комп'ютеру;

– ATTACKER → ROUTER: ця стрілка показує, що зловмисний комп'ютер надсилає фальшивий ARP-відповідь маршрутизатору;

– ROUTER → CLIENT 1: ця стрілка показує, що маршрутизатор перенаправляє фальшивий ARP-відповідь CLIENT 1;

– CLIENT 1 → ATTACKER: ця стрілка показує, що CLIENT 1 надсилає дані CLIENT 2 зловмисному комп'ютеру, думаючи, що це CLIENT 2;

– ATTACKER → CLIENT 2: ця стрілка показує, що зловмисний комп'ютер перенаправляє дані CLIENT 2.

У цій ситуації зловмисний комп'ютер може перехоплювати та змінювати дані, що надсилаються між CLIENT 1 та CLIENT 2, наприклад:

- вкрасти конфіденційну інформацію, яка передається між CLIENT 1 та CLIENT 2;
- змінити вміст даних, що передаються між CLIENT 1 та CLIENT 2;
- відправити спам або віруси CLIENT 2.

Отже, ця схема чітко ілюструє принцип роботи MITM-атак з використанням ARP-спуфінгу і показує, як зловмисники можуть використовувати цю техніку для атак на комп'ютери та мережі.

Тепер розглянемо варіанти атак за методами зламу шифрування даних.

Більшість сучасних веб-браузерів і мережевих програм мають функції конфіденційності, які не дозволяють їм читати пакети, навіть якщо вони надсилаються через незахищену мережу. Однак нападники типу «людина

посередині» мають кілька прийомів у своєму рукаві.

1) Викрадення SSL

SSL — це криптографічний протокол, який забезпечує безпечне з’єднання між клієнтами та серверами.

Щоб переконатися, що з’єднання є легітимним, кожна сторона процесу рукостискання TCP отримує відкритий ключ. Шахрай зламується і передає обом сторонам підроблений відкритий ключ. Потім сервер і клієнт обмінюються пакетами, коли встановлюють безпечне з’єднання. Фактичний зв’язок, з іншого боку, контролюється особою посередині [10].

На зображенні (рис. 1.4) представлена схема, що ілюструє принцип роботи Викрадення SSL.

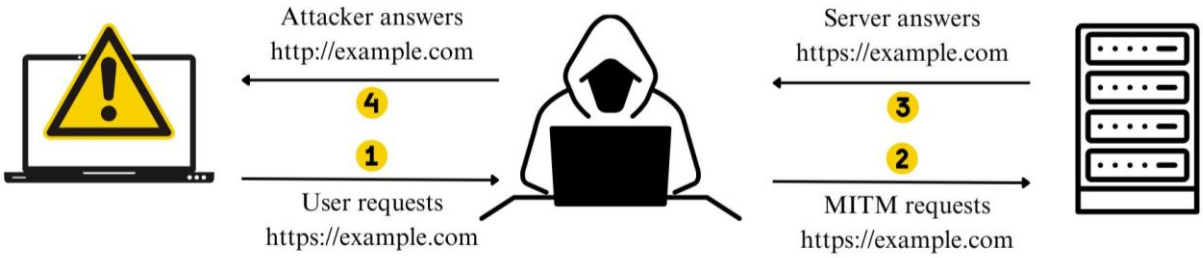

Рисунок 1.4 – Принцип роботи Викрадення SSL

На ньому зображено три елемента:

– комп'ютер користувача: цей комп'ютер зображений зліва і підписаний як "Користувач". Він намагається зв'язатися з веб-сайтом, який зображений праворуч;

– зловмисний комп'ютер: цей комп'ютер зображений посередині і підписаний як "Зловмисник". Він перехоплює трафік між комп'ютером користувача та веб-сайтом;

– веб-сайт: цей веб-сайт зображений праворуч і підписаний як "Веб-сайт". Він є кінцевою точкою, з якою намагається зв'язатися користувач.

Стрілки на діаграмі показують напрямок руху трафіку:

– користувач → зловмисник: ця стрілка показує, що комп'ютер користувача надсилає запит на веб-сайт зловмисному комп'ютеру;

– зловмисник → веб-сайт: ця стрілка показує, що зловмисний комп'ютер перенаправляє запит користувача на веб-сайт;

– веб-сайт → зловмисник: ця стрілка показує, що веб-сайт надсилає відповідь на запит користувача зловмисному комп'ютеру;

– зловмисник → користувач: ця стрілка показує, що зловмисний комп'ютер перенаправляє відповідь веб-сайту користувачеві.

У цій ситуації зловмисний комп'ютер може перехоплювати та змінювати трафік між комп'ютером користувача та веб-сайтом.

Наприклад, зловмисник може:

– вкрасти конфіденційну інформацію, таку як паролі або номери кредитних карток;

– встановити шкідливе програмне забезпечення на комп'ютер користувача;

– перенаправити користувача на фейковий веб-сайт, який виглядає як легітимний.

Отже, дана схема чітко ілюструє принцип роботи MITM-атак і показує, як зловмисники можуть використовувати цю техніку для атак на комп'ютери та мережі.

2) Зачистка SSL

Цей вектор атаки передбачає перехоплення автентифікації TLS на початкових етапах з’єднання. Потім зловмисник знижує рівень з’єднання з HTTPS («S» означає «захищений») до HTTP. Останній передбачає, що дані передаються незашифрованими.

Це дозволяє людині посередині легко читати те, що жертва надсилає на сервер, оскільки немає шифрування. Це важливо, оскільки деякі веб-браузери двічі перевіряють тип з’єднання (HTTPS) і можуть надіслати сповіщення жертві, якщо буде виявлено небезпечний метод [11].

На зображенні (рис. 1.5) представлена схема, що ілюструє принцип роботи

Зачистки SSL.

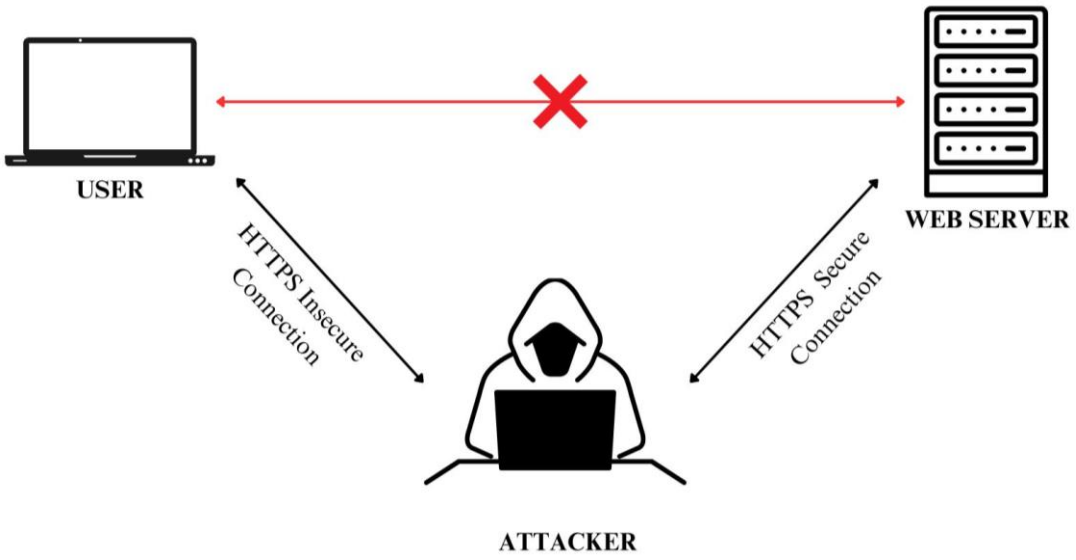

Рисунок 1.5 – Принцип роботи Зачистки SSL

На схемі зображено чотири елемента:

– комп'ютер користувача: цей комп'ютер зображений зліва і підписаний як "Користувач". Він намагається зв'язатися з веб-сайтом, який зображений праворуч;

– зловмисний комп'ютер: цей комп'ютер зображений посередині і підписаний як "Зловмисник". Він перехоплює трафік між комп'ютером користувача та веб-сайтом;

– псевдо-сервер SSL/TLS: цей комп'ютер зображений праворуч від зловмисного комп'ютера і підписаний як "Псевдо-сервер SSL/TLS". Він створений зловмисником для імітації легітимного веб-сайту;

– веб-сайт: цей веб-сайт зображений праворуч і підписаний як "Веб-сайт". Він є кінцевою точкою, з якою намагається зв'язатися користувач.

Стрілки на діаграмі показують напрямок руху трафіку:

– користувач → зловмисний комп'ютер: ця стрілка показує, що комп'ютер користувача надсилає запит на веб-сайт зловмисному комп'ютеру;

– зловмисний комп'ютер → псевдо-сервер SSL/TLS: ця стрілка показує, що зловмисний комп'ютер перенаправляє запит користувача на псевдо-сервер SSL/TLS;

– псевдо-сервер SSL/TLS → зловмисний комп'ютер: ця стрілка показує, що псевдо-сервер SSL/TLS надсилає відповідь на запит користувача зловмисному комп'ютеру;

– зловмисний комп'ютер → користувач: ця стрілка показує, що зловмисний комп'ютер перенаправляє відповідь псевдо-сервера SSL/TLS користувачеві.

У цій ситуації зловмисний комп'ютер може перехоплювати та змінювати трафік між комп'ютером користувача та псевдо-сервером SSL/TLS.

Отже, схема ілюструє принцип роботи атак «Зачистки SSL» і показує, як зловмисники можуть використовувати цю техніку для атак на комп'ютери та мережі.

3) SSL BEAST

Метою таких атак є використання вразливостей у TLS 1.0. Цей підхід вимагає введення шкідливого коду JavaScript у комп’ютер жертви. Це дозволяє зловмиснику перехоплювати пакети. Саме так і можна підробити CBC (Cipher Block Chain).

Таким чином, шахраї можуть розшифрувати файли cookie та маркери автентифікації. Ви можете легко використовувати свій обліковий запис для входу на веб-сайти, відвідувані через заражену мережу [12].

На зображенні (рис. 1.6) представлена схема, що ілюструє принцип роботи SSL BEAST.

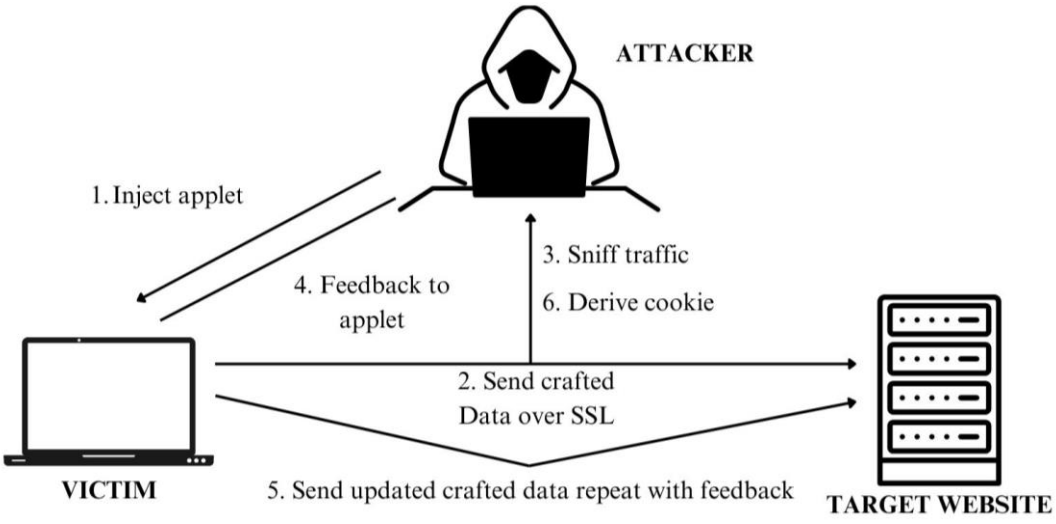

Рисунок 1.6 – Принцип роботи SSL BEAST

На ньому зображено три пристроя:

– комп'ютер користувача: цей комп'ютер зображений зліва і підписаний як "Користувач". Він намагається зв'язатися з веб-сайтом, який зображений праворуч;

– зловмисний комп'ютер: цей комп'ютер зображений посередині і підписаний як "Зловмисний комп'ютер". Він перехоплює трафік між комп'ютером користувача та веб-сайтом;

– веб-сайт: цей веб-сайт зображений праворуч і підписаний як "Веб-сайт". Він є кінцевою точкою, з якою намагається зв'язатися користувач.

Стрілки на діаграмі показують напрямок руху трафіку:

– користувач → зловмисний комп'ютер: ця стрілка показує, що комп'ютер користувача надсилає запит на веб-сайт зловмисному комп'ютеру;

– зловмисний комп'ютер → веб-сайт: ця стрілка показує, що зловмисний комп'ютер перенаправляє запит користувача на веб-сайт;

– веб-сайт → зловмисний комп'ютер: ця стрілка показує, що веб-сайт надсилає відповідь на запит користувача зловмисному комп'ютеру;

– зловмисний комп'ютер → користувач: ця стрілка показує, що зловмисний комп'ютер перенаправляє відповідь веб-сайту користувачеві.

Отже, схема ілюструє принцип роботи атак «SSL BEAST» і показує, як зловмисники можуть використовувати цю техніку для атак, щоб отримати потрібну інформацію користувача.

4) Спуфінг HTTPS

Ця тактика передбачає створення підробленого сертифіката та надсилання його в браузер жертви під час підключення до веб-сайту, який спочатку вважається безпечним. Жертви обмінюються пакетами з сервером, думаючи, що пакети зашифровані та безпечні, хоча насправді це не так. Посередники можуть легко читати всі мережеві пакети [13].

На зображенні (рис. 1.7) представлена схема, що ілюструє принцип роботи Спуфінг HTTPS.

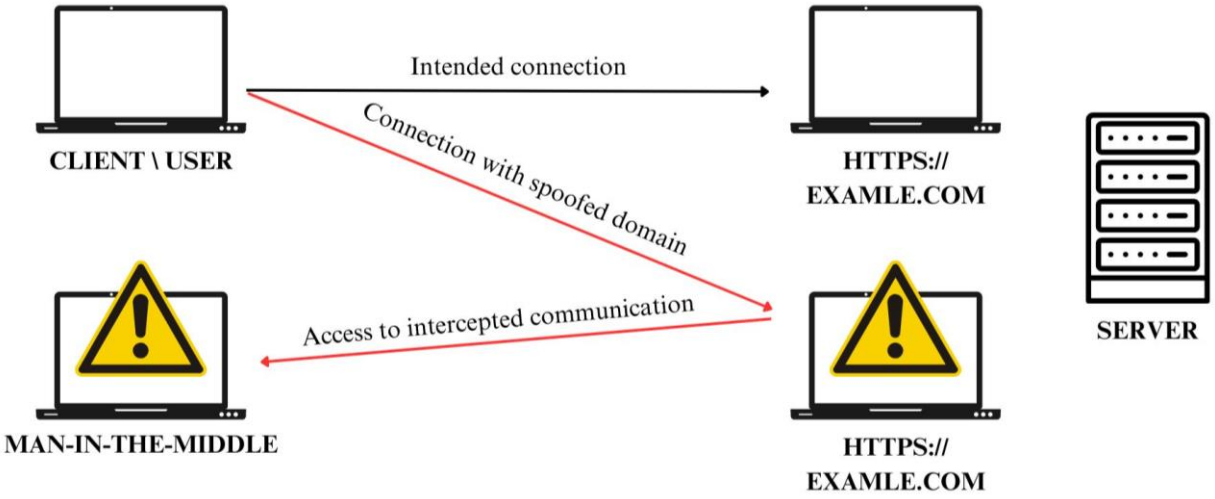

Рисунок 1.7 – Принцип роботи Спуфінг HTTPS

На ньому зображено три комп'ютери:

– комп'ютер користувача: цей комп'ютер зображений зліва і підписаний як "Користувач". Він намагається зв'язатися з веб-сайтом, який зображений праворуч;

– зловмисний комп'ютер: цей комп'ютер зображений посередині і підписаний як "Зловмисний комп'ютер". Він перехоплює трафік між комп'ютером користувача та веб-сайтом;

– веб-сайт: цей веб-сайт зображений праворуч і підписаний як "Веб-сайт". Він є кінцевою точкою, з якою намагається зв'язатися користувач.

Стрілки на діаграмі показують напрямок руху трафіку:

– користувач → зловмисний комп'ютер: ця стрілка показує, що комп'ютер користувача надсилає запит на веб-сайт зловмисному комп'ютеру;

– зловмисний комп'ютер → веб-сайт: ця стрілка показує, що зловмисний комп'ютер перенаправляє запит користувача на веб-сайт;

– веб-сайт → зловмисний комп'ютер: ця стрілка показує, що веб-сайт надсилає відповідь на запит користувача зловмисному комп'ютеру;

– зловмисний комп'ютер → користувач: ця стрілка показує, що зловмисний комп'ютер перенаправляє відповідь веб-сайту користувачеві.

Отже, на цій діаграмі чітко показано, як працює підробка HTTPS, і показано, як зловмисники можуть використовувати цю техніку для атаки на комп'ютери та мережі.

Тепер розглянемо застарілі варіанти. Є два способи розшифровки файлів, які наразі не дуже популярні. Це спричинено занадто складними комбінаціями та використанням технологій, які були витіснені більш сучасними.

1) Використання мережевого сніффера

Сніфери схожі на приховану камеру, яка сидить у темному кутку і отримує все, що відбувається в кімнаті. Потім той, хто встановлює це спостереження, отримує ці дані – і придумує, що робити далі.

Сніфери - це прикладне програмне забезпечення, вбудоване програмне забезпечення або апаратний модуль, який призначений для збору даних, надісланих із системи всієї мережі та до неї. Вони не втручаються в передачу даних, просто копіюють інтернет-пакети, до яких вони можуть отримати доступ. Не обов'язково шкідливий інструмент, сніфери можуть стати в нагоді мережевим адміністраторам, Інтернет-провайдерам і спеціалістам із безпеки. Однак вони також є потенційним інструментом злодіїв даних та інформації. Апаратні сніфери можуть бути (і іноді є) вбудованими в маршрутизатори, модеми та інші типи вузлів. Програмні сніфери можна інсталювати на серверах, проміжних пристроях і кінцевих комп'ютерах.

Щоб пояснити, що робить аналізатор пакетів (інша назва сніфера), окрім порівняння його з дорожнім контрольно-пропускним пунктом, де перевіряються всі транспортні засоби, що проїжджають, нам доведеться пояснити, як кінцеві точки отримують дані в нормальних умовах без залучення сніфера.

Зазвичай клієнт отримує та переглядає лише дані, призначені для його IP-адреси. Однак в межах однієї мережі «повз» кожного клієнта проходить набагато більше даних. Таким чином, на відміну від звичайної шпигунської програми, яка фіксує лише трафік, пов'язаний із певною кінцевою точкою, сніффер може реєструвати трафік із усієї мережі (або її доступної частини), таким чином, будучи набагато більш всеосяжним інструментом [14].

На зображенні (рис. 1.8) представлена схема, що ілюструє принцип роботи

Сніфера.

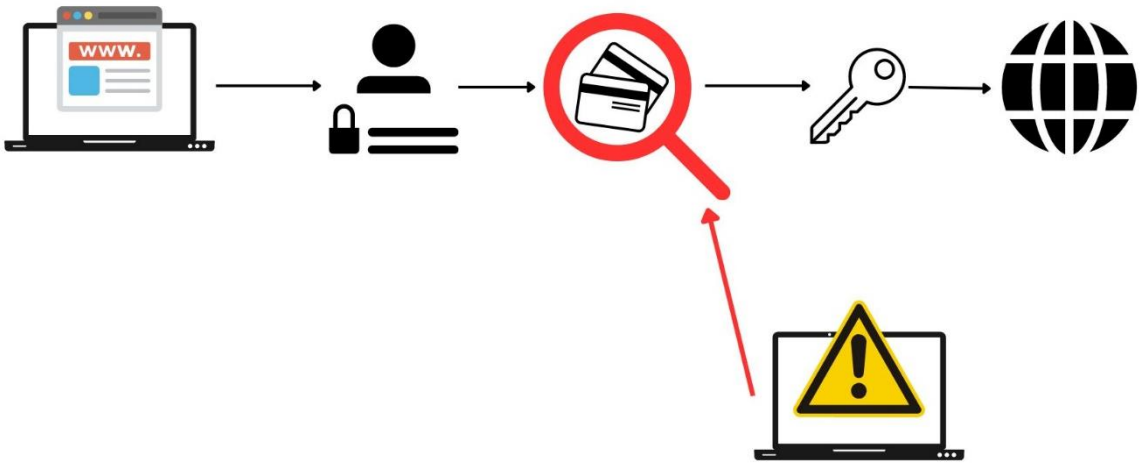

Рисунок 1.8 – Принцип роботи Сніфера

Локальні дротові мережі загального користування умовно можна розділити на дві групи, і специфіка роботи аналізаторів залежить від структури мереж. Таким чином, мережі-концентратори (де весь трафік протікає між усіма кінцевими точками) ідеально підходять для пасивного аналізу, який важко виявити, тоді як мережі комутаторів, які фільтрують дані (через їх величезні обсяги), вимагають додаткових дій з боку аналізатора. Аналізатору потрібно додати більше даних до трафіку, щоб контролювати весь потік даних. Таким чином, виявити такого активного сніфера набагато легше.

Моніторинг загального мережевого трафіку потребує спеціальних налаштувань (безладний режим) мережевого адаптера пристрою аналізатора. Сніфери також відрізняються за своєю здатністю збирати трафік, декодувати та аналізувати дані. У сніферах також є багато налаштувань. Оператор аналізатора може фільтрувати пакети для аналізу за різними критеріями.

2) Імітація вишки стільникового зв’язку

Ще більше зусиль потрібно для імітації веж стільникового зв’язку.

Конкретна інформація від телекомунікаційних компаній щодо спеціалізованого апаратного обладнання, частот і дозволів є дорогою та важкодоступною для широкого загалу.

Проте підключення жертви до вежі означає лише те, що її пакети будуть

отримані в зашифрованому вигляді (98% часу) [15].

Тобто, атака "людина посередині" (MITM) залишається актуальною загрозою, хоча її методи та наслідки добре вивчені. Зловмисники постійно розробляють нові методи обходу заходів безпеки, тому важливо бути пильними, слідкувани за новими методами та вживати додаткових заходів для захисту своїх даних.

1.5 Наслідки та небезпеки MITM-атак

MITM-атаки можуть мати серйозні негативні наслідки для користувачів та організацій. Ось деякі з найпоширеніших:

– втрата конфіденційних даних: зловмисники, які здійснюють MITM-атаку, можуть перехопити та прочитати будь-який трафік, який проходить через їхню мережу. Це може включати особисту інформацію, таку як паролі, номери кредитних карток, та медичні записи;

– фішингові атаки: зловмисники можуть використовувати MITM-атаки для створення фейкових веб-сайтів, які виглядають як легітимні. Ці веб-сайти можуть використовуватися для збору особистої інформації від користувачів або для поширення шкідливого програмного забезпечення;

– втрата фінансів: MITM-атаки можуть використовуватися для крадіжки грошей з банківських рахунків або кредитних карток. Зловмисники можуть перехопити інформацію про транзакції та використовувати її для здійснення несанкціонованих платежів;

– втрата репутації: MITM-атаки можуть завдати шкоди репутації організації. Якщо буде виявлено, що організація не вжила належних заходів для захисту даних своїх клієнтів, це може призвести до втрати довіри та клієнтів;

– відмова в обслуговуванні: MITM-атаки можуть використовуватися для блокування доступу користувачів до веб-сайтів або онлайн-сервісів. Це може призвести до значних фінансових втрат для організацій, які залежать від онлайн-торгівлі.

Ось кілька додаткових небезпек MITM-атак:

– зміна даних: зловмисники можуть змінювати дані, які передаються між користувачем та сервером. Це може призвести до серйозних проблем, наприклад, до помилкових медичних діагнозів або фінансових втрат;

– шантаж: зловмисники можуть використовувати MITM-атаки, щоб шантажувати користувачів. Вони можуть перехопити особисту інформацію користувачів і погрожувати оприлюднити її, якщо користувачі не заплатять їм викуп;

– втрата контролю над комп'ютером: зловмисники можуть використовувати MITM-атаки, щоб отримати контроль над комп'ютером користувача. Це може дозволити їм встановлювати шкідливе програмне забезпечення, красти дані або використовувати комп'ютер для поширення спаму.

Наприклад, найвідоміша атака типу MITM сталася задовго до появи комп’ютерів та Інтернету і отримала назву «план Бабінгтона». На фото (рис. 1.9) показаний зразок шифрованого листування між Бабінгтоном та Марією Стюарт.

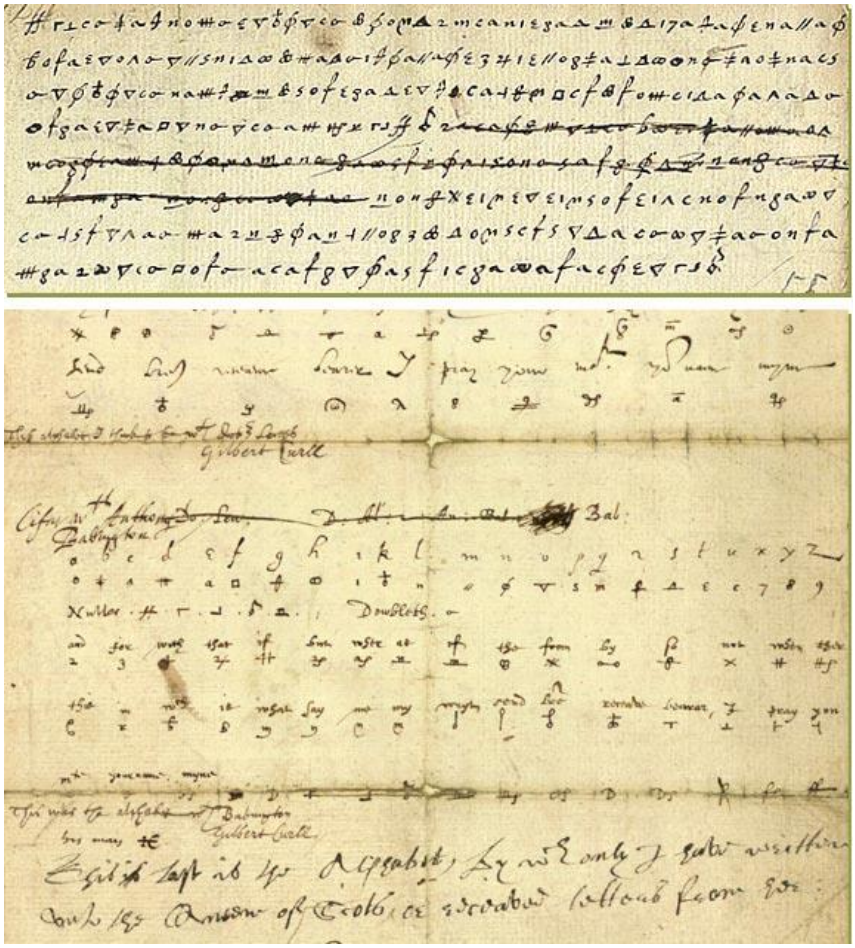

Рисунок 1.9 – Зразок шифрованого листування

У 1568 році прихильники ув'язненої Марії Стюарт, королеви Шотландії, написали шифрований лист, закликаючи її підтримати замах на королеву Англії Єлизавету I.

Відповіді Мері були перехоплені агентами Єлизавети, які змінили лист, щоб вийшло так, ніби Мері просила учасників змови назвати себе. Змовники відповіли на цей лист, додавши список імен усіх своїх співзмовників.

Цей лист знову був перехоплений «посередниками», що пізніше призвело до страти Марії та її послідовників [16].

Тепер розглянемо більш сучасні випадки:

1) Зламані облікові записи 10 000 користувачів Office 365

Популярним прикладом атаки типу «людина посередині» є злом облікових записів Office 365. Відома хакерська група «Lapsus$» часто використовує різні тактики для отримання несанкціонованого доступу до мереж і облікових записів. У 2022 році вони здійснили успішну атаку типу "людина посередині", спрямовану на понад 10 000 користувачів Office 365, підробивши цільову сторінку Office 365. Викравши облікові дані та сеансові файли cookie, зловмисники обійшли протоколи MFA та отримали несанкціонований доступ до облікових записів електронної пошти жертв, згодом використовуючи цей плацдарм для проведення кампаній BEC (компрометація бізнес-електронної пошти), націлених на інші організації. Хоча не було повідомлень про серйозні порушення, Microsoft використала цей випадок, щоб опублікувати звіт про те, як зазвичай відбувається ця атака та як організації можуть захистити себе від цих типів атак [17].

2) Фішинг Reddit і атака MITM

Під час нещодавнього інциденту 2023 року популярна платформа соціальних мереж Reddit стала об’єктом ретельно спланованої фішингової атаки на одного з її співробітників.

Перейшовши за посиланням, співробітники були спрямовані на оманливу, але переконливу копію інтернет-порталу Reddit.

Там вони вводили свої дані доступу та несвідомо передавали їх зловмисникам. В результаті стався витік контактної інформації сотень співробітників.

На щастя, хакери не отримали доступу до основних операційних систем Reddit, де зберігається більшість даних [18].

Тим не менш, цей інцидент став протверезним нагадуванням про постійну вразливість і проблеми підтримки надійних заходів кібербезпеки.

3) Документи НАТО, викриті в порушення урядом Португалії

У вересні 2022 року Міністерство оборони португальського уряду було причетне до порушення кібербезпеки, яке мало масштабний вплив.

Порушення пов’язане з несанкціонованим розголошенням конфіденційних документів НАТО, які продавалися в темній мережі.

Це тривожна ситуація, яка викликає занепокоєння далеко за межами наших кордонів.

У ході розслідування було встановлено, що злом стався через незахищений канал зв'язку. Це було критичним недоглядом, який дозволив зловмиснику отримати конфіденційну інформацію.

Що зробило цю атаку ще більш підступною, так це її прихований характер. Зловмисники діяли таємно, уникаючи виявлення, збираючи дані, надіслані через незахищені канали [19].

Тож, важливо пам'ятати, що MITM-атаки можуть мати серйозні наслідки для користувачів та організацій. Тому важливо вживати заходів для захисту від цих атак.

1.6 Існуючі методи захисту від MITM-атак

Атаки MITM важко виявити, і зловмисники можуть атакувати будь-кого в Інтернеті. Хакери можуть отримати облікові дані користувача клієнта, атакуючи веб-сайти та програми, які потребують автентифікації для входу. Вони також можуть націлюватися на компанії з веб-сайтами або програмами, які зберігають клієнтську або фінансову інформацію.

Розглянемо декілька методів та технологій, які можна використовувати для захисту бездротових мереж від MITM-атак:

1) Надійні та унікальні паролі

Ненадійні паролі або паролі, які легко вгадати, спрощують зловмисникам отримати неавторизований доступ до ваших облікових записів або мережі. Потрібно переконатись, що ваші паролі достатньо довгі та містять комбінацію великих і малих літер, цифр і спеціальних символів. Крім того, потрібно переконатись, що користувачі мають різні паролі для кожного облікового запису. Якщо один обліковий запис зламано, наявність унікальних паролів гарантує безпеку інших облікових записів.

Наприклад, StrongDM – це хмарна платформа доступу до інфраструктури, призначена для оптимізації керування доступом до інфраструктури.

Вона доповнює надійні паролі, пропонуючи додатковий рівень безпеки завдяки централізованому контролю доступу через RBAC, ABAC і PBAC . Цей доданий рівень гарантує, що навіть якщо користувач має надійний пароль, йому все одно знадобляться спеціальні дозволи для доступу до конфіденційних ресурсів. Крім того, доступ до ресурсів не вимагає від кінцевого користувача введення імені користувача та пароля. Це усуває розкриття облікових даних і зменшує ризик того, що зловмисник знайде дійсні облікові дані [20].

2) Двофакторна автентифікація

Двофакторна автентифікація (2FA) вимагає від користувача, надати другу форму перевірки, наприклад унікальний код, надісланий на мобільний пристрій, на додаток до пароля. Два методи перевірки значно знижують ризик несанкціонованого доступу, навіть якщо пароль зламано. Потрібно використовувати 2FA, де це можливо, особливо для облікових записів, які містять конфіденційну інформацію або надають доступ до фінансових транзакцій.

Ось приклад програм, які генерують одноразові паролі (OTP) для використання з різними веб-сайтами та службами:

– Google Authenticator [21];
– Authy (також включає підтримку біометричної аутентифікації).

3) Автентифікація без пароля

Безпарольна автентифікація повністю усуває довіру до паролів і використовує

альтернативні методи, такі як біометрія (розпізнавання відбитків пальців або обличчя) або апаратні маркери для підтвердження особи. Цей метод надійніше та зручніше аутентифікує користувачів і усуває ризик зламу або викрадення паролів – ефективно запобігаючи атакам MITM, націленим на слабкі або повторно використані паролі.

Наведені нижче програми пропонують різні методи автентифікації без пароля, такі як push-повідомлення, біометрична аутентифікація та апаратні ключі безпеки:

– Duo Security;

– Microsoft Authenticator [22].

4) Шифрування своїх даних

Шифрування захищає дані від перехоплення під час передачі, перетворюючи їх у нечитабельний формат, який можуть розшифрувати лише авторизовані сторони з відповідними ключами шифрування. Навіть якщо зловмисники перехоплять дані користувача, вони не зможуть розшифрувати інформацію.

Ось програми, які можна використовувати для шифрування файлів, дисків та розділів:

– VeraCrypt;

– BitLocker [23].

Криптографічний збій займає друге місце в списку 10 найбільших вразливостей OWASP за 2023 рік. Потрібно слідкувати за актуальністю методів і протоколів шифрування, щоб забезпечити найкращий захист своїх даних. Для додаткової впевненості, що ви передаєте конфіденційні дані через зашифроване з’єднання, потрібно знайти символ замка в адресному рядку веб-переглядача або використовуйте веб-сайти, URL-адреси яких починаються з «https://», щоб вказати, що з’єднання безпечне.

5) Обережність із громадськими мережами Wi-Fi

Загальнодоступні мережі Wi-Fi сумно відомі своєю вразливістю до атак MITM. Зловмисники можуть встановити фальшиві точки доступу або перехопити дані, що передаються через ці мережі. Підключаючись до загальнодоступної мережі Wi-Fi, будьте обережні та дотримуйтесь цих практичних порад:

– уникайте доступу до конфіденційної інформації або проведення фінансових операцій у загальнодоступних мережах Wi-Fi;

– використовуйте віртуальну приватну мережу (VPN), щоб зашифрувати підключення до Інтернету та захистити свої дані від перехоплення;

– перевірте законність мережі в закладі або персоналі місця перед підключенням.

Ось приклад VPN програм, що шифрує ваш трафік і приховує ваш IP-адресу, коли ви підключені до загальнодоступних мереж Wi-Fi:

– Avast SecureLine VPN;

– ExpressVPN.

Розуміння того, як запобігти атакам типу "людина посередині" під час використання загальнодоступних мереж Wi-Fi, значно зменшить ризик стати жертвою атак.

6) Перевірка сертифікатів SSL

Під час доступу до веб-сайтів або онлайн-сервісів перед введенням облікових даних або конфіденційної інформації потрібно перевіряти сертифікати SSL , щоб забезпечити безпечне з’єднання. Сертифікати SSL підтверджують автентичність веб-сайту та шифрують дані, що передаються між користувачем і сервером. Крім того, знайшовши символ замка в адресному рядку та натиснувши його – можна переглянути деталі сертифіката. Потрібно переконатись, що сертифікат видано надійним органом і відповідає веб-сайту, який ми відвідуємо [24].

7) Використання віртуальної приватної мережі

Віртуальна приватна мережа створює безпечний зашифрований тунель між вашим пристроєм та Інтернетом. Завдяки маршрутизації вашого інтернет-з’єднання через сервер VPN усі дані, що передаються на ваш пристрій і з нього, зашифровуються, що запобігає перехопленню зловмисниками чи маніпулюванню з’єднанням.

VPN корисні під час підключення до загальнодоступних мереж Wi-Fi або доступу до Інтернету з ненадійних місць. Переконайтесь, що ваш провайдер VPN пропонує надійні протоколи шифрування та не записує ваші дані.

8) Будьте уважні до спроб фішингу

Фішинг – це тактика, за допомогою якої зловмисники намагаються обманом змусити користувачів розкрити конфіденційну інформацію, часто видаючи себе за законні організації, такі як банки чи онлайн-сервіси, за допомогою підроблених електронних листів, веб-сайтів або повідомлень.

Потрібно навчитись, як запобігти атакам "людина посередині" від спроб фішингу за допомогою цих вказівок:

– будьте обережні з небажаними електронними листами або повідомленнями, які просять надати особисту інформацію;

– перевірте легітимність веб-сайтів, вручну ввівши URL-адресу замість натискання посилань;

– ретельно перевірте відправників електронної пошти та знайдіть підозрілі або неправильно написані адреси електронної пошти;

– дізнавайтесь про новітні методи фішингу та будьте в курсі потенційних загроз.

Наприклад, є Google Safe Browsing – це безкоштовна служба від Google, яка попередить вас, якщо ви відвідаєте веб-сайт, який підозрюється у фішингу або шкідливому програмному забезпеченні [25].

Та PhishTank – це веб-база даних, яка містить інформацію про відомі фішингові веб-сайти. Ви можете перевірити URL-адресу веб-сайту, щоб побачити, чи є він у списку.

9) Регулярно перевірка своєї мережі

Регулярний моніторинг мережі може допомогти виявити та запобігти атакам типу "людина посередині". Аналізуючи мережевий трафік, ви можете виявити будь-які аномалії або підозрілі дії, які можуть свідчити про поточну атаку. Розгляньте можливість впровадження систем виявлення або запобігання вторгненням, які відстежуватимуть вашу мережу на наявність будь-яких ознак неавторизованого доступу чи зв’язку.

Крім того, ведення журналів мережевої активності може надати цінну інформацію в разі атаки. Ці журнали можна використовувати, щоб відстежити

джерело атаки, виявити скомпрометовані системи та вжити відповідних заходів для зменшення збитку.

10) Оновлення програмних забезпечень та пристроїв

Потрібно регулярно оновлювати та виправляти програмне забезпечення та пристрої, щоб усунути вразливості, якими можуть скористатися зловмисники. Оновлення систем гарантує, що ви використовуєте найновіші заходи безпеки, зменшуючи ризик атаки. Це включає оновлення операційної системи та всіх програмних програм, які використовуєте, наприклад веб-браузерів, антивірусного програмного забезпечення та плагінів. Зловмисники часто націлюються на застаріле програмне забезпечення, яке, швидше за все, має відомі вразливості, якими можна скористатися.

Отже, аналіз цих методів захисту від MITM-атак показує, що до кожного метода - потрібно мати окрему програму.

Розглянувши кожний метод, було вирішено створити програмне забезпечення, яке буде в собі об'єднувати декілька ключових методів захисту від атаки "людина посередині". Такий підхід дозволить отримати повноцінний програмний продукт без необхідності використання потужного апаратного забезпечення та скачування декількох окремих програм.

Додаток буде містити функції: виявлення та захисту від атаки типу Man-in-the-Middle (MITM) у бездротових мережах, включаючи підроблені SSL-сертифікати, аномальні ARP-запити та шифрування/дешифрування файлів.

Крім того, створення додатку дає нам можливість постійно вдосконалювати його та додавати нові функції і оновлення версій, відповідно до потреб користувача та нових тенденцій у MITM-атак.

Загалом, створення власної програми для аналізу, виявлення та захисту від MITM у бездротових мережах, включаючи підроблені SSL-сертифікати, аномальні ARP-запити, зміни в мережевому трафіку та шифрування/дешифрування файлів - дозволить нам забезпечити унікальний функціонал та зручний інтерфейс для користувачів.

1.7 Висновок та постановка задач

У цьому розділі було досліджено теоретичні основи захисту бездротових мереж від MITM-атак. Було розкрито актуальність та значущість дослідження даної тематики, проаналізовано сутність та класифікацію MITM-атак, а також особливості їх реалізації в бездротових мережах. Окремо розглянуто методи та інструменти, які використовуються зловмисниками для здійснення подібних атак, а також описано їхні наслідки та небезпеки. На завершення - було представлено огляд існуючих методів захисту від MITM-атак.

Отже, на основі проведеного дослідження можна зробити наступні висновки:

– MITM-атаки становлять серйозну загрозу для безпеки бездротових мереж, оскільки вони дозволяють зловмисникам перехоплювати та маніпулювати трафіком, що передається між користувачами та легітимними серверами;

– існує декілька типів MITM-атак, які відрізняються один від одного методами реалізації та використовуваними інструментами;

– специфіка бездротових мереж робить їх більш уразливими до MITM-атак через низку факторів, таких як відкритий доступ до середовища передачі даних та обмежені можливості аутентифікації та шифрування;

– зловмисники використовують різні методи та інструменти для реалізації MITM-атак, включаючи фейкові точки доступу, атаки типу "людина посередині", атаки з використанням протоколу SSL/TLS.

Проаналізувавши переваги та недоліки вже існуючих програм для виявлення та захисту від атаки типу Man-in-the-Middle (MITM) у бездротових мережах було визначено наступні завдання, які необхідно виконати для розробки програмного забезпечення:

– підключення до мережі для моніторингу пакетів;

– виявлення та захист мережі від аномальних ARP-запитів;

– виявлення підроблених SSL-сертифікатів;

– шифрування та дешифрування файлів з конфіденційною інформацією.

2 ПРОЕКТУВАННЯ ПРОГРАМИ ДЛЯ АНАЛІЗУ, ВИЯВЛЕННЯ ТА ЗАХИСТУ ВІД АТАКИ ТИПУ MITM У БЕЗДРОТОВИХ МЕРЕЖАХ

2.1 Розробка моделі програми

Діаграми діяльності - це візуальний інструмент, який використовується для опису послідовності дій, що відбуваються в процесі. Вони складаються з наступних елементів:

– дії (actions): позначають окремі кроки або завдання, які виконуються в рамках процесу. Відображаються у вигляді прямокутників з округленими кутами;

– рішення (decisions): показують умови або розгалуження, які впливають на подальший хід процесу. Зображуються у вигляді ромбів з декількома гілками, що вказують на можливі варіанти;

– вибір (choice): використовується для позначення альтернативних шляхів або дій, з яких користувач може вибрати один. Відображається у вигляді горизонтальної лінії, розгалуженої на кілька гілок.

Діаграми діяльності є цінним інструментом при розробці інтерфейсу програмного додатку, адже вони дозволяють:

– візуалізувати послідовність дій та взаємодію об'єктів. Це допомагає розробнику чітко уявити, як користувач буде взаємодіяти з програмою;

– виявити проблемні ситуації та помилки. Завдяки візуальному представленню можна легше знайти недоліки та внести необхідні зміни;

– поліпшити комунікацію зі стейкхолдерами. Діаграми дають чітке уявлення про процес, що полегшує обговорення та прийняття рішень;

– підвищити зручність використання інтерфейсу. Діаграми допомагають зосередитися на важливих аспектах взаємодії з користувачем та раціонально впорядкувати функціональність програми [26].

Таким чином, діаграми діяльності є ефективним інструментом, який допомагає розробити зручний та інтуїтивно зрозумілий інтерфейс програмного додатку.

Тому, виконаємо розробку структури інтерфейсу програмного додатку з використанням діаграм діяльності (рис. 2.1), для того щоб отримати: чіткість та логічність програми для аналізу, виявлення та захисту від атаки типу Man-in-the-Middle у бездротових мережах.

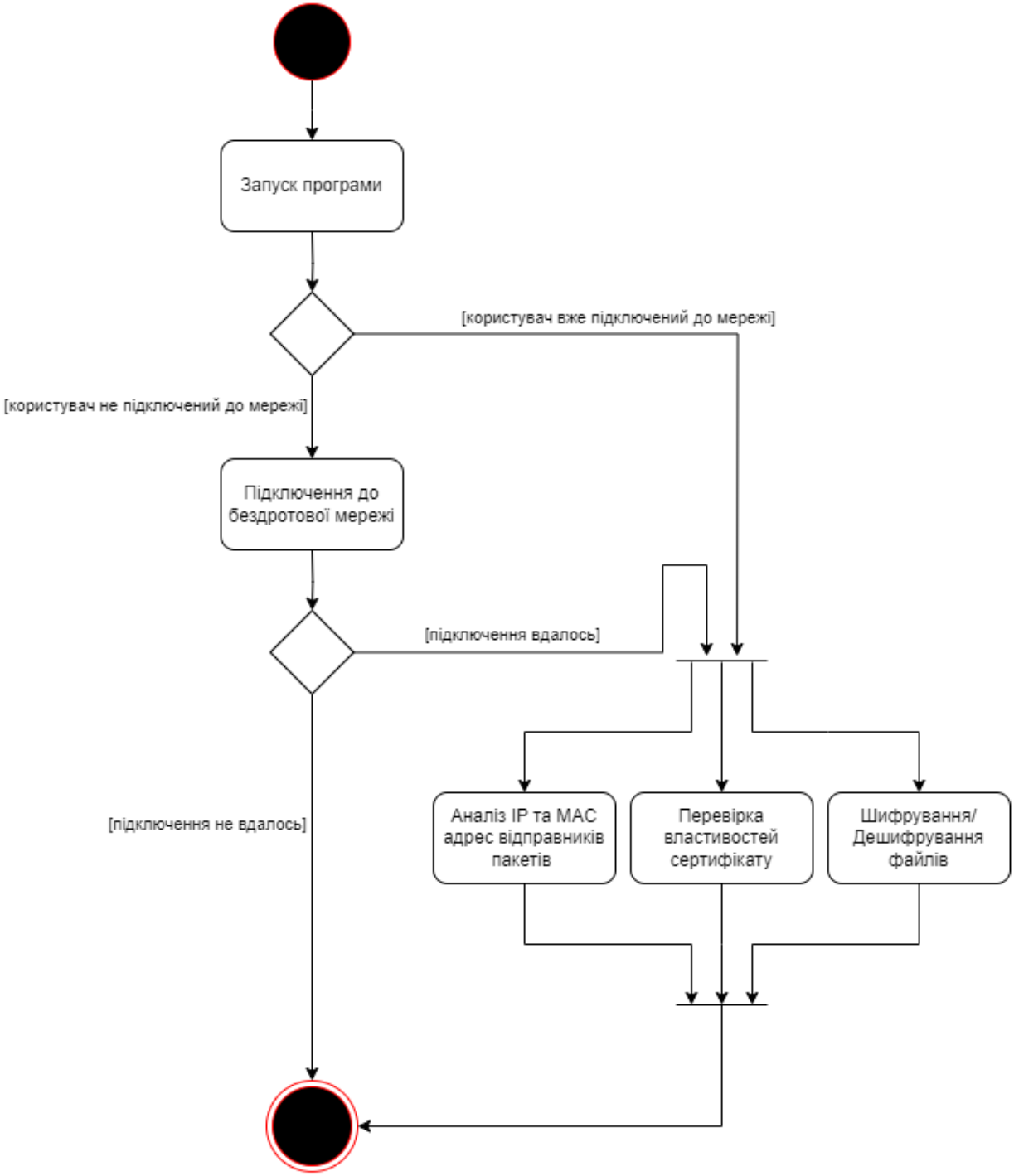

Рисунок 2.1 – Діаграма діяльності

2.2 Розробка алгоритмів роботи програми

Для реалізації програми для аналізу, виявлення та захисту від атаки типу Man-in-the-Middle (MITM) у бездротових мережах, включаючи підроблені SSL-сертифікати, аномальні ARP-запити-відповіді та зміни в мережевому трафіку потрібно створити окремі блок-схеми, що будуть визначати послідовність кроків та взаємодій, щоб забезпечити правильне та ефективне функціонування системи.

Для початку, розглянемо алгоритм підключення до бездротової мережі (рис. 2.2):

Крок 1. Початок:

Користувач запускає програму для підключення до бездротової мережі.

Крок 2. Пошук бездротових мереж:

Програма сканує доступні бездротові мережі в діапазоні дії користувача.

Крок 3. Аналіз сигналу знайдених бездротових мереж:

Програма аналізує силу сигналу кожної знайденої мережі.

Крок 4. Вибір бездротової мережі:

Користувач вибирає мережу, до якої хоче підключитися, з переліку знайдених.

Крок 5. Перевірка необхідності введення даних:

Система перевіряє, чи потребує обрана мережа введення даних для підключення.

Крок 5.1. Так:

Користувачу пропонується ввести необхідні дані, такі як пароль або ім'я мережі.

Крок 5.2. Ні:

Система автоматично намагається підключитися до мережі.

Крок 6. Підключення пройшло успішно?:

Крок 6.1. Так:

Користувачу відображається повідомлення про успішне підключення до мережі.

Крок 6.2. Ні:

Система видає повідомлення про помилку та пропонує користувачу спробувати підключитися до іншої мережі або повторити спробу з тією ж мережею.

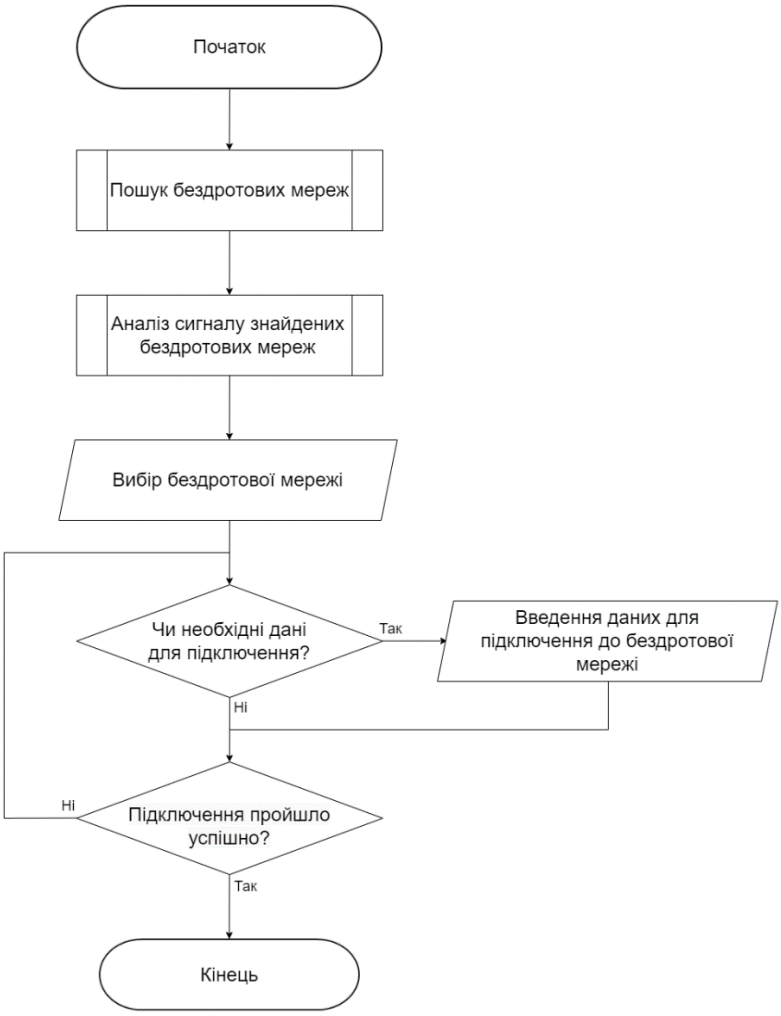

Рисунок 2.2 – Блок-схема підключення до БМ

Тепер можна розглянути алгоритми головного функціювання програми. Почнемо з моніторингу пакетів бездротової мережі (рис. 2.3):

Крок 1. Початок:

Процес моніторингу розпочинається.

Крок 2. Моніторинг пакетів бездротової мережі:

Система моніторить трафік в бездротовій мережі, записуючи всі передані пакети.

Крок 3. Аналіз IP та MAC адрес відправників пакетів:

Система аналізує IP та MAC адреси відправників пакетів, щоб визначити джерело трафіку.

Крок 4. Перевірка MAC адреси в списку доступу:

Система перевіряє MAC адресу відправника в списку доступу.

Крок 4.1. Так:

Якщо MAC адреса знаходиться в списку доступу, пакет дозволяється.

Крок 4.2. Ні:

Якщо MAC адреса не знаходиться в списку доступу, система блокує запит користувача.

Крок 5. Відображення некоректного запиту:

Якщо запит користувача заблоковано, система відображає повідомлення про некоректний запит.

Крок 6. Кінець:

Процес моніторингу пакетів завершується.

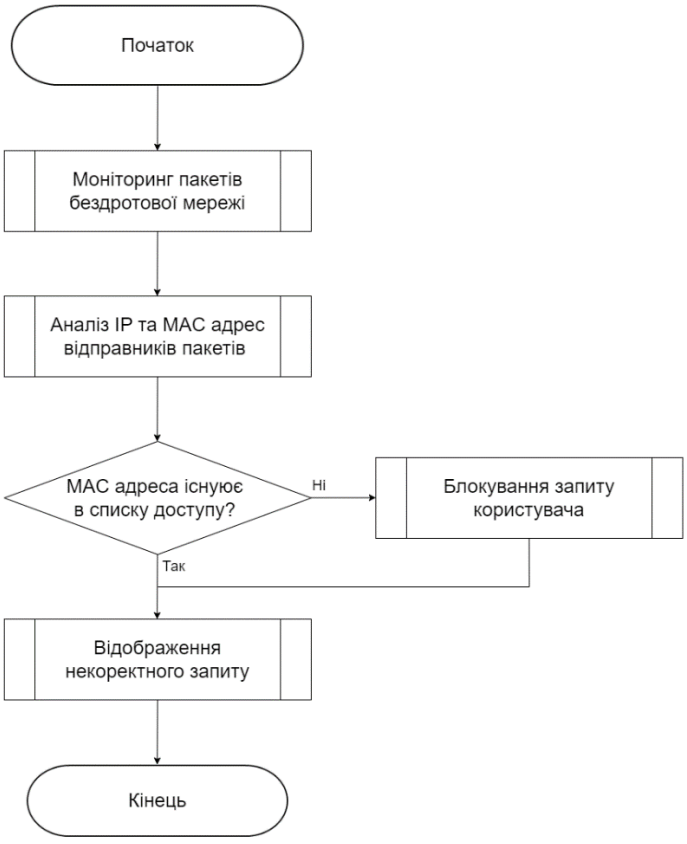

Рисунок 2.3 – Блок-схема моніторингу пакетів БМ

Переходимо до розглядання алгоритму перевірки на валідність SSL сертифікату (рис. 2.4):

Крок 1. Початок:

Користувач розпочинає процес отримання SSL сертифіката.

Крок 2. Завантаження SSL сертифіката:

Користувач завантажує SSL сертифікат з веб-сайту постачальника сертифікатів.

Крок 3. Завантаження SSL сертифіката:

Користувач перевіряє властивості сертифіката, такі як термін дії, тип сертифіката, доменне ім'я та видавець.

Крок 4. Перевірка властивостей сертифіката:

Користувач встановлює SSL сертифікат на сервер.

Крок 5. Перевірка сертифіката:

Сервер перевіряє дійсність сертифіката.

Крок 5.1. Сертифікат дійсний:

Користувачу відображається повідомлення про успішну установку сертифіката.

Крок 5.2. Сертифікат не дійсний:

Користувачу відображається повідомлення про помилку.

Крок 6. Кінець:

Процес отримання SSL сертифіката завершується.

Рисунок 2.4 – Блок-схема перевірка властивостей SSL сертифікату

І наостанок, побудуємо алгоритм шифрування та дешифрування (рис.2.6) файлів.

Функціонал шифрування та дешифрування є важливим інструментом для захисту даних. Він використовується для забезпечення конфіденційності, цілісності та автентичності інформації. Шифрування перетворює дані в незрозумілий формат, який може прочитати лише той, хто має ключ дешифрування. Це гарантує, що навіть якщо зловмисник отримає доступ до даних, він не зможе їх прочитати.

Функціонал дешифрування використовується для перетворення зашифрованих даних назад у зрозумілий формат. Цей функціонал використовується в різних сферах, таких як захист паролів користувачів, шифрування електронних листів та захист даних на жорстких дисках.

Отже, алгоритм шифрування (рис. 2.5):

Крок 1. Початок:

Користувач ініціює процес шифрування.

Крок 2. Вибір файлу для шифрування:

Користувач вибирає файл, який хоче зашифрувати, на своєму комп'ютері.

Крок 3. Введення ключа шифрування:

Користувач вводить ключ шифрування, який буде використовуватися для шифрування файлу.

Крок 4. Процес шифрування файлу:

Система використовує введений ключ для шифрування файлу.

Крок 5. Завантаження зашифрованого файлу:

Користувач може завантажити зашифрований файл на свій комп'ютер.

Крок 6. Кінець:

Процес шифрування завершується.

Рисунок 2.5 – Блок-схема шифрування файлу

Алгорит дешифрування (рис. 2.6):

Крок 1. Початок:

Користувач ініціює процес дешифрування.

Крок 2. Завантаження файлу для дешифрування:

Користувач вибирає файл, який хоче дешифрувати, і завантажує його в систему.

Крок 3. Введення ключа дешифрування:

Користувач вводить ключ дешифрування, який використовується для розшифровки файлу.

Крок 4. Процес дешифрування файлу:

Система використовує введений ключ для дешифрування файлу.

Крок 5. Завантаження дешифрованого файлу:

Користувач може завантажити дешифрований файл на свій комп'ютер.

Крок 6. Кінець:

Процес дешифрування завершується.

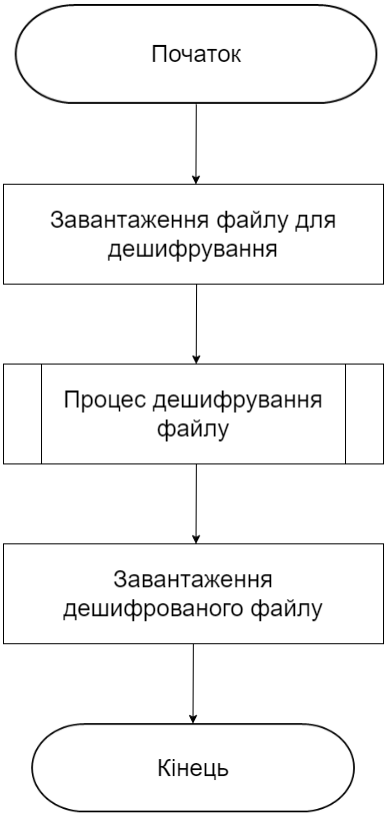

Рисунок 2.6 – Блок-схема дешифрування файлу

2.3 Проектування інтерфейсу програми

Важливим етапом проектування інтерфейсу програми є розробка її структури. Цей етап потребує розуміння потреб користувачів та ефективного розміщення елементів. Основна мета - створити зручну та зрозумілу для користувачів платформу.

Юзабіліті визначає зручність та привабливість програми. Стандарти юзабіліті включають правила, які роблять інтерфейс зрозумілим, простим та без зайвих ускладнень:

– простота та зрозумілість: чітке та логічне розташування елементів;
– консистентність: однорідний дизайн та поведінка елементів;
– ефективність: швидкий та легкий доступ до необхідних функцій [27].

При розробці програми для аналізу MITM-атак, шифрування та дешифрування файлів були використані наступні принципи:

– інформативність: відображення даних про стан мережі, результати аналізу та статусу шифрування/дешифрування;

– інтерактивність: надання можливості налаштовувати параметри аналізу, вчиняти дії для захисту від атак та керувати шифруванням;

– ефективність: чітке, лаконічне та своєчасне відображення інформації.

Структура сторінки програми має бути логічною та зручною для користувачів, щоб вони могли легко орієнтуватися та знаходити потрібну інформацію та потрібний функціонал.

Структура інтерфейсу програмного додатку включає кілька ключових компонентів, які взаємодіють один з одним і забезпечують зручність і надійність для користувача під час використання програми.

Розглянемо основні компоненти структури інтерфейсу:

1) Головний екран (рис. 2.7). Верхній колонтитул містить назву програми та основні елементи навігації. Також, з самого права – є кнопка «Додаткова інформація». У центрі уваги, великий блок – де буде відображатись моніторинг пакетів бездротової мережі.

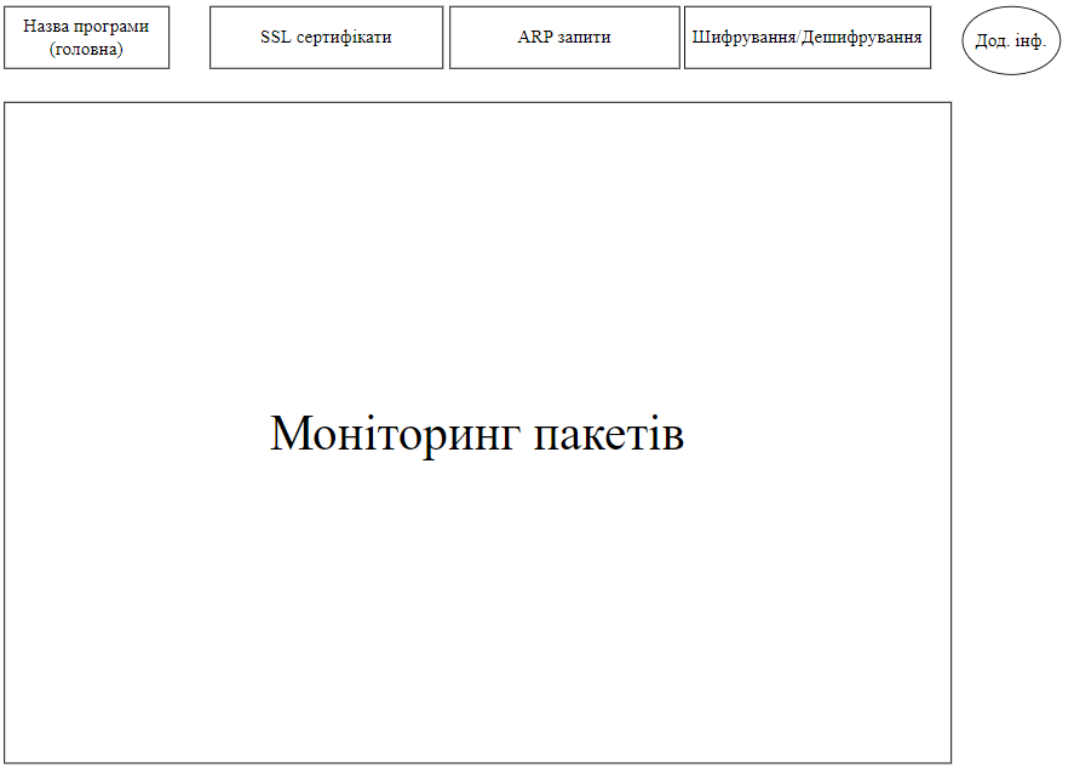

Рисунок 2.7 – Структура головної сторінки

2) Навігаційне меню. Навігація має бути логічною, зрозумілою та простою, щоб користувачі, які відкривають програму вперше, могли швидко зрозуміти, де знайти потрібний функціонал.

Загалом структури навігацій можна розділити на три основні типи: стандартна, каскад та павутина. В нашій програмі, ми будемо використовувати «Павутинну» структуру (рис. 2.8). Адже, посилання на вкладки організовані таким чином, що кожна сторінка містить посилання на інші сторінки. Така структура дозволяє користувачам легко перемикатися з однієї вкладки на іншу.

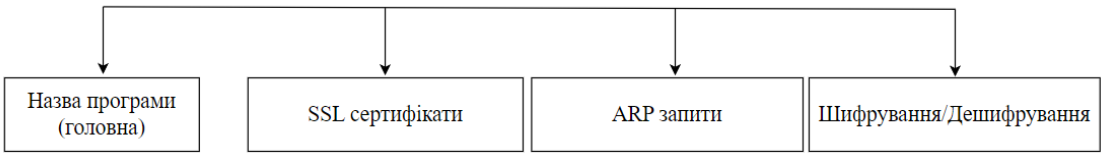

Рисунок 2.8 – Структура навігації

3) Екран підключення до бездротової мережі (рис. 2.9). Центральний екран змінюється і з’являється два блоки. У верхньому буде відображатись перелік бездротових мереж в доступному радіусу для підключення. У нижньому блоці потрібно буде ввести логін та пароль для підключення.

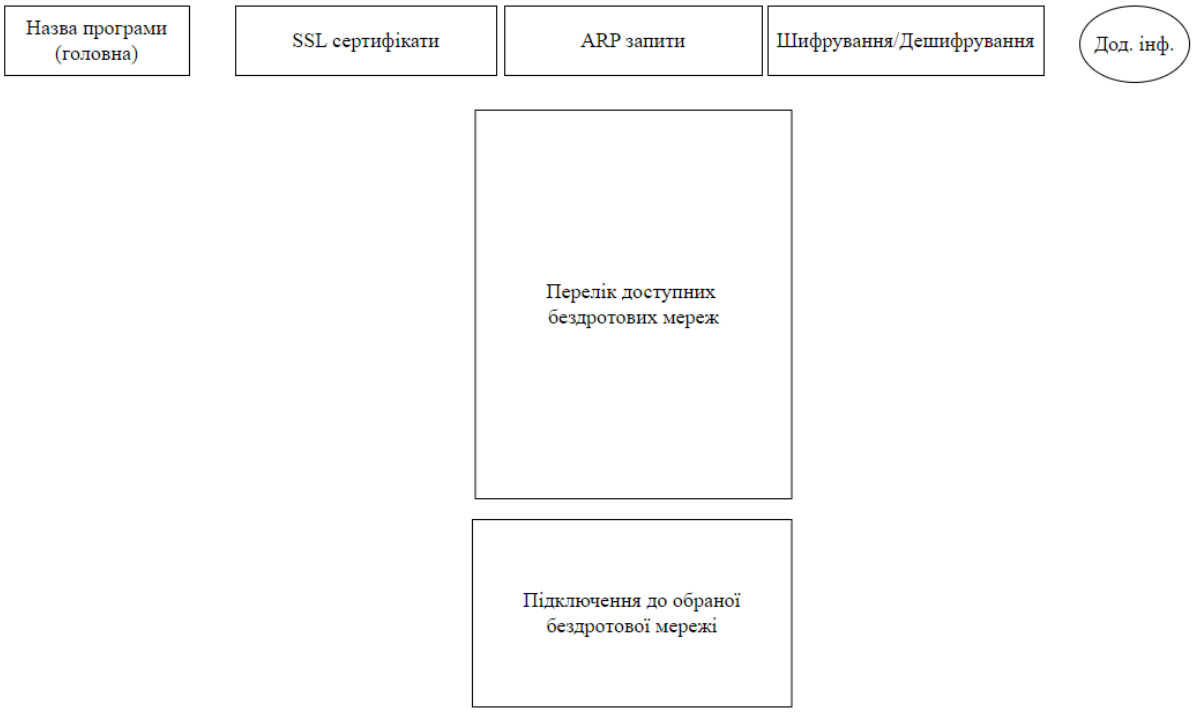

Рисунок 2.9 – Структура підключення до БМ

Після підключення, буде відображатись інформація про підключену бездротову мережу (рис. 2.10) з можливістю завантажити «XML профіль точки доступа» на комп’ютер.

Рисунок 2.10 – Структура інформації про підключену БМ

4) Екран для перевірки валідності SSL сертифікатів (рис. 2.11). У верхньому блоці потрібно вставити посилання на сайт, по якому ми хочемо зробити перевірку. У нижньому блоці з’явиться інформація про SSL сертифікати даного сайта.

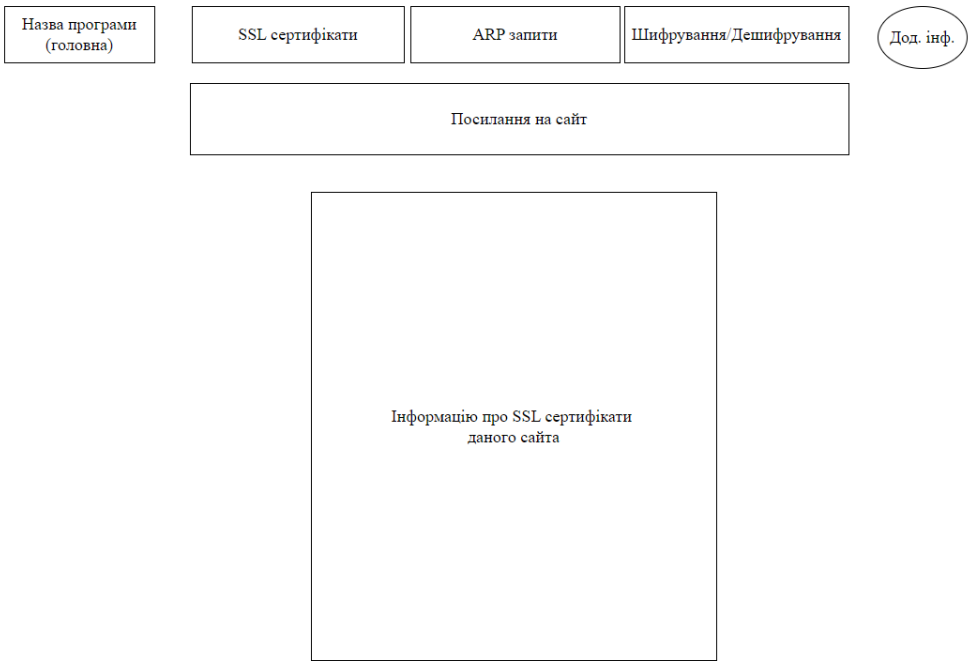

Рисунок 2.11 – Структура перевірки SSL сертифікатів

5) Екран для шифрування/дешифрування файлів (рис. 2.12). Головний екран буде умовно поділений на дві частини: шифрування та дешифрування.

Для шифрування потрібно буде завантажити файл, натиснути на кнопку «Зашифрувати» і на виході користувач отримує початковий зашифрований файл.

Такий самий принцип і з дешифруванням. Для дешифрування потрібно буде завантажити зашифрований файл, натиснути на кнопку «Дешифрувати» і на виході користувач отримує початковий розшифрований файл.

Якщо програма встановлена на комп'ютерах двох різних користувачів, це дає їм можливість обмінюватись конфіденційною інформацією без ризиків на витік даних. Файл, який буде зашифрований програмою, можна буде розшифрувати тільки за допомогою цієї самої програми.

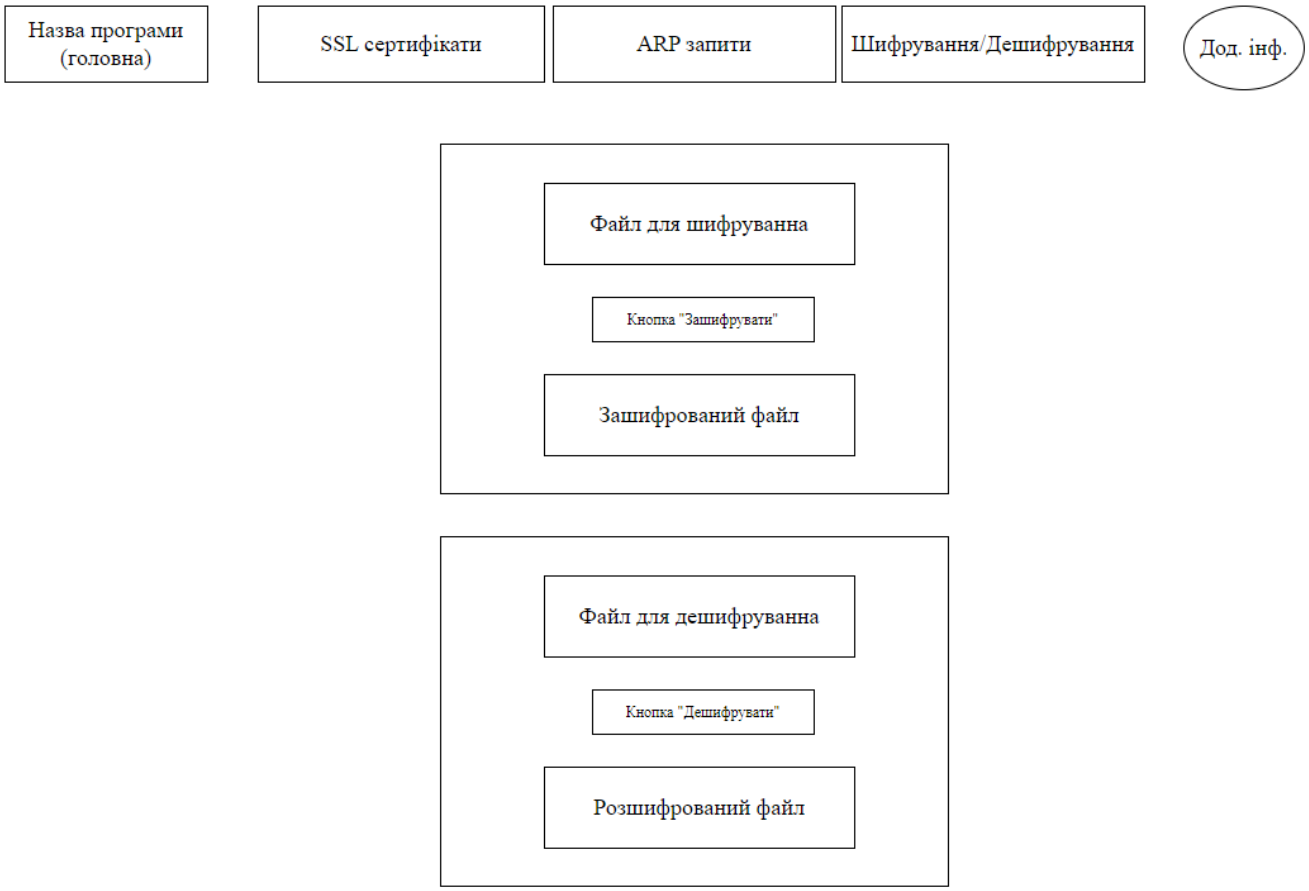

Рисунок 2.11 – Структура шифрування/дешифрування файлів

6) Додаткова інформація та зворотній зв'язок користувача. В цьому блоці буде відображатись загальна інформація, така як: назва програми, поточна версія програми, мінімальна конфігурація та пошта для зворотного зв'язку.

Отже, успішне проектування інтерфейсу програми аналізу MITM-атак, шифрування та дешифрування файлів потребує розуміння принципів юзабіліті та їх застосування для створення зручного та ефективного інструменту для користувачів.

2.4 Висновок

У другому розділі було створено модель програми, визначення алгоритмів її роботи та проектування інтерфейсу, що дозволить забезпечити ефективну і зручну роботу з інструментом для кінцевих користувачів.

Розробка моделі програми базувалася на аналізі основних методів здійснення MITM-атак та їхніх проявів у бездротових мережах. Була створена архітектура, яка включає модулі для моніторингу мережевого трафіку, виявлення підозрілих активностей, зокрема підроблених SSL-сертифікатів і аномальних ARP-запитів-відповідей. Модель також передбачала інтеграцію засобів шифрування та дешифрування файлів, що дозволяє підвищити рівень захисту переданих даних.

Розроблені алгоритми роботи програми забезпечують ефективне виявлення атак MITM за допомогою аналізу трафіку та ідентифікації аномалій. Алгоритми виявлення підроблених SSL-сертифікатів дозволяють своєчасно реагувати на спроби компрометації безпеки з'єднань, а виявлення аномальних ARP-запитів-відповідей забезпечує захист від спуфінгу в локальних мережах. Використання шифрування та дешифрування файлів допомагає зберегти конфіденційність даних навіть у разі успішної атаки.

Проектування інтерфейсу програми було спрямоване на створення інтуїтивно зрозумілого і зручного для користувача середовища, яке дозволяє швидко реагувати на виявлені загрози та здійснювати необхідні дії для їх нейтралізації. Інтерфейс забезпечує доступ до основних функцій програми, Таким чином, створена програма є потужним інструментом для забезпечення безпеки бездротових мереж від атак типу MITM, що значно підвищує загальний рівень захисту інформації користувачів.

3 РОЗРОБКА ПРОГРАМНОГО ЗАСТОСУНКУ

3.1 Вибір засобів для реалізації програми

В даний час існує багато різних мов програмування, на яких можна реалізувати цей проект, але в даному випадку була обрана мова програмування C#.

C# (вимовляється як «сі-шарп») — це сучасна об’єктно-орієнтована мова програмування. Вона є частиною широко відомої сім’ї мов C і знайомий кожному, хто користувався C, C++ або Java.

C# має кілька функцій, які забезпечують надійність і стабільність програми:

– автоматичне збирання сміття звільняє пам'ять, зайняту невикористаними об'єктами;

– обробка винятків забезпечує структурований підхід до виявлення та відновлення помилок;

– лямбда-вирази підтримують функціональне програмування;

– синтаксис запиту дозволяє маніпулювати даними з будь-якого джерела;

– асинхронні операції спрощують створення розподілених систем;

– відображення шаблонів забезпечує простий спосіб розділення даних і алгоритмів у сучасних розподілених системах.

Завдяки C# можна написати наступну програму:

– настільний додаток;

– сервіси;

– додаток для мобільних пристроїв;

– ігрова програма;

– бізнес-рішення.

Ключові переваги C#:

– низький бар'єр для входу: початківцям легко розпочати програмування, а досвідчені розробники можуть швидко підібрати нові проекти;

– добре розроблений: C# поєднує найкращі тенденції програмування з таких мов, як Java, C++, і постійно розвивається за допомогою нових дизайнів;

– читабельний синтаксис: полегшує читання коду інших людей і розуміння власного коду, написаного давно;

– багато бібліотек: бібліотеки доступні для різноманітних завдань, включаючи роботу з Excel, зображеннями та Discord [28].

Для написання коду було вибрано інтегроване середовище розробки Visual Studio Code.

Visual Studio — це стартовий майданчик для написання, налагодження, компіляції коду та інших опублікованих програм.

Це багатофункціональна програма, яка підтримує різні аспекти розробки програмного забезпечення.

Окрім стандартного редактора, Visual Studio містить компілятор, засоби завершення коду, графічний дизайнер та багато інших функцій, які спрощують процес розробки.

Visual Studio доступна для Windows і Mac, оптимізованими для розробки міжплатформних і мобільних програм [29].

Отже, було проведено обґрунтування вибору засобів для реалізації програми, для аналізу, виявлення та захисту від атаки типу Man-in-the-Middle (MITM) у бездротових мережах, включаючи підроблені SSL-сертифікати, аномальні ARP-запити-відповіді та зміни в мережевому трафіку.

3.2 Програмна реалізація

Для створення програми було використано паттерн MVVM (Model-View-ViewModel), який дозволяє відокремити логіку додатку від її візуальної частини (рис. 3.2.1). Даний паттерн є архітектурним, тобто він задає загальну архітектуру програми.

Рисунок 3.1 – Компоненти паттерну MVVM

Модель описує дані, які використовуються у даній програмі.

Модель може містити логіку, яка безпосередньо пов’язана з цими даними, наприклад логіку, яка перевіряє властивості моделі.

У той же час ця модель не повинна містити логіку, пов’язану з відображенням даних або взаємодією з візуальними елементами керування.

View визначає візуальний інтерфейс для взаємодії користувачів із програмою.

З точки зору WPF, представлення — це код у XAML, який визначає інтерфейс у формі кнопок, текстових полів та інших візуальних елементів.

ViewModel або модель презентації з’єднує модель і презентацію за допомогою механізму зв’язування даних. Коли змінюється значення властивості моделі. Вона також містить логіку для отримання даних із моделі та передачі цих даних у представлення та визначає логіку для оновлення даних у моделі [30].

Для початку розробимо отримання списку доступних бездротових мереж:

```
List<AccessPoint>list = new List<AccessPoint>();
if (_client.NoWifiAvailable)
{
  return list;
}

WlanInterface[] interfaces = _client.Interfaces;
foreach (WlanInterface wlanInterface in interfaces)
{
WlanAvailableNetwork[] availableNetworkList =
wlanInterface.GetAvailableNetworkList((WlanGetAvailableNetworkFlags)0);
List<WlanAvailableNetwork> list2 = new List<WlanAvailableNetwork>();
  WlanAvailableNetwork[] array = availableNetworkList;
  for (int j = 0; j < array.Length; j++)
  {
    WlanAvailableNetwork network = array[j];
    bool flag = !string.IsNullOrEmpty(network.profileName);
    if (!availableNetworkList.Where((WlanAvailableNetwork n) => n.Equals(network)    &&
!string.IsNullOrEmpty(n.profileName)).Any() || flag)
    {
     list2.Add(network);
    }
  }

  foreach (WlanAvailableNetwork item in list2)
  {
    list.Add(new AccessPoint(wlanInterface, item));
```

```
  }
}

return list;
```

Цей код визначає метод для отримання списку доступних бездротових мереж. Спочатку він створює порожній список точок доступу. Якщо немає доступних мереж, повертає цей порожній список. Інакше, проходить через усі інтерфейси бездротових мереж, які доступні клієнту, і для кожного інтерфейсу отримує список доступних мереж. Потім створює тимчасовий список для мереж, які мають ім'я профілю, або які унікальні серед інших мереж із ім'ям профілю. Додає ці мережі до тимчасового списку, а згодом додає їх як об'єкти AccessPoint до основного списку, який потім повертається як результат.

Тепер можемо написати підключення до бездротової мережі:

```
var accessPoints = _wifi.GetAccessPoints().OrderByDescending(ap => ap.SignalStrength);
var selectedIndex = int.Parse(AccessPointsListView.SelectedIndex);

if (selectedIndex > accessPoints.ToArray().Length || accessPoints.ToArray().Length == 0)
{
  return;
}

AccessPoint selectedAP = accessPoints.ToList()[selectedIndex];
AuthRequest authRequest = new AuthRequest(selectedAP);
bool overwrite = true;

if (authRequest.IsPasswordRequired)
{
  if (selectedAP.HasProfile)
  {
    overwrite = false;
  }
  else
  {
    authRequest.Password = PasswordPromptBox.Text;
  }
}

selectedAP.ConnectAsync(authRequest, overwrite, OnConnectedComplete);
```

Цей код відповідає за підключення до бездротової мережі. Спочатку отримується список точок доступу, упорядкований за силою сигналу в порядку

спадання. Потім визначається індекс вибраної точки доступу з інтерфейсу користувача. Якщо вибраний індекс перевищує кількість доступних точок або якщо список точок доступу порожній, метод припиняється. Інакше вибирається точка доступу за заданим індексом, створюється запит на авторизацію для цієї точки. За замовчуванням, дозволяється перезапис профілю. Якщо для з'єднання потрібен пароль, перевіряється, чи має точка доступу профіль. Якщо так, перезапис не потрібен, інакше пароль задається з поля вводу паролю. В кінці запускається асинхронне підключення до вибраної точки доступу з відповідними параметрами авторизації, і задається метод для обробки завершення підключення.

Тепер налаштовуємо блокування ARP-poisoning запиту з використанням білого списку:

```
string[] whiteList = { "a8:42:a1:5d:43:45", "1e:6f:6f:eb:fb:2f" };
var senderMac = arpPacket.SenderHardwareAddress.GetAddressBytes().ToString();

if (!whiteList.Contains(senderMac))
{
var packet = device.GetPacket();
var parsedPacket = Packet.ParsePacket(packet.LinkLayerType, packet.Data);

parsedPacket.ToString().Replace("Keep-Alive", "BLOCKED");
device.BlockRequests(senderMac);

return false;
}

return true;
```

Цей код налаштовує блокування запитів ARP-poisoning, використовуючи білий список. Спочатку створюється масив MAC-адрес, які є у білому списку. Потім отримується MAC-адреса відправника ARP-пакету і перетворюється на рядок. Якщо MAC-адреса відправника не входить до білого списку, отримується черговий пакет з пристрою, розбирається на рівні канального шару, і якщо у пакеті міститься рядок "Keep-Alive", він замінюється на "BLOCKED". Після цього пристрій блокує запити від цієї MAC-адреси і метод повертає значення false. Якщо ж MAC-адреса є у білому списку, метод повертає true, дозволяючи запит.

Наступним функціоналом є валідація SLL сертифіката:

```
var validCertificate = new X509Certificate2(_validCertificatePath);
var validChain = new X509Chain(validCertificate).Build();

if (validChain.ChainElements.Count != requestCertificateChain.ChainElements.Count)
{
  return false;
}

for (int i = 0; i < validChain.ChainElements.Count; i++)
{
if (validChain.ChainElements[i].Certificate.Thumbprint !=
requestCertificateChain.ChainElements[i].Certificate.Thumbprint)
  {
  sslPolicyErrors = SslPolicyErrors.RemoteCertificateChainErrors;
  }
}

if (sslPolicyErrors == SslPolicyErrors.None)
{
  return true;
}

return false;
```

Цей код виконує валідацію SSL-сертифіката. Спочатку завантажується дійсний сертифікат з файлу і будується його ланцюжок. Потім перевіряється, чи кількість елементів у цьому ланцюжку співпадає з кількістю елементів у ланцюжку сертифіката, що надійшов з запитом. Якщо кількість не співпадає, метод повертає false. Далі для кожного елемента ланцюжка порівнюються відбитки (thumbprints) сертифікатів. Якщо хоча б один з відбитків не співпадає, фіксується помилка політики SSL (SslPolicyErrors.RemoteCertificateChainErrors). Якщо жодних помилок політики SSL не виявлено, метод повертає true, підтверджуючи дійсність сертифіката. В іншому випадку повертається false, що означає недійсність сертифіката.

Тепер напишемо код для шифрування файлу:

```
public byte[] EncryptData(byte[] dataToEncrypt, byte[] key, byte[] iv)
    {
       using (Aes aesAlgorithm = Aes.Create())
       {
          aesAlgorithm.Key = key;
          aesAlgorithm.IV = iv;
```

```
            ICryptoTransform encryptor = aesAlgorithm.CreateEncryptor(aesAlgorithm.Key,
aesAlgorithm.IV);

            using (MemoryStream memoryStream = new MemoryStream())
            {
               using (CryptoStream cryptoStream = new CryptoStream(memoryStream, encryptor,
CryptoStreamMode.Write))
               {
                  cryptoStream.Write(dataToEncrypt, 0, dataToEncrypt.Length);
                  cryptoStream.FlushFinalBlock();
               }

               return memoryStream.ToArray();
            }
         }
      }
```

Цей код реалізує метод для шифрування даних за допомогою алгоритму AES. Спочатку створюється екземпляр алгоритму AES і встановлюються його ключ (key) та вектор ініціалізації (IV). Потім створюється об'єкт шифрування (encryptor) з використанням ключа та вектора ініціалізації. Після цього створюється потік пам'яті (MemoryStream), в який будуть записані зашифровані дані. Усередині цього потоку пам'яті створюється криптопотік (CryptoStream), що здійснює шифрування даних при запису. Дані для шифрування (dataToEncrypt) записуються у криптопотік, а після завершення запису викликається метод FlushFinalBlock, щоб завершити процес шифрування. Нарешті, зашифровані дані з потоку пам'яті перетворюються в масив байтів і повертаються як результат.

Та дешифрування файлу:

```
      public static byte[] DecryptData(byte[] dataToDecrypt, byte[] key, byte[] iv)
      {
         using (Aes aesAlg = Aes.Create())
         {
            aesAlg.Key = key;
            aesAlg.IV = iv;

            ICryptoTransform decryptor = aesAlg.CreateDecryptor(aesAlg.Key, aesAlg.IV);

            using (MemoryStream msDecrypt = new MemoryStream(dataToDecrypt))
            {
               using (CryptoStream csDecrypt = new CryptoStream(msDecrypt, decryptor,
CryptoStreamMode.Read))
               {
```

```
                byte[] decryptedBytes = new byte[dataToDecrypt.Length];
                int bytesRead = csDecrypt.Read(decryptedBytes, 0, decryptedBytes.Length);
                return decryptedBytes;
            }
        }
    }
}
```

Цей код реалізує метод для дешифрування даних, також, за допомогою алгоритму AES. Спочатку створюється екземпляр алгоритму AES і встановлюються його ключ (key) та вектор ініціалізації (IV). Потім створюється об'єкт для дешифрування (decryptor) з використанням ключа та вектора ініціалізації. Створюється потік пам'яті (MemoryStream) з даними для дешифрування (dataToDecrypt). Усередині цього потоку пам'яті створюється криптопотік (CryptoStream), який здійснює дешифрування даних під час читання. Далі, створюється масив байтів для збереження дешифрованих даних. Дані читаються з криптопотоку в цей масив байтів, і метод повертає дешифровані байти як результат.

3.4 Тестування роботи програми

Тестування функціональності програми — це процес визначення якісних показників щодо придатності продукту, бізнес-логіки та основної ідеї, для якої створено продукт.

Також визначається ступінь готовності проекту за всіма технічними та нетехнічними критеріями.

Одним із головних показників роботи програми – це навантаження системи. На рис. 3.2 видно що додаток майже не навантажує систему і використовує мало оперативної пам’яті.

Рисунок 3.2 – Тестування навантаження програми на ПК

Отже, відкривши програму нас зустрічає головний екран (рис. 3.3).

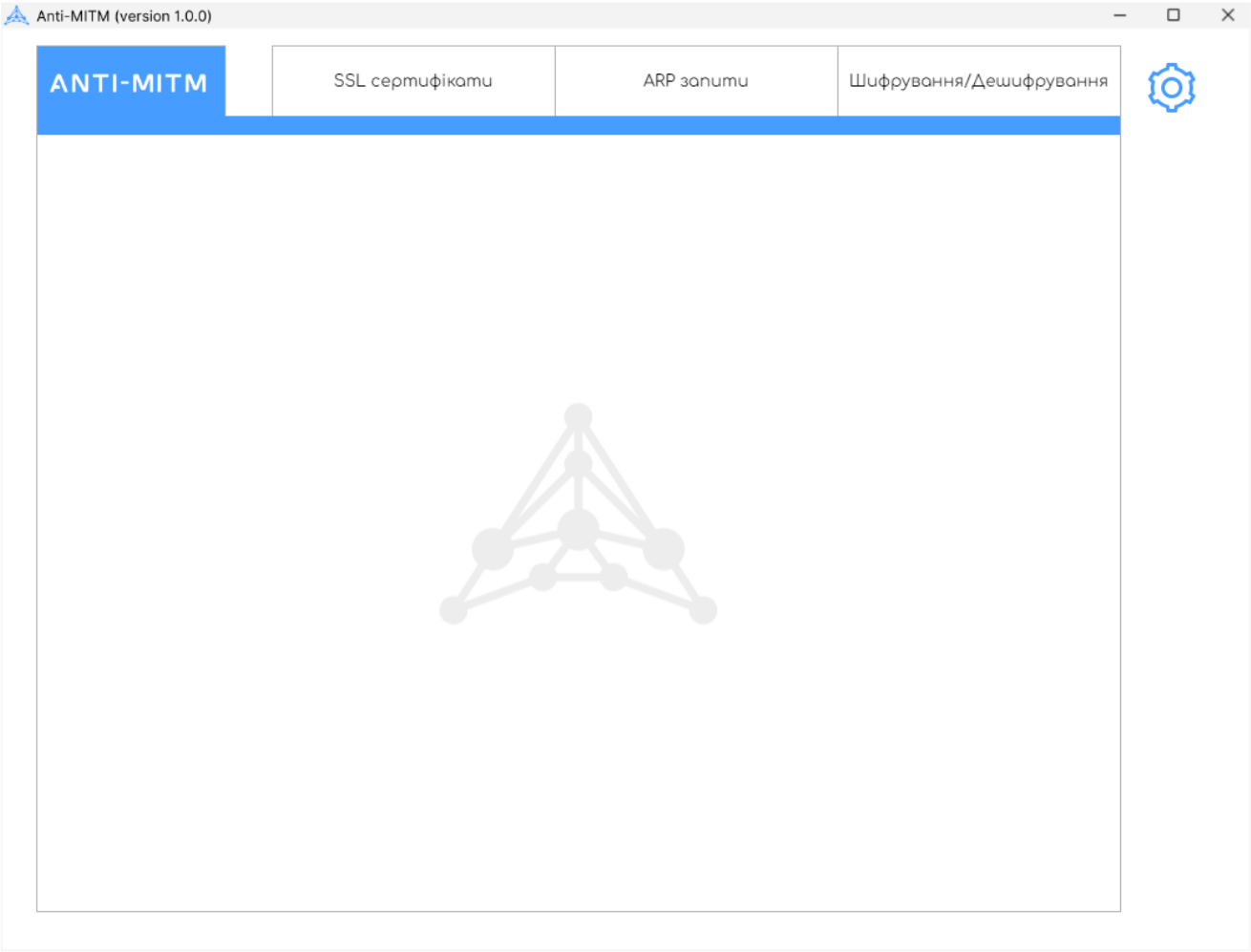

Рисунок 3.3 – Говна сторінка програми

Як бачимо, програма виконана у приємних та світлих кольорах: білий, сірий та синій.

Зверху, де навігація – вкладка, яка підписана назвою програми «ANTI-MITM» підсвічується синім і дає нам зрозуміти що ми наразі знаходимось на головній сторінці.

Також, є інші сторінки у навігації, а саме: «SSL сертифікати», «ARP запити» та «Шифрування/Дешифрування».

З самого права, є іконка «Налаштування», де буде виводитись додаткова інформація по програмі.

Головний простір програми поки пустий, адже ми ще не приєднались до бездротової мережі.

Тому перейдемо у вкладку, «ARP запити» (рис. 3.4)

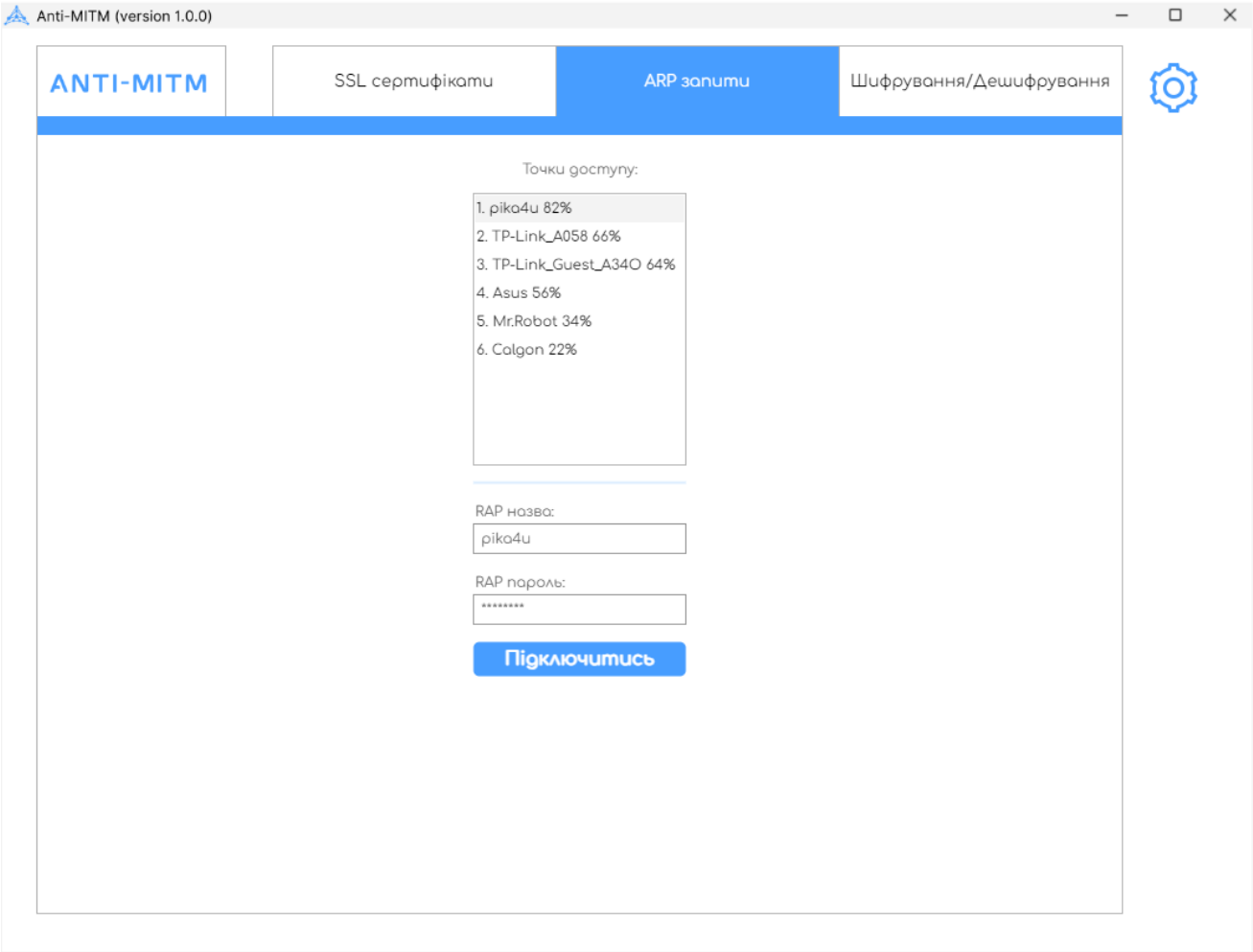

Рисунок 3.4 – Вкладка для підключення до БМ

Бачимо що в навігації синім кольором замалювалась вкладка «ARP запити».

У цій вкладці є перелік назв точок доступу, до яких можна приєднатись і рівень сигналу у відсоткову значенні та кнопка "Підключитись", яка дозволяє підключитися до вибраної точки доступу.

Нижче ми можемо ввести назву бездротової мережі та її пароль, для того щоб зробити підключення.

Для тестування, підключаємось до бездротової мережі «pika4u», вона має найкращий рівень сигналу – 82%.

Після підключення, ця вкладка змінює свій вид (рис. 3.5)

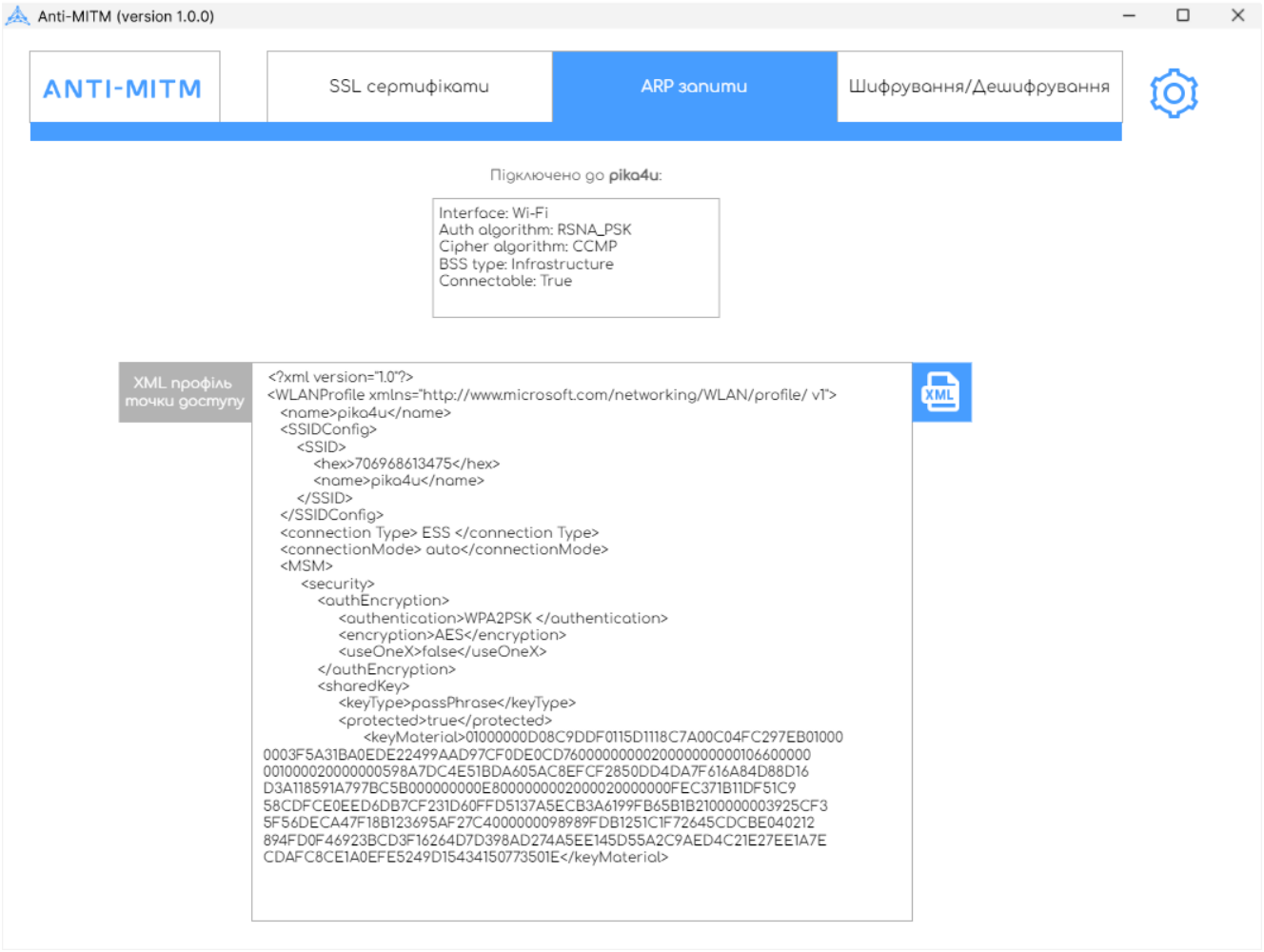

Рисунок 3.5 – Вкладка, коли ми підключитись до БМ

Тепер ми бачимо інформацію про мережу, до якої ми зробили підключення.

А саме:

– Інтерфейс: Wi-Fi
– Назва точки доступу (SSID): pika4u
– Тип шифрування: WPA2PSK
– Алгоритм шифрування: AES
– MAC-адреса: 01:00:00:00:D0:8C:9D:DF

Також, ми можемо завантажити на комп’ютер XML профіль точки доступу. Натиснувши на синю іконку «XML»

Перейшовши на головну сторінку, ми можемо побачити моніторинг пакетів даної бездротової мережі (рис. 3.6).

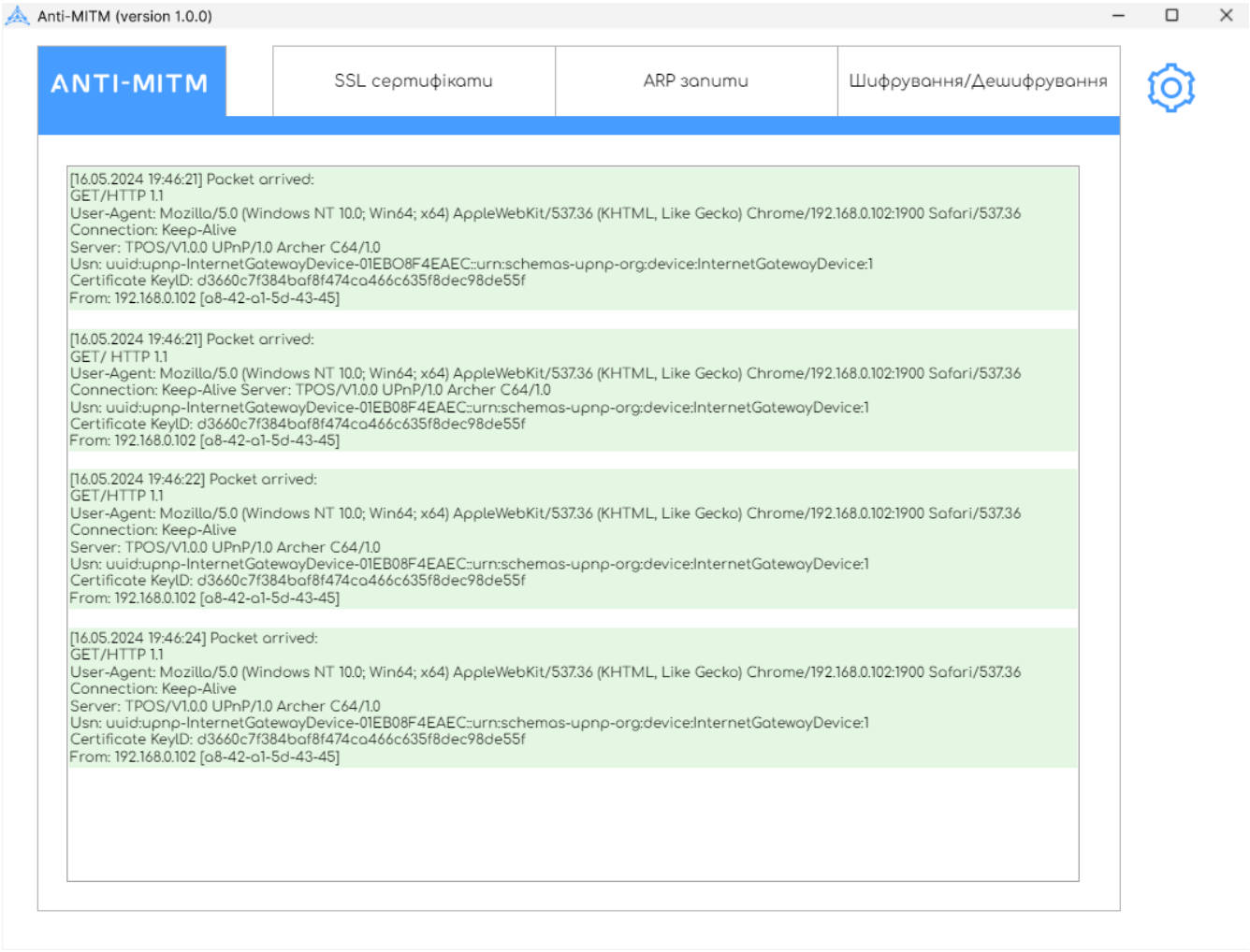

Рисунок 3.6 – Головна з виведеним моніторингом пакетів БМ

У даному випадку в журналі відображаються записи про чотири пакети. До прикладу, останній пакет містить наступну інформацію:

– час: 16:05 2024 19:46:24. Це час, коли було отримано пакет;

– тип запиту: GET/HTTP 11. Це тип запиту, який було надіслано на сервер;

– агент користувача: Mozilla/5.0 (Windows NT 100, Win64; x64) AppleWebKit/53736 (KHTML, like Gecko) Chrome/192.168.0.102:1900 Safari/537.36. Це інформація про тип браузера та пристрою, який надіслав запит;

– підключення: Keep-Alive. Це означає, що після надсилання пакета з'єднання буде підтримуватися;

– сервер: TPOS/V100 UPnP/10 Archer C64/10. Це інформація про тип сервера, який надіслав пакет;

– GatewayDevice-01EB08F4EAECumschemos-upnp-orgdeviceInternet GotawayDevice. Це інформація про шлюз, який використовується для підключення пристрою до Інтернету;

– certificate keyiD: 3660c7f384baf8f474ca466c635fädecide55. Це ідентифікатор сертифіката сервера;

– usn: uuictupnp-Internet. Це унікальний ідентифікатор пристрою, який отримує пакет;

– from: 192.168.0.102 [08-42-01-50-43-45]. Це IP-адреса та MAC-адреса пристрою, який отримує пакет.

Тепер, за допомогою створеного додаткового програмного засобу, було здійснено симуляцію аномального ARP-запиту для того, щоб продемонструвати як програма блокує подібний запит (рис. 3.7).

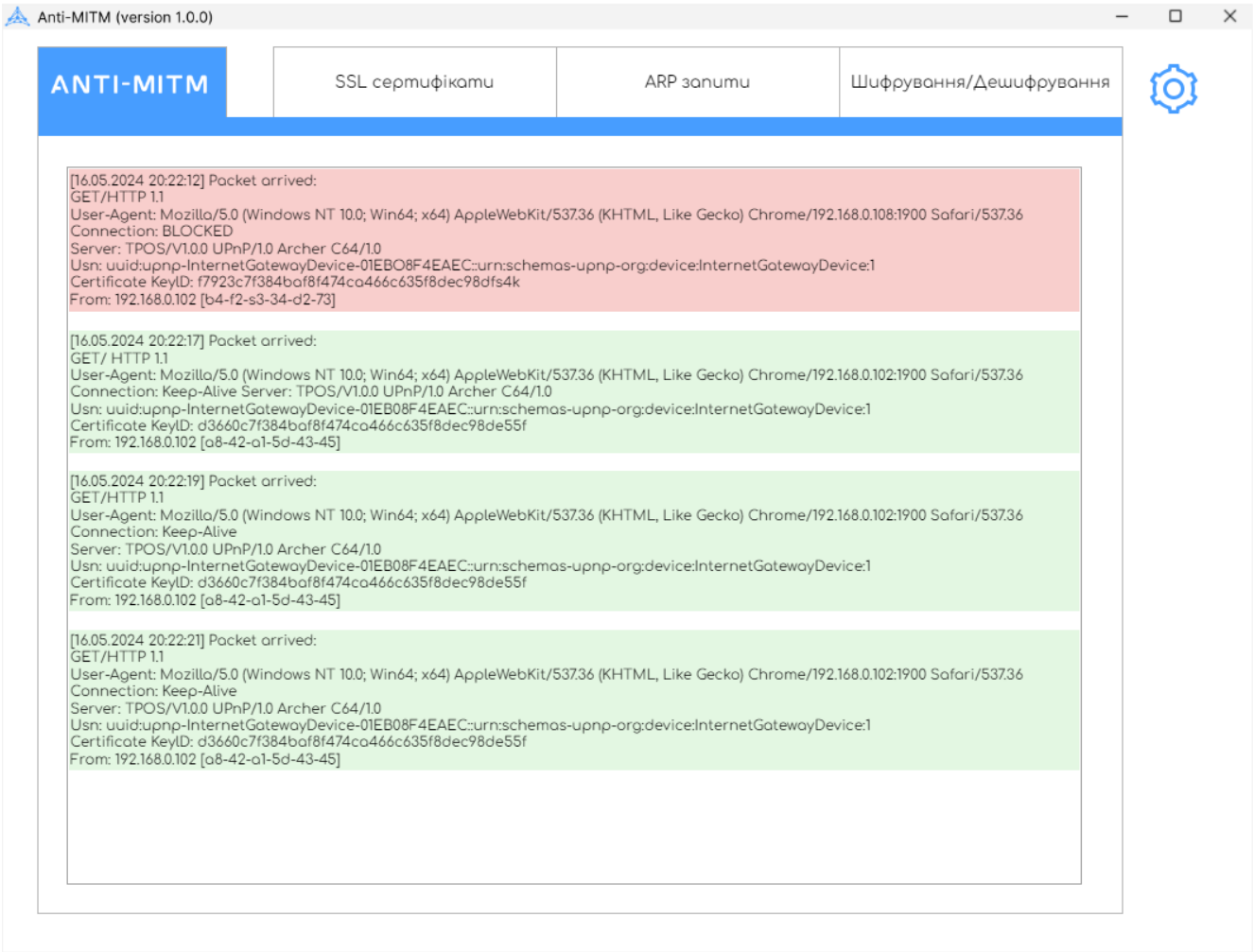

Рисунок 3.7 – Виявлення та блокування небезпечного ARP-запиту

Перейдемо на наступну сторінку, «SSL сертифікати» (рис. 3.8).

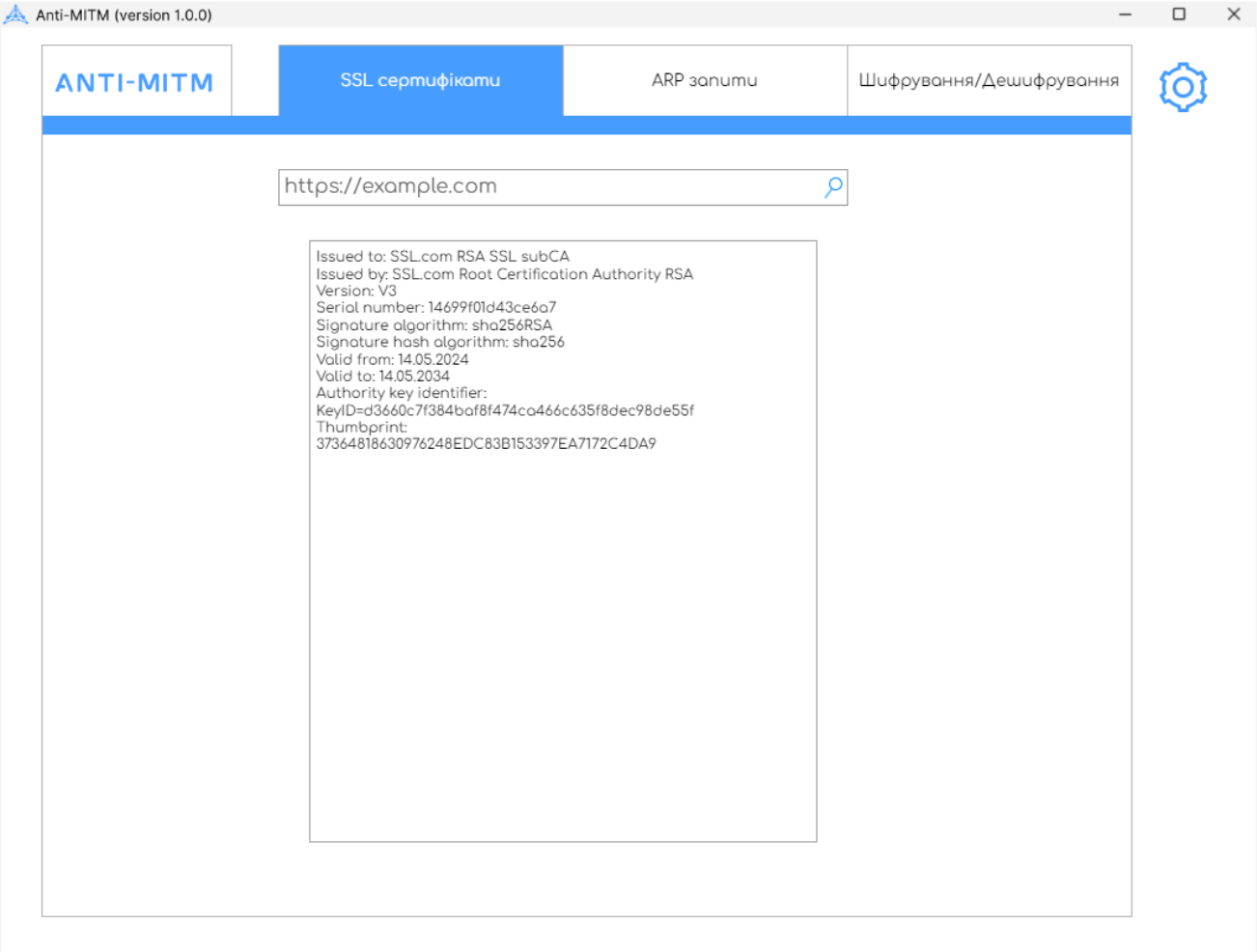

Рисунок 3.8 – Вкладка для перевірки SSL сертифікатів

Вводимо посилання на будь-який сайт і отримаємо інформацію по SSL сертифікату, а саме:

– видано: SSL.com RSA SSL subCA;

– видано: SSL.com Root Certification Authority RSA. Ця інформація показує, хто видав сертифікат SSL. Наприкоад «SSL.com» – це компанія, яка видає сертифікати SSL;

– версія: V3. Ця інформація показує версію протоколу SSL, який використовується сертифікатом. V3 – це найновіша версія протоколу SSL;

– серійний номер: 14699f01d43ce6a7. Цей унікальний номер використовується для ідентифікації сертифіката SSL;

– алгоритм підпису: sha256RSA. Цей алгоритм використовується для шифрування даних, які передаються між веб-сайтом і браузером користувача;

– хеш алгоритму підпису: sha256. Цей хеш-алгоритм використовується для перевірки цілісності сертифіката SSL;

– дійсний з: 14.05.2024. Ця дата показує, коли сертифікат SSL став дійсним;

– дійсний до: 14.05.2034. Ця дата показує, коли сертифікат SSL перестане бути дійсним;

– ідентифікатор ключа авторитету: KeyID=d3660c7f384bof8f474ca466c635f8dec98de55f. Цей ідентифікатор використовується для ідентифікації центру сертифікації, який видав сертифікат SSL;

– відбиток пальця: 37364818630976248EDC83B153397EA7172C4DA9. Цей унікальний відбиток пальця використовується для ідентифікації сертифіката SSL.

Тепер можемо переходити до наступної вкладки, «Шифрування/Дешифрування» (рис. 3.9).

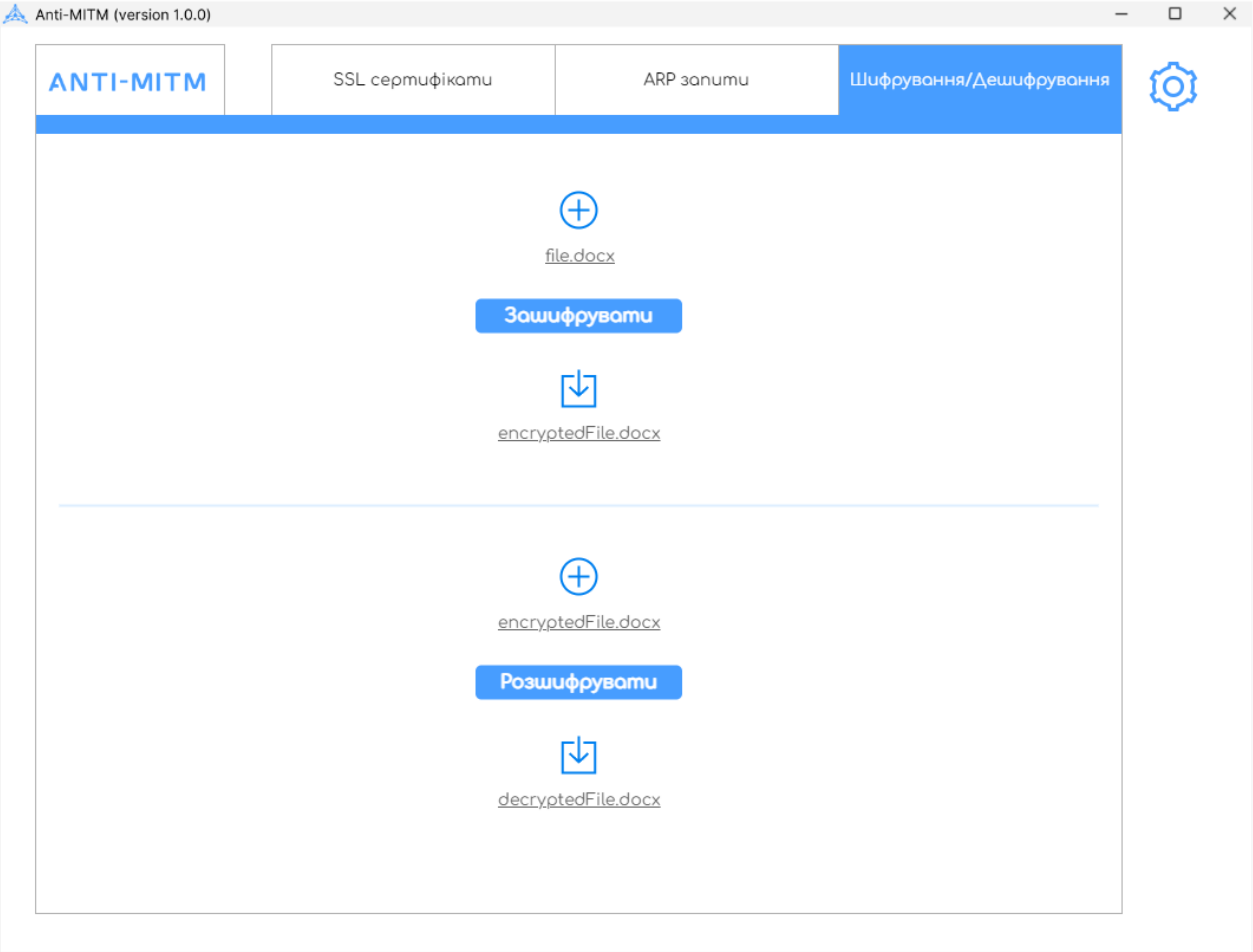

Рисунок 3.9 – Вкладка для шифрування/дешифрування файлів

На цьому рисунку 3.9 показано, як ми спочатку завантажуємо файл «file.docx» (рис. 3.10) для шифрування.

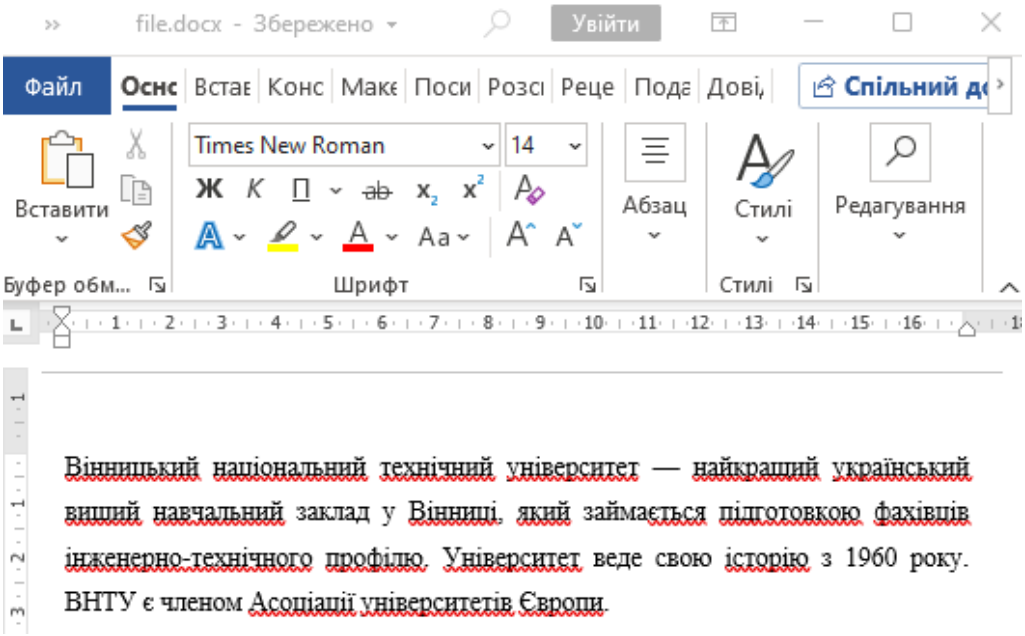

Рисунок 3.10 – Вхідний файл «file.docx»

Натискаємо кнопку «Зашифрувати» і на виході отримаємо зашифрований вхідний файл, який називається «encryptedFile.docx» (рис. 3.11).

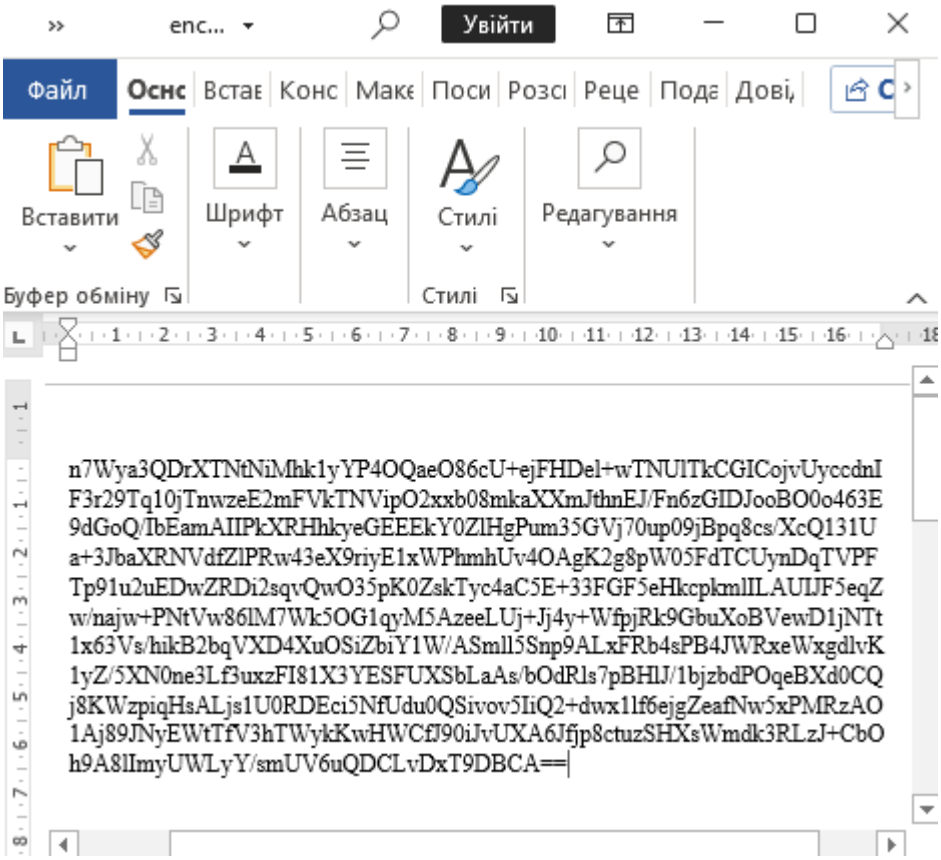

Рисунок 3.11 – Зашифрований вхідний файл «file.docx»

Так само, ми можемо завантажити зашифрований файл «encryptedFile.docx», натиснути кнопку «Розшифрувати» і на виході отримаємо розшифрований файл «decryptedFile.docx».

І наостанок, перейдемо у «Налаштування» (рис. 3.12), де виводиться інформація про версію програми, мінімальна конфігурація, та пошта для зворотного зв’язку.

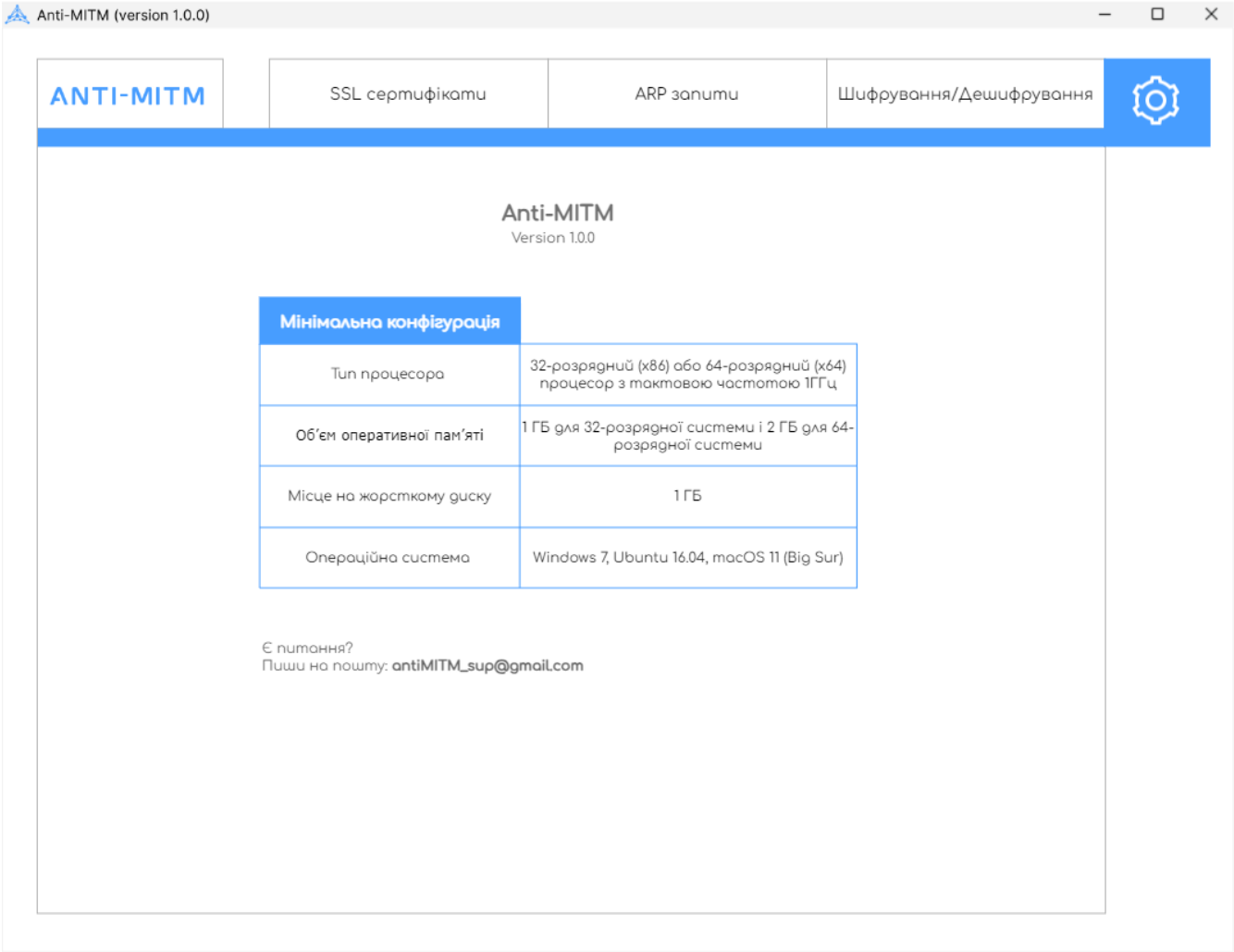

Рисунок 3.12 – Додаткова інформація у вкладці «Налаштування»

На зображенні представлена мінімальна конфігурація для даної програм. Якщо є запитання, можна звертатися на електронну пошту: antiMITM_sup@gmail.com.

Наявність способу для зворотного зв'язку є надзвичайно важливою для покращення програмного забезпечення. Зворотний зв'язок від користувачів допомагає розробникам виявляти та виправляти помилки, розуміти потреби та

очікування користувачів, а також визначати, які функції є найбільш корисними або потребують доопрацювання.

3.5 Розробка інструкції користувача

Інструкція користувача містить вимоги до технічної підтримки програмних продуктів у таблицях 3.1 та 3.2. У них детально описано мінімальні та рекомендовані конфігурації комп’ютера, необхідні для успішної роботи програмних продуктів .

Ознайомившись з цими таблицями ви зможете переконатися, що даний програмний продукт буде належним чином функціонувати на вашому комп’ютері.

Таблиця 3.1 – Мінімальна конфігурація:

Тип процесора	32-розрядний (x86) або 64-розрядний (x64) процесор з тактовою частотою 1ГГц
Об’єм оперативної пам’яті	1 ГБ для 32-розрядної системи і 2 ГБ для 64-розрядної системи
Місце на жорсткому диску	1 ГБ
Операційна система	Windows 7, Ubuntu 16.04, macOS 11 (Big Sur)

Таблиця 3.2 – Рекомендована конфігурація:

Тип процесора	32-розрядний (x86) або 64-розрядний (x64) процесор з тактовою частотою 2ГГц
Об’єм оперативної пам’яті	2 ГБ для 32-розрядної системи і 4 ГБ для 64-розрядної системи
Місце на жорсткому диску	2 ГБ

Операційна система	Windows 10, Ubuntu 20.04, macOS 13 (Ventura)

Після запуску програми, користувач має підключитись до доступної бездротової мережі, для того щоб можна було запустити моніторинг пакетів та у разі небезпеки – блокувати аномальні запити. Також, користувач може робити перевірку валідності SSL сертифікатів і робити Шифрування/Дешифрування файлів, для конфіденційного обміну інформації.

3.6 Висновок

У третьому розділі було детально розглянуто процес створення програми, спрямованої на аналіз, виявлення та захист від атак типу Man-in-the-Middle (MITM) у бездротових мережах. Початковий етап включав вибір засобів для реалізації програмного застосунку, що вимагав аналізу різноманітних технологій та інструментів з метою вибору тих, що найбільш ефективно вирішували б поставлені завдання.

Програмна реалізація застосунку була проведена відповідно до обраного підходу та врахування вимог безпеки. Кожен етап розробки був уважно спланований та реалізований з урахуванням потреб проекту в аналізі мережевого трафіку, виявленні аномалій та забезпеченні захисту від MITM-атак.

Тестування роботи програми було важливим етапом, що дозволило перевірити її ефективність у виявленні та захисті від MITM-атак. Під час цього етапу виконувалися тестові сценарії, спрямовані на симуляцію атаки, а також перевірялася коректність реагування програми на них.

Завершальним етапом була розробка інструкції користувача, яка мала на меті забезпечити зрозуміле та ефективне використання програми кінцевими користувачами.

Таким чином, цей розділ забезпечив повний цикл створення програмного продукту для захисту мереж від MITM-атак.

ВИСНОВКИ

У ході виконання дипломної роботи було виконано усі поставлені задачі. Було розглянуто основні аспекти захисту бездротових мереж від атак типу Man-in-the-Middle (MITM), зокрема підроблених SSL-сертифікатів, аномальних ARP-запитів-відповідей та змін у мережевому трафіку. Проаналізовано теоретичні основи MITM-атак, методи їх реалізації та наслідки для бездротових мереж. Проведений аналіз дозволив виявити основні загрози та вразливості, які використовуються зловмисниками для проведення таких атак.

В межах цієї роботи було розроблено модель програми, визначено алгоритми її роботи та спроектовано користувацький інтерфейс. Основна увага приділялася створенню ефективних механізмів для виявлення та запобігання MITM-атакам, що дозволить покращити захищеність бездротових мереж від зловмисних впливів.

Ми обрали засоби для реалізації програми, проведено її програмну реалізацію, а також тестування та налаштування для забезпечення коректної роботи. Важливим аспектом є створення інструкцій для користувачів, що допоможе спростити процес експлуатації розробленого програмного забезпечення та забезпечить його ефективне використання.

У підсумку, результатом проведеного дослідження стала створена програма, яка здатна ефективно виявляти та запобігати MITM-атакам у бездротових мережах, а саме: аномальні ARP запити, не валідні SSL сертифікати та шифрування/дешифрування файлів з дуже важливою та конфіденційною інформацією. Практичне застосування даного програмного забезпечення може бути корисним для організацій, що використовують бездротові мережі у своїй діяльності.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Атаки MITM: огляд і систематика підходів до захисту. URL: https://ieeexplore.ieee.org/document/9817541 (дата звернення 7.05.2024).

2. Ресурси з кібербезпеки та інформаційної безпеки. URL: https://knowledgebase.paloaltonetworks.com/KCSArticleDetail?id=kA10g000000Cm8wCAC (дата звернення 7.05.2024).

3. P.W. Singer and Allan Friedman. Cybersecurity and Cyberwar: What Everyone Needs to Know / P.W. Singer and Allan Friedman. – USA: Oxford University Press, 2014. – 64 p (дата звернення 11.05.2024).

4. Jon Erickson. Hacking: The Art of Exploitation / Jon Erickson – USA: No Starch Press, 2008. – 102 p (дата звернення 13.05.2024).

5. Johnny Cache, Joshua Wright, and Vincent Liu. Hacking Exposed Wireless: Wireless Security Secrets & Solutions / Johnny Cache, Joshua Wright, and Vincent Liu – USA: McGraw Hill Professional, 2010. – 314 p (дата звернення 14.05.2024).

6. Charlie Kaufman, Radia Perlman, and Mike Speciner. Network Security: Private Communication in a Public World / Charlie Kaufman, Radia Perlman, and Mike Speciner – USA: PTR Prentice Hall, 1995. – 250 p (дата звернення 16.05.2024).

7. Що означає спуфінг? Виявлення та запобігання. URL: https://hideez.com/uk/blogs/news/spoofing-prevention (дата звернення 17.05.2024).

8. Що таке DNS підміна (DNS спуфінг). URL: https://ukeywaf.com/baza/shho-take-dns-pidmina-dns-spufing/ (дата звернення 17.05.2024).

9. Що таке ARP-спуфінг?. URL: https://ukeywaf.com/baza/shho-take-arp-spufing/ (дата звернення 17.05.2024).

10. Викрадення SSL. URL: https://www.invicti.com/learn/mitm-ssl-hijacking/ (дата звернення 17.05.2024).

11. Що таке SSL Stripping атака?. URL: https://www.checkpoint.com/cyber-hub/cyber-security/what-is-cyber-attack/what-is-an-ssl-stripping-attack/ (дата звернення 17.05.2024).

12. Що таке BEAST Attack. URL: https://www.acunetix.com/blog/web-security-zone/what-is-beast-attack/ (дата звернення 17.05.2024).

13. Що таке Спуфінг: визначення та приклади. URL: https://gridinsoft.ua/spoofing (дата звернення 18.05.2024).

14. Сніфери (Sniffers). URL: https://ukeywaf.com/baza/snifery-sniffers/ (дата звернення 18.05.2024).

15. Визначення атаки посередника. URL: https://gridinsoft.ua/mitm (дата звернення 18.05.2024).

16. Атаки типу Man-In-The-Middle: що треба знати кожному. URL: https://www.imena.ua/blog/man-in-the-middle/ (дата звернення 18.05.2024).

17. Microsoft: Фішинг обійшов MFA в атаках на 10 000 організацій. URL: https://www.bleepingcomputer.com/news/security/microsoft-phishing-bypassed-mfa-in-attacks-against-10-000-orgs/ (дата звернення 18.05.2024).

18. Пост про ситуацію, від адміністратора Reddit. URL: https://www.reddit.com/r/reddit/comments/10y427y/we_had_a_security_incident_heres_what_we_know/ (дата звернення 18.05.2024).

19. Сотні документів НАТО, надісланих до Португалії, були виявлені для продажу в темній мережі. URL: https://www.portugalresident.com/hundreds-of-nato-documents-sent-to-portugal-discovered-for-sale-on-dark-web/ (дата звернення 18.05.2024).

20. 10 способів запобігти атакам людини посередині (MITM). URL: https://www.strongdm.com/blog/man-in-the-middle-attack-prevention (дата звернення 19.05.2024).

21. Як захистити свій обліковий запис за допомогою двоетапної автентифікації. URL: https://support.google.com/accounts/answer/185839 (дата звернення 19.05.2024).

22. Microsoft Authenticator що це?. URL: https://simpla.com.ua/blog/microsoft-authenticator-shcho-ce (дата звернення 19.05.2024).

23. Що таке шифрування BitLocker і як воно працює?. URL: https://www.sysdevlabs.com/uk/articles/storage-technologies/what-is-bitlocker-and-how-does-it-work/ (дата звернення 19.05.2024).

24. Як перевірити чи має веб-сайт SSL-сертифікат?. URL: https://rx-name.ua/blog/yak-pereviryty-chy-maye-veb-sajt-ssl-sertyfikat (дата звернення 19.05.2024).

25. Фішинг. URL: https://www.eset.com/ua/support/information/entsiklopediya-ugroz/fishing/ (дата звернення 19.05.2024).

26. Діаграма діяльності в UML: символ, компоненти та приклад. URL: https://www.guru99.com/uk/uml-activity-diagram.html (дата звернення 19.05.2024).

27. Юзабіліті сайту. URL: https://redstone.media/usability-saytu (дата звернення 19.05.2024).

28. Joseph Albahari and Ben Albahari. C# 9.0 in a Nutshell: The Definitive Reference / Joseph Albahari and Ben Albahari – USA: O'Reilly, 2021. – 873 p (дата звернення 20.05.2024).

29. Найкращий редактор коду: Visual Studio Code. URL: https://dou.ua/forums/topic/47918/ (дата звернення 21.05.2024).

30. Ryan Vice, Mohammad Azam. MVVM Survival Guide for Enterprise Architectures in Silverlight and WPF / Ryan Vice, Mohammad Azam – USA: Packt Publishing, 2012. – 158 p (дата звернення 22.05.2024).

ДОДАТКИ

ДОДАТОК А – ТЕХНІЧНЕ ЗАВДАННЯ

Міністерство освіти і науки України

Вінницький національний технічний університет

Факультет менеджменту та інформаційної безпеки

ЗАТВЕРДЖУЮ

Голова секції "Управління інформаційною безпекою" кафедри МБІС
д.т.н., професор

_____ Юрій ЯРЕМЧУК

«22» березня 2024 р

ТЕХНІЧНЕ ЗАВДАННЯ

до бакалаврської дипломної роботи на тему:

«Розробка програми для аналізу, виявлення та захисту від атаки типу Man-in-the-Middle (MITM) у бездротових мережах, включаючи підроблені SSL-сертифікати, аномальні ARP-запити-відповіді та зміни в мережевому трафіку»

08-72.БДР.015.00.098 ТЗ

Керівник бакалаврської дипломної роботи:
к.т.н., проф., каф., МБІС:

_____ Яремчук Ю. Є.

«22» березня 2024 р

Вінниця – 2024 року

1. Найменування та область застосування

Розробка програми для аналізу, виявлення та захисту від атаки типу Man-in-the-Middle (MITM) у бездротових мережах, включаючи підроблені SSL-сертифікати, аномальні ARP-запити-відповіді та зміни в мережевому трафіку Область застосування: захист користувачів від MITM атак у бездротових мережах.

2.Підстава для розробки

Розробка виконується на основі наказу ректора ВНТУ № 80 від 11.03.2024 р.

3. Мета та призначення розробки

3.1 Мета розробки: створення програмного модуля для аналізу, виявлення та захисту від атак типу Man-in-the-Middle (MITM) у бездротових мережах шляхом виявлення підроблених SSL-сертифікатів, аномальних ARP-запитів-відповідей та змін у мережевому трафіку.

3.2 Призначення: розроблений програмний модуль попереджає користувача про зловмисні ARP запити і блокує їх, також виявляє підроблені SSL-сертифікати.

4. Джерела розробки

4.1 Юзабіліті сайту. URL: https://redstone.media/usability-saytu.

4.2 Joseph Albahari and Ben Albahari. C# 9.0 in a Nutshell: The Definitive Reference / Joseph Albahari and Ben Albahari – USA: O'Reilly, 2021. – 873 p.

4.3 Найкращий редактор коду: Visual Studio Code. URL: https://dou.ua/forums/topic/47918/.

4.4 Ryan Vice, Mohammad Azam. MVVM Survival Guide for Enterprise Architectures in Silverlight and WPF / Ryan Vice, Mohammad Azam – USA: Packt Publishing, 2012. – 158 p.

5. Вимоги до програми

5.1 Вимоги до функціональних характеристик:

5.1.1 Інтерфейс програмного модуля має бути інтуїтивно зрозумілим і зручним для користувачів;

5.1.2 Впровадження модуля не повинно потребувати використання спеціалізованого ліцензійного програмного забезпечення;

5.1.3 Програмний модуль має забезпечувати виявлення атак типу Man-in-the-Middle (MITM) у бездротових мережах і своєчасно попереджати користувачів.

5.2 Вимоги до надійності:

5.2.1 Програмний модуль повинен функціонувати без збоїв, а у випадку критичних ситуацій виводити відповідні попередження;

5.2.2 Має бути доступна форма зворотного зв'язку для надання підтримки користувачам програми.

5.2.3 Програмний модуль повинен виконувати свої функції.

5.3 Вимоги до складу і параметрів технічних засобів:

– процесор – 32-розрядний (x86) або 64-розрядний (x64), з тактовою частотою 1ГГц;

– оперативна пам’ять – не менше 1 ГБ для 32-розрядної системи і 2 ГБ для 64-розрядної системи;

– середовище функціонування – операційна система сімейство Windows;

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

6. Вимоги до програмної документації

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

7. Вимоги до технічного захисту інформації

7.1 Необхідно забезпечити захист розроблюваного програмного засобу від несанкціонованого використання.

7.2 Можливість отримання доступу користувачів до інформаційних ресурсів таких як чорні та білі списки.

8. Техніко-економічні показники

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

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

9. Стадії та етапи розробки

№ з/п	Назва етапів бакалаврської дипломної роботи	Початок	Закінчення
1	Визначення напрямку бакалаврської роботи, формулювання теми	22.03.2024	25.03.2024
2	Аналіз предметної області обраної теми	26.03.2024	29.03.2024
3	Розробка алгоритму роботи	16.04.2024	28.04.2024
4	Написання бакалаврської роботи на основі розробленої теми	12.05.2024	22.05.2024
5	Передзахист бакалаврської дипломної роботи		
6	Виправлення, уточнення, корегування бакалаврської дипломної роботи		
7	Захист бакалаврської дипломної роботи		

10. Порядок контролю та прийому

10.1 До приймання бакалаврської дипломної роботи надається:

- ПЗ до бакалаврської дипломної роботи;
- демонстрація результату бакалаврської дипломної роботи;
- презентація;
- відзив керівника роботи;
- відзив рецензента.

Технічне завдання до виконання прийняла _____ Матяш-Рибачук О. С.

ДОДАТОК Б – ЛІСТИНГ ПРОГРАМИ

```
using PacketDotNet;
using SharpPcap;
using SimpleWifi;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net.NetworkInformation;
using System.Threading.Tasks;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;

namespace WpfApp
{
   public partial class ConnectionPage : Page
   {
      public IEnumerable<AccessPoint>? AccessPoints { get; set; }
      public IEnumerable<string>? AccessPointsString { get; set; }
      public IEnumerable<ILiveDevice>? DevicesCollection { get; set; }

      private Wifi? _wifi;

      public ConnectionPage()
      {
         InitializeComponent();
         InitData();
      }

      private void InitData()
      {
         _wifi = new Wifi();
         _wifi.ConnectionStatusChanged += OnConnectionStatusChanged;

         if (_wifi.NoWifiAvailable)
         {
            WiFiStatusLabel.Content = "No WiFi card found";
         }

         AccessPoints = GetAccessPoints();
         AccessPointsString = GetAccessPointsString();

         AccessPointsListView.ItemsSource = AccessPointsString;
```

```
        GetConncetionStatus();
        GetAccessPointProfile();
    }

    private IEnumerable<AccessPoint> GetAccessPoints()
    {
        IEnumerable<AccessPoint> accessPoints = _wifi.GetAccessPoints().OrderByDescending(ap =>
ap.SignalStrength);

        return accessPoints;
    }

    private IEnumerable<string> GetAccessPointsString()
    {
        IEnumerable<AccessPoint> accessPoints = _wifi.GetAccessPoints().OrderByDescending(ap =>
ap.SignalStrength);

        int i = 1;

        List<string> accessPointsString = new List<string>();

        foreach (var ap in accessPoints)
        {
            accessPointsString.Add($"{i++}. {ap.Name ?? "NA"} {ap.SignalStrength}%");
        }

        return accessPointsString;
    }

    private void GetConncetionStatus()
    {
        if (_wifi.ConnectionStatus == WifiStatus.Connected)
        {
            var connectedAccessPoint = GetAccessPoints().FirstOrDefault(ap => ap.IsConnected);

            WifiStatusLabel.Content = $"Connected to {connectedAccessPoint?.Name}";
            ConnectedAccessPointInfo.Text = $"{connectedAccessPoint}\r\n";
        }
        else
        {
            WifiStatusLabel.Content = "Not connected";
            ConnectedAccessPointInfo.Text = "";
        }
    }

    private void GetAccessPointProfile()
    {
        var connectedAccessPoint = GetAccessPoints().FirstOrDefault(ap => ap.IsConnected);
```

```
        if (ConnectionsList.SelectedItem > accessPoints.ToArray().Length ||
accessPoints.ToArray().Length == 0)
        {
            return;
        }

        AccessPoint selectedAP = accessPoints.ToList()[ConnectionsList.SelectedItem];

        if (selectedAP.HasProfile)
        {
            ConnectionProfileTextBox.Text = selectedAP.GetProfileXML();
        }
    }

    private void OnConnectionStatusChanged(object sender, WifiStatusEventArgs e)
    {
        GetConncetionStatus();
    }

    private void Connect()
    {
        var accessPoints = GetAccessPoints();

        if (ConnectionsList.SelectedItem > accessPoints.ToArray().Length ||
accessPoints.ToArray().Length == 0)
        {
            return;
        }

        AccessPoint selectedAP = accessPoints.ToList()[ConnectionsList.SelectedItem];
        AuthRequest authRequest = new AuthRequest(selectedAP);

        if (authRequest.IsPasswordRequired)
        {
            if (authRequest.IsUsernameRequired)
            {
                authRequest.Username = ConnectionNameTextBox.Text;
            }

            if (selectedAP.IsValidPassword(ConnectionPasswordBox.Text))
            {
                authRequest.Password = PasswordPrompt(selectedAP);
            }
        }

        selectedAP.ConnectAsync(authRequest);
    }
```

```
    }
}

using SharpPcap;
using SharpPcap.LibPcap;
using PacketDotNet;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.CompilerServices;

namespace ManInTheMiddleAnalyzer
{
    public class ArpMonitor
    {
        private Dictionary<string, string> _arpTable = new Dictionary<string, string>();
        private ILiveDevice? _device;

        public void StartArpMonitoring()
        {
            var devices = CaptureDeviceList.Instance;

            if(devices.Count < 1)
            {
                return;
            }

            var _device = devices[0];

            device.OnPacketArrival += Device_OnPacketArrival;

            device.Open(DeviceModes.Promiscuous);
            device.Capture();
            device.Close();
        }

        private void Device_OnPacketArrival(object sender, PacketCapture e)
        {
            var packet = e.GetPacket();
            var parsedPacket = Packet.ParsePacket(packet.LinkLayerType, packet.Data);
            var arpPacket = parsedPacket.Extract<ArpPacket>();

            if(arpPacket != null)
            {
                var senderIp = arpPacket.SenderProtocolAddress.ToString();
                var senderMac = arpPacket.SenderHardwareAddress.ToString();

                if(_arpTable.ContainsKey(senderIp) && _arpTable[senderIp] != senderMac)
```

```
            {
                _device.BlockRequests(senderMac);
            }

            _arpTable[senderIp] = senderMac;
        }
    }
  }
}

using Microsoft.Extensions.Logging;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Net.Security;
using System.Security.Cryptography;
using System.Security.Cryptography.X509Certificates;
using System.Text;
using System.Threading.Tasks;

namespace ManInTheMiddleAnalyzer
{
  class CertificateValidator
  {
    private ILogger<CertificateValidator> _logger;

    public CertificateValidator(ILogger<CertificateValidator> logger)
    {
      _logger = logger;
    }

    public async Task ConfigureRequest()
    {
      string url = SiteLinkTextBox.Text;

      HttpClientHandler handler = new HttpClientHandler();
      handler.ServerCertificateCustomValidationCallback = ServerCertificateValidation;

      using (HttpClient client = new HttpClient(handler))
      {
        try
        {
          HttpResponseMessage response = await client.GetAsync(url);
          response.EnsureSuccessStatusCode();

          string content = await response.Content.ReadAsStringAsync();
        }
```

```
            catch (HttpRequestException e)
            {
                _logger.Error(e.Message);
            }
        }
    }

    private static bool ServerCertificateValidation(HttpRequestMessage requestMessage,
X509Certificate2 certificate, X509Chain chain, SslPolicyErrors sslErrors)
    {
        var validCertificate = new X509Certificate2(_validCertificatePath);
        var validChain = new X509Chain(validCertificate).Build();

        if (validChain.ChainElements.Count != requestCertificateChain.ChainElements.Count)
        {
            return false;
        }

        for (int i = 0; i < validChain.ChainElements.Count; i++)
        {
            if (validChain.ChainElements[i].Certificate.Thumbprint !=
requestCertificateChain.ChainElements[i].Certificate.Thumbprint)
            {
                sslPolicyErrors = SslPolicyErrors.RemoteCertificateChainErrors;
            }
        }

        if (sslPolicyErrors == SslPolicyErrors.None)
        {
            return true;
        }

        return false;
    }
  }
}

using System;
using System.Windows;
using System.Windows.Navigation;

namespace WpfApp
{
  public partial class MainWindow : Window
  {
    public MainWindow()
    {
      InitializeComponent();
```

```
            MainWindowFrame.NavigationUIVisibility = NavigationUIVisibility.Hidden;
        }

        private void ArpRequestsPageButton_Click(object sender, RoutedEventArgs e)
        {
            if (IsFrameContentUsed(typeof(ArpRequestPage)))
            {
                MainWindowFrame.RemoveBackEntry();
            }

            var arpRequestsPage = new ArpRequestsPage();
            MainWindowFrame.Content = arpRequestsPage;
        }

        private void CertificatePageButton_Click(object sender, RoutedEventArgs e)
        {
            if (IsFrameContentUsed(typeof(CertificatePage)))
            {
                MainWindowFrame.RemoveBackEntry();
            }

            var certificatePage = new CertificatePage();
            MainWindowFrame.Content = certificatePage;
        }

        private void EncryptionPageButton_Click(object sender, RoutedEventArgs e)
        {
            if (IsFrameContentUsed(typeof(EncryptionPage)))
            {
                MainWindowFrame.RemoveBackEntry();
            }

            var encryptionPage = new EncryptionPage();
            MainWindowFrame.Content = encryptionPage;
        }

        private bool IsFrameContentExists()
        {
            return MainWindowFrame.Content != null;
        }

        private bool IsFrameContentUsed(Type type)
        {
            return MainWindowFrame.Content.GetType() == type;
        }
    }
}
```

ДОДАТОК В – ГРАФІЧНА ЧАСТИНА

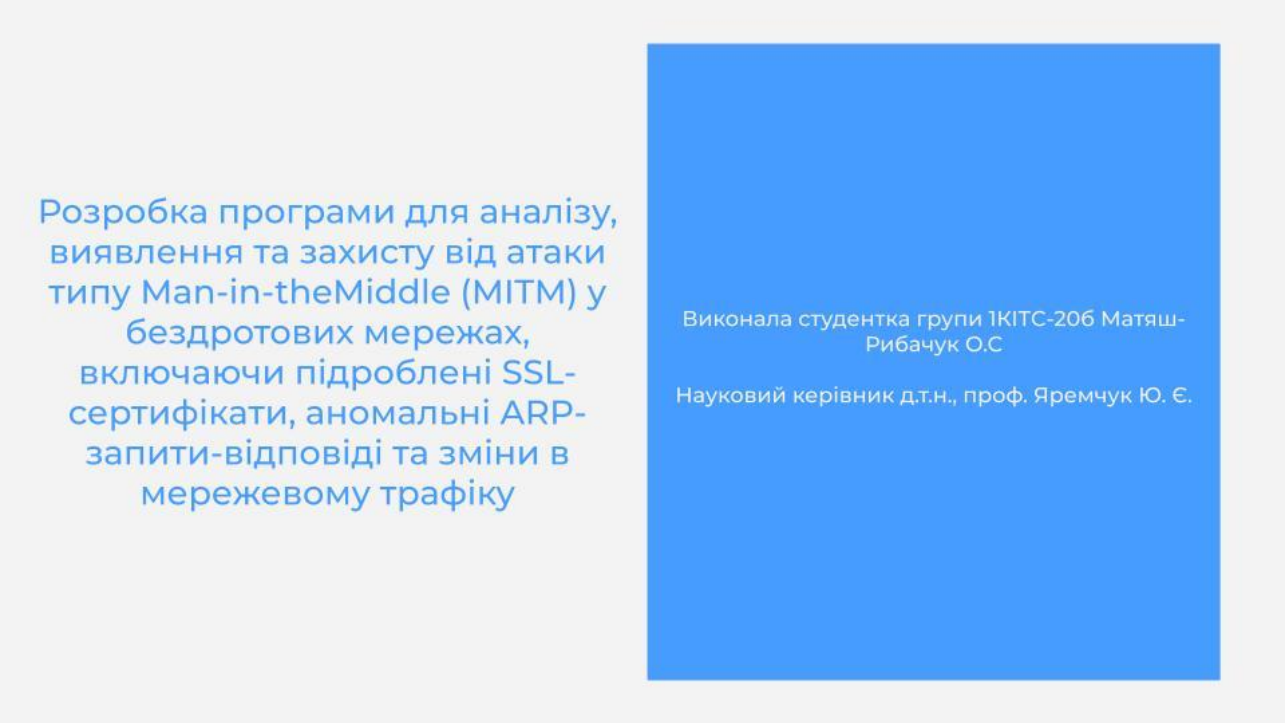

Рисунок В.1 – Назва роботи

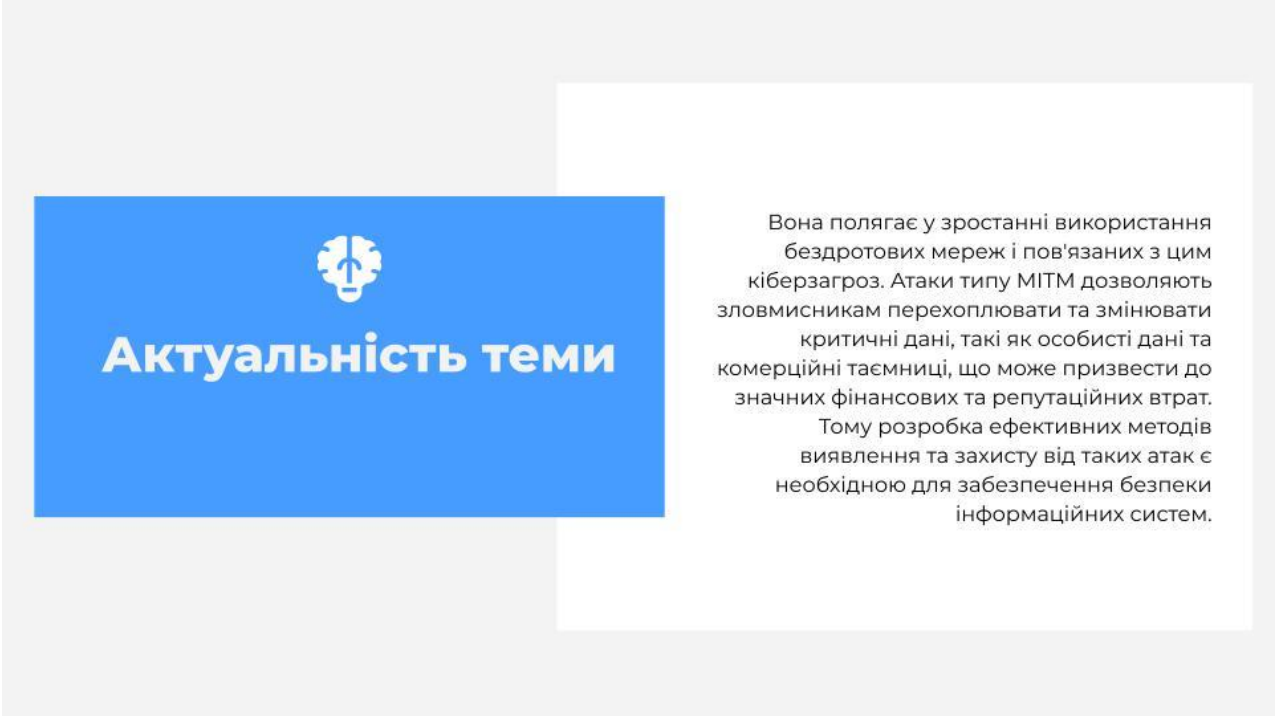

Рисунок В.2 – Актуальність теми

Рисунок В.3 – Мета та об'єкт дослідження

Рисунок В.4 – Основні задачі

Рисунок В.5 – Головний алгоритм роботи

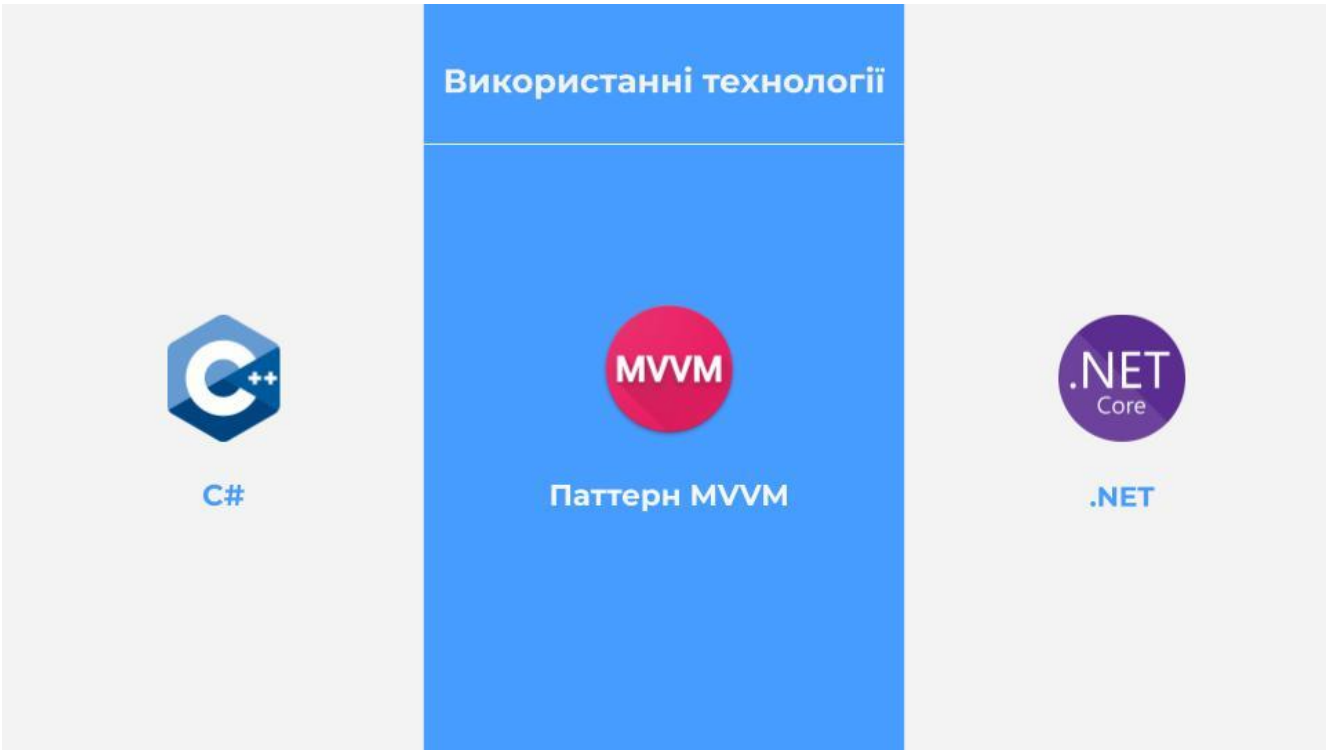

Рисунок В.6 – Використанні технології

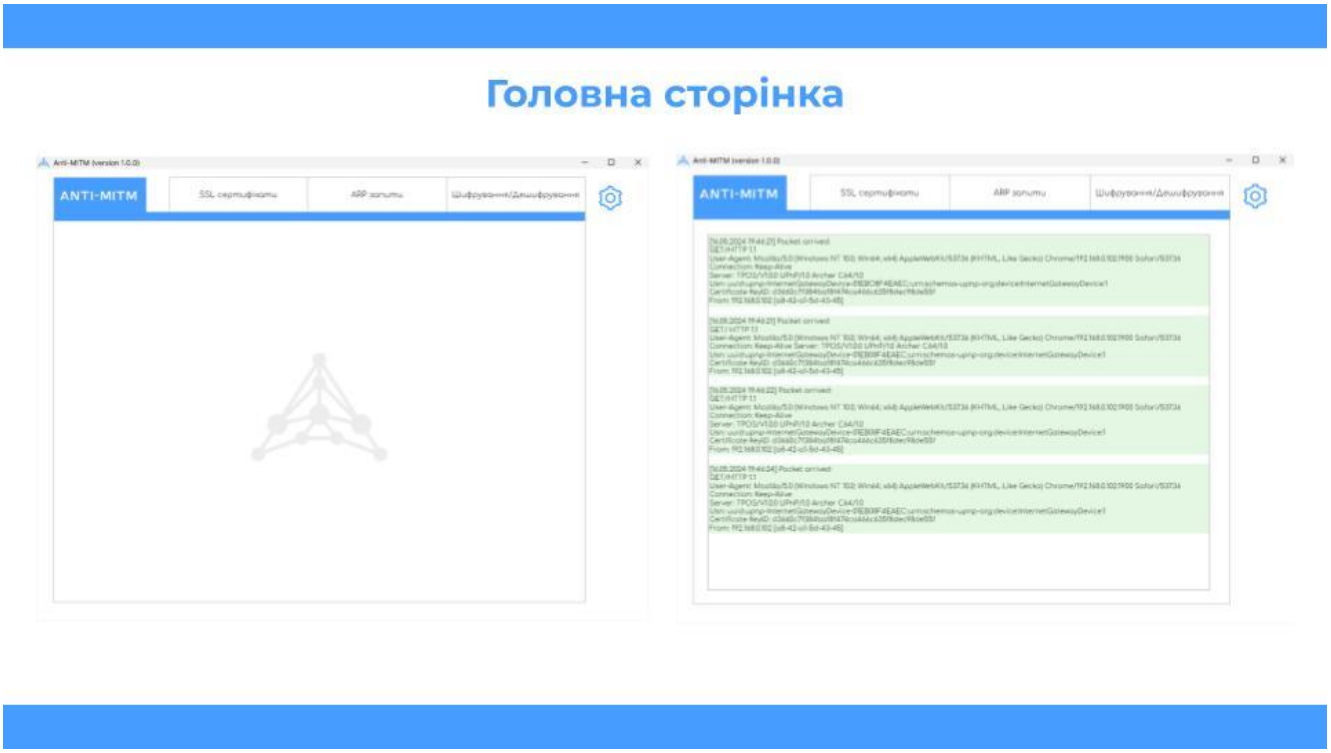

Рисунок В.7 – Головна сторінка програми

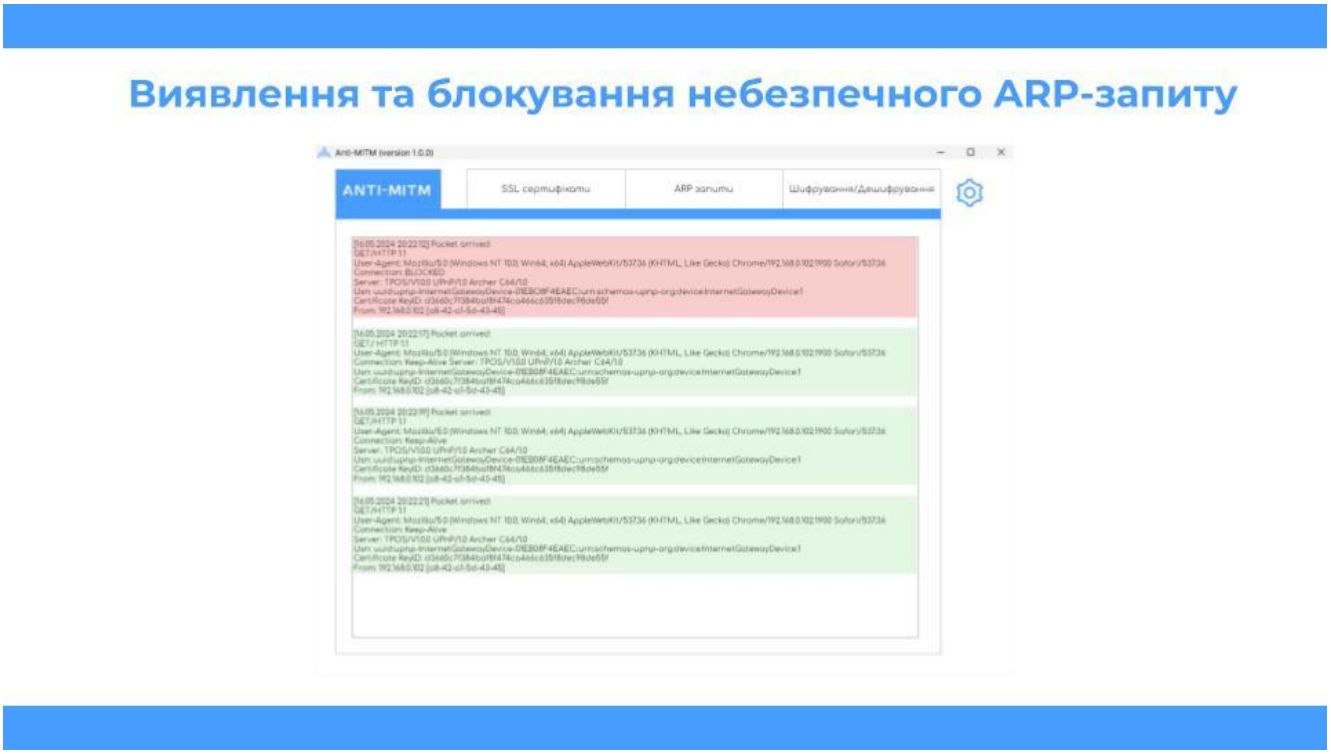

Рисунок В.8 – Виявлення та блокування небезпечного ARP-запиту

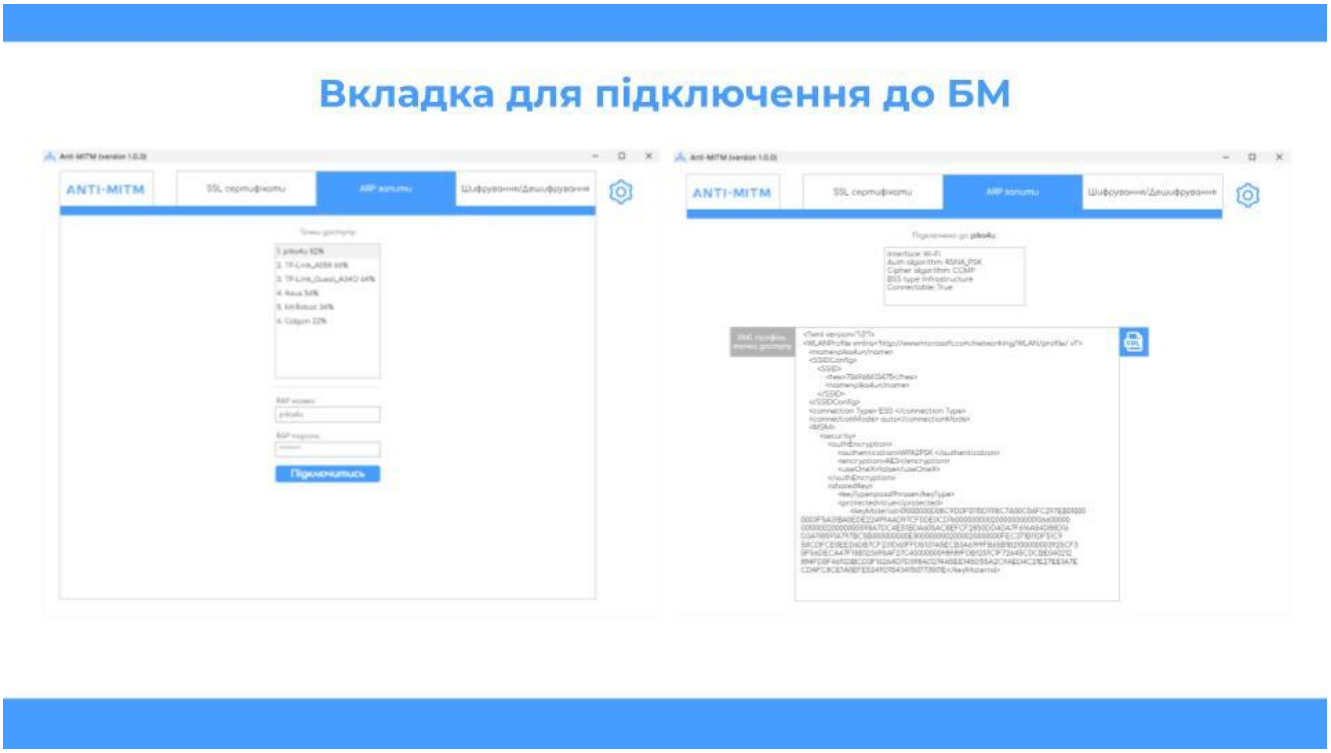

Рисунок В.9 – Вкладка для підключення до БМ

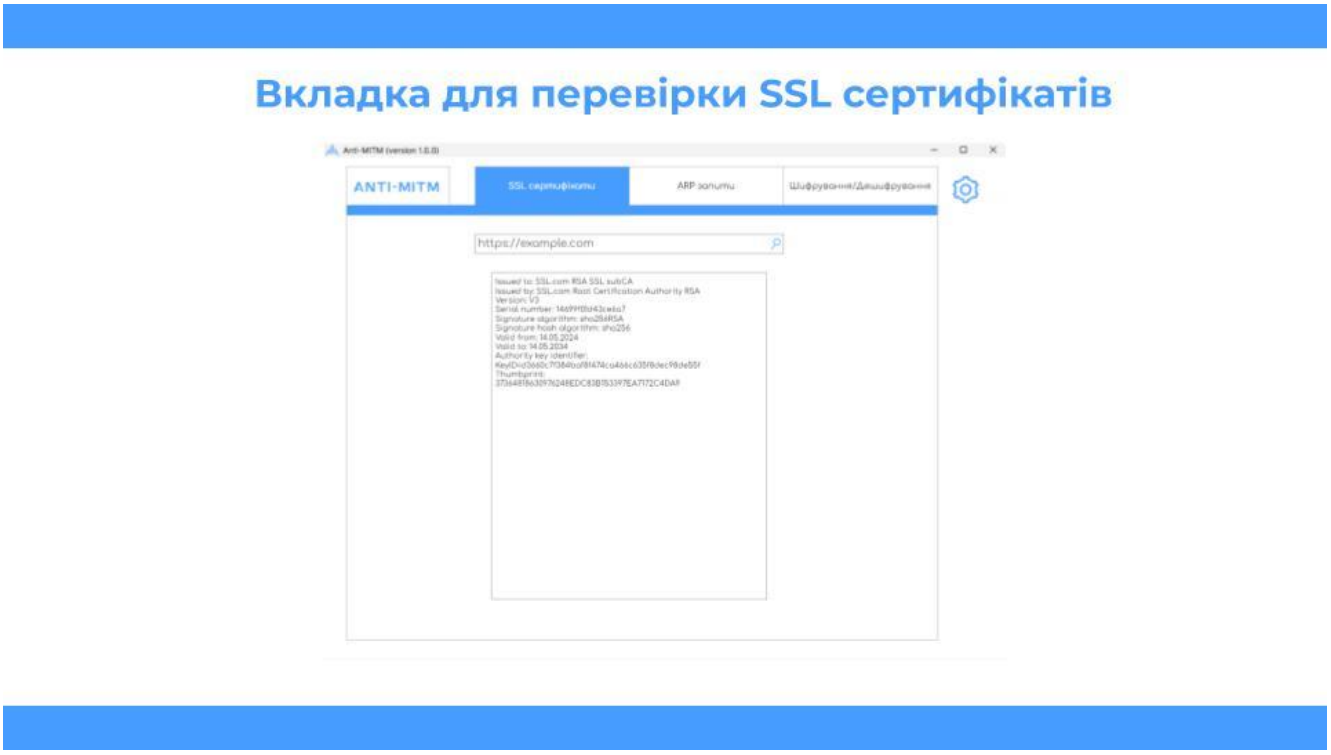

Рисунок В.10 – Вкладка для первірки SSL сертифікатів

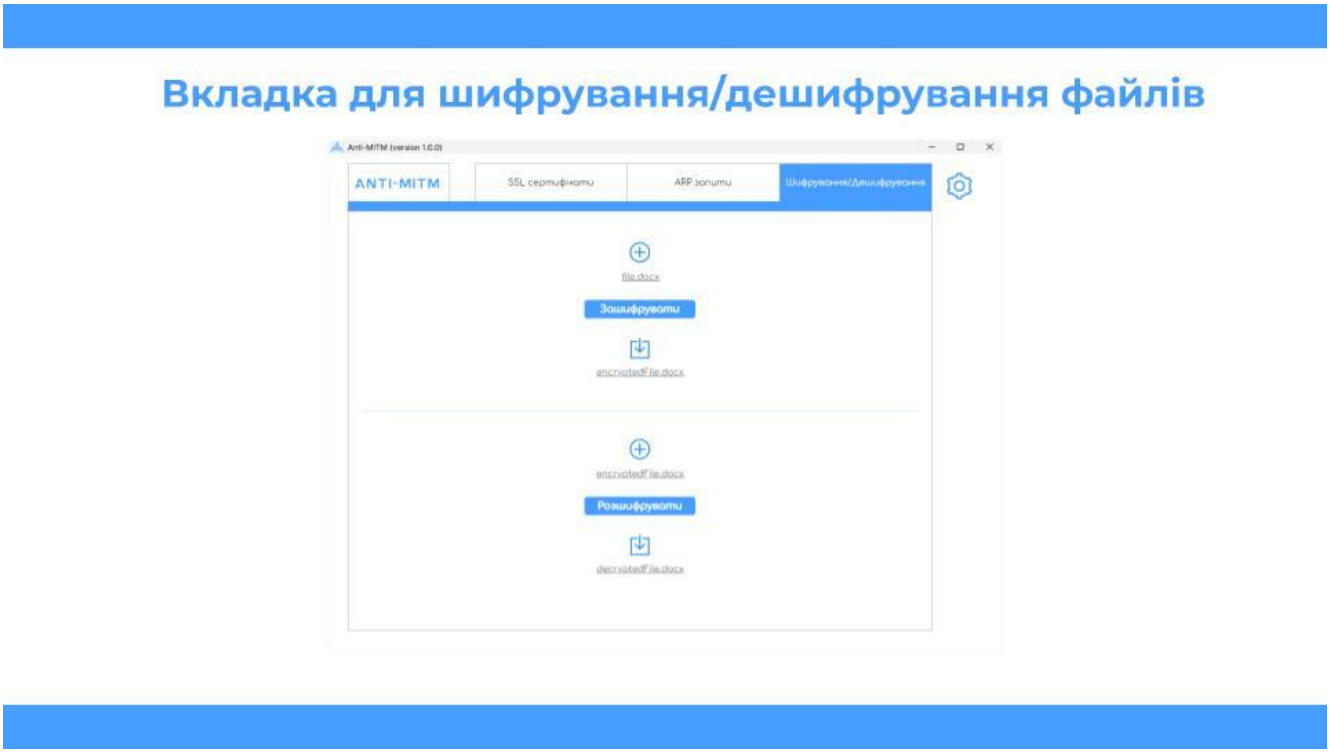

Рисунок В.11 – Вкладка для шифруванна/дешифрування файлів

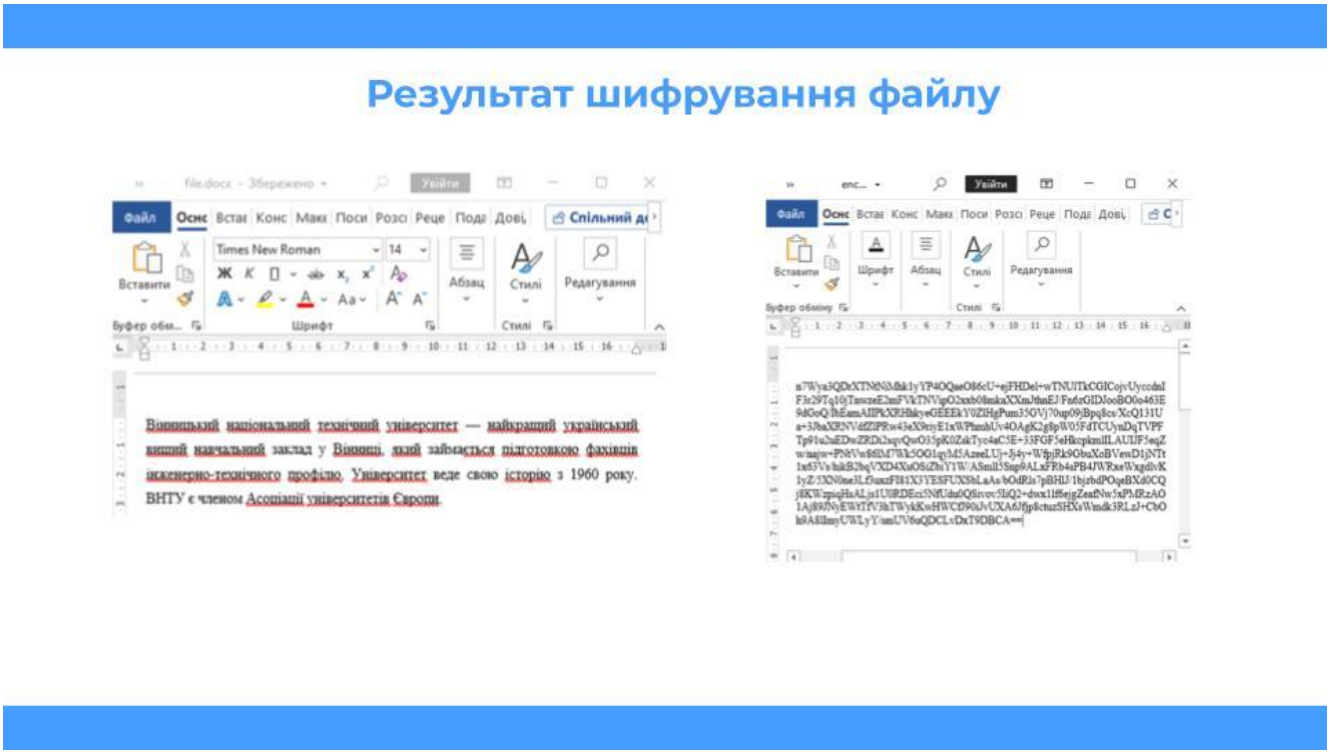

Рисунок В.12 – Приклад зашифрованого файлу

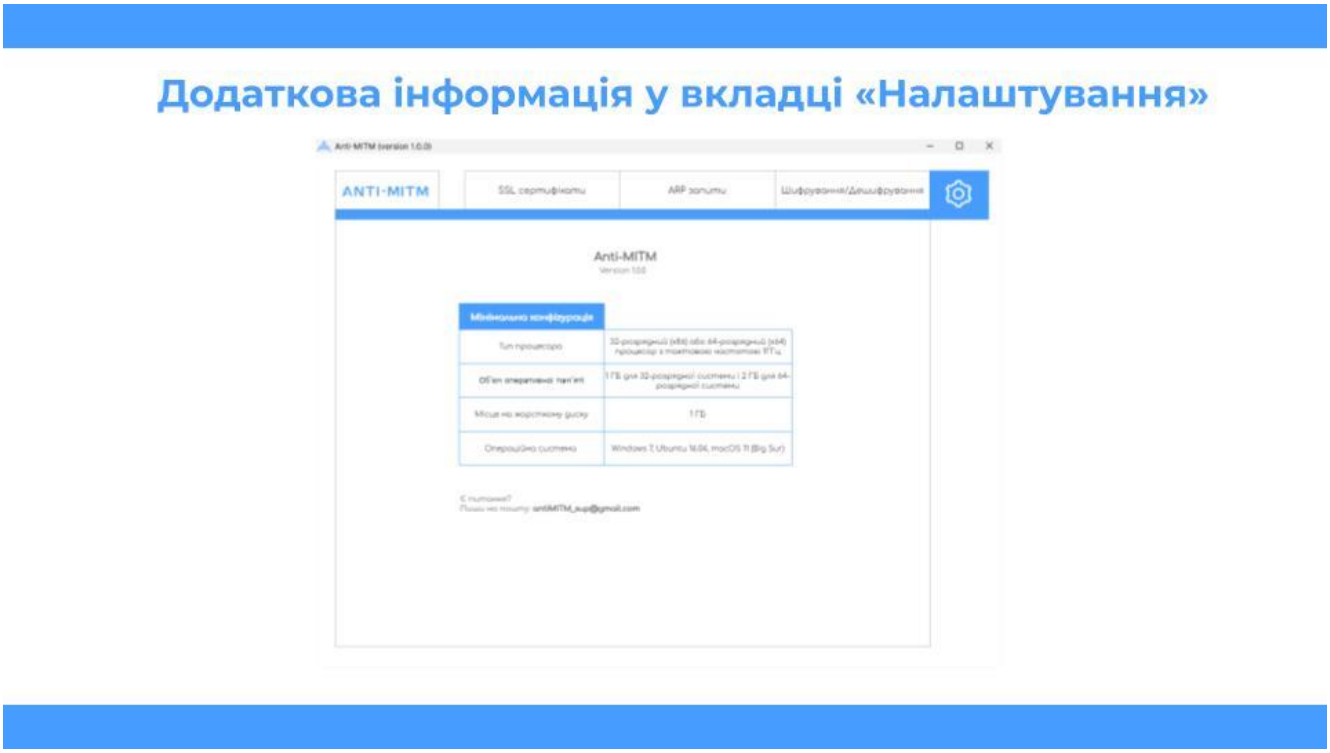

Рисунок В.13 – Додаткова інформація про програму

Рисунок В.14 - Висновок

ПРОТОКОЛ
ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ
НА НАЯВНІСТЬ ТЕКСТОВИХ
ЗАПОЗИЧЕНЬ

Назва роботи: Розробка програми для аналізу, виявлення та захисту від атаки типу Man-in-the-Middle (MITM) у бездротових мережах, включаючи підроблені SSL-сертифікати, аномальні ARP-запити-відповіді та зміни в мережевому трафіку

Тип роботи: бакалаврська дипломна робота
(БДР, МКР)

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

Показники звіту подібності Unicheck

Оригінальність 98 % Схожість 2 %

Аналіз звіту подібності (відмітити потрібне):

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

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

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

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

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

Автор роботи _____ Матяш-Рибачук О.С.
(підпис) (прізвище, ініціали)

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