Київський національний університет імені Тараса Шевченка

Факультет інформаційних технологій

Кафедра програмних систем і технологій

УДК 004.42

На правах рукопису

ВИПУСКНА КВАЛІФІКАЦІЙНА БАКАЛАВРСЬКА РОБОТА

Тема: “Розроблення та програмна реалізація модифікованої стеганографічної моделі передачі даних в протоколі IPv6”

Спеціальність – 121 “Інженерія програмного забезпечення”

ПОЯСНЮВАЛЬНА ЗАПИСКА
ВКБР.ІПЗ - 22.00.00.000 ІПЗ

Студент

ІПЗ-42 _____ /Максим ІВАСЕНКО/

Науковий керівник

к. ф.-м. н., доц._____ /Ольга СУПРУН/

Консультант
з питань нормоконтролю

_____ /Тамара ЧАПОВСЬКА/

Допускається до захисту
Завідувач кафедри

д.т.н., проф. _____ /Олексій БИЧКОВ/

Київ – 2021

Рішенням Екзаменаційної комісії
випускна кваліфікаційна робота студента

_____

захищена з оцінкою

_____

Голова Екзаменаційної комісії
д. т. н., проф. Віктор ВИШНІВСЬКИЙ

Київський національний університет імені Тараса Шевченка
Факультет інформаційних технологій
Кафедра програмних систем і технологій
Освітньо-кваліфікаційний рівень бакалавр
Спеціальність 121 “Інженерія програмного забезпечення”

ЗАТВЕРДЖЕНО
Зав. кафедри програмних систем і технологій
_____(Олексій БИЧКОВ)

ЗАВДАННЯ

НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТУ

Івасенку Максиму Володимировичу

1. Тема бакалаврської роботи “Розроблення та програмна реалізація модифікованої стеганографічної моделі передачі даних в протоколі IPv6”
керівник роботи Ольга СУПРУН, к.ф.-м.н., доц.
затверджені на засіданні кафедри програмних систем і технологій, протокол №6 від «11» листопада 2020 р.
2. Строк подання студентом роботи «01» червня 2021 р.
3. Вихідні дані до роботи Стеганографічні моделі вбудовування інформації у протоколах IPv4, IPv6; теоретичні концепції мережевої стеганографії
4. Зміст розрахунково - пояснювальної записки (перелік питань, які потрібно розробити)
1. Аналіз існуючих мережевих стеганографічних моделей.
2. Проектування моделі для вбудовування, шифрування та дешифрування даних.
3. Програмна реалізація стеганографічної моделі та алгоритмів.
4. Аналіз отриманих результатів, порівняння із існуючим підходом.
5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)
1. Зображення прихованого каналу передачі даних у моделі (рис. 2.8, ст. 41)
2. Зображення загальної архітектури системи (рис. 3.1, ст. 45)
3. Зображення реалізованої архітектури системи (рис. 3.2, ст. 47)
4. Порівняння продуктивності моделей на стороні відправника (рис. 3.3, ст. 50)
5. Порівняння продуктивності моделей на стороні отримувача (рис. 3.4, ст. 51)
6. Консультанти розділів проекту (роботи)

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
Розділ 1 “Аналіз підходів до побудови моделей у мережевій стеганографії”	Супрун О.М.	20.01.2021	20.01.2021
Розділ 2 “Проектування стеганографічної моделі передачі даних по протоколу IPv6”	Супрун О.М.	20.01.2021	20.01.2021
Розділ 3 “Програмна реалізація моделі та аналіз результатів”	Супрун О.М.	20.01.2021	20.01.2021

7. Дата видачі завдання 13 жовтня 2020 р.

Керівник _____/Ольга СУПРУН/

Завдання прийняв до виконання _____/Максим ІВАСЕНКО/

КАЛЕНДАРНИЙ ПЛАН

№ п/п	**Назви етапів бакалаврської роботи**	**Строк виконання етапів роботи**	**Примітка**
1	Уточнення постановки задачі	15.10.2020-12.11.2020	Виконано
2	Підбір і вивчення літератури	13.11.2020-14.12.2020	Виконано
3	Аналіз існуючих методів, концепцій, моделей та алгоритмів вирішення завдання	15.12.2020-19.01.2021	Виконано
4	Обгрунтування вибору рішення	20.01.2021-25.01.2021	Виконано
5	Вивчення існуючих підходів до розробки моделі в протоколі IPv6 зі стеганографічним каналом	26.01.2021-13.02.2021	Виконано

6	Розробка стеганографічної моделі	14.02.2021-25.03.2021	Виконано
7	Опис розробленої моделі	26.03.2021-02.04.2021	Виконано
8	Програмна реалізація мережевої моделі для вбудовування інформації	03.04.2021-22.04.2021	Виконано
9	Аналіз результатів та порівняння із існуючими моделями	23.04.2021-12.05.2021	Виконано
10	Оформлення і друк пояснювальної записки	13.05.2021-23.05.2021	Виконано
11	Оформлення презентації	24.05.2021-26.05.2021	Виконано
12	Отримання рецензії	30.05.2021	
13	Затвердження пояснювальної записки роботи завідувачем кафедри	1.06.2021-12.06.2021	
14	Захист дипломної роботи	22.06.2021	

Студент – бакалавр _____/Максим ІВАСЕНКО/

Керівник роботи _____/Ольга СУПРУН/

АНОТАЦІЯ

Випускна кваліфікаційна бакалаврська робота: 66 с., 17 рис., 3 табл., 3 додат., 20 джерел.

Тема: Розроблення та програмна реалізація модифікованої стеганографічної моделі передачі даних в протоколі IPv6.

Об'єкт дослідження: мережева стеганографія.

Мета роботи: підвищення надійності та стійкості процесу передачі даних по стеганографічному каналу в протоколі IPv6.

Предмет дослідження: стеганографічна модель вбудовування інформації в проколі IPv6.

Результати дослідження:

Досліджено підходи до реалізації мережевої стеганографії та існуючі алгоритми. Запропоновано стеганографічну модель для вбудовування інформації при передачі даних по протоколу IPv6. Розроблено додатки для демонстрації роботи моделі, проведено аналіз продуктивності розробленої та існуючої систем.

Висновок

В результаті дослідження було отримано підхід, що значно підвищує надійність та стійкість моделі за рахунок використанням протоколу Діффі-Хеллмана на еліптичних кривих (ECDH) для шифрування та методу цифрового підпису ECDSA. В запропонованый моделі забезпечується перевірка цілісності та неушкодженості пакетів на стороні отримувача.

СТЕГАНОГРАФІЯ, МОДЕЛЬ, ПРОТОКОЛ IPV6, МЕРЕЖЕВА СТЕГАНОГРАФІЯ, АЛГОРИТМИ.

АННОТАЦИЯ

Выпускная квалификационная бакалаврская работа: 66 с., 17 рис., 3 табл., 3 доп., 20 источников.

Тема: Разработка и программная реализация модифицированной стеганографической модели передачи данных в протоколе IPv6.

Объект исследования: сетевая стеганография.

Цель работы: разработка и модификация стеганографической модели для повышения надежности и устойчивости процесса передачи данных по стеганографическому каналу в протоколе IPv6.

Предмет исследования: стеганографическая модель встраивания информации в проколе IPv6.

Результаты исследования:

Исследованы подходы к реализации сетевой стеганографии и существующие алгоритмы. Предложено стеганографическую модель для встраивания информации при передаче данных по протоколу IPv6. Разработаны приложения для демонстрации работы модели, проведен анализ производительности разработанной и существующей систем.

Вывод

В результате исследования был получен подход, что значительно повышает надежность и устойчивость модели за счет использованием протокола Диффи-Хеллмана на эллиптических кривых (ECDH) для шифрования и метода цифровой подписи ECDSA. В предложенной модели обеспечивается проверка целостности и невредимости пакетов на стороне получателя.

СТЕГАНОГРАФИЯ, МОДЕЛЬ, ПРОТОКОЛ IPV6, СЕТЕВАЯ СТЕГАНОГРАФИЯ, АЛГОРИТМЫ.

ANNOTATION

Final qualifying bachelor's work: 66 p., 17 fig., 3 tab., 3 app., 20 sources.

Topic: Development and software implementation of a modified steganographic model of data transmission in the IPv6 protocol.

Object of research: network steganography.

Purpose: development and modification of the steganographic model to improve the reliability and stability of the data transmission process over the steganographic channel in the IPv6 protocol.

Subject of research: a steganographic model of embedding information in an IPv6 protocol.

Results of the study:

Approaches to the implementation of network steganography and existing algorithms are investigated. A steganographic model for embedding information when transmitting data over IPv6 is proposed. Applications have been developed to demonstrate the operation of the model, and the performance of the developed and existing systems has been analyzed.

Conclusion

As a result of the study, an approach was obtained that significantly increases the reliability and stability of the model by using the Diffie-Hellman protocol on elliptic curves (ECDH) for encryption and the ECDSA digital signature method. In the proposed model, the integrity and integrity of packets is checked on the recipient side.

STEGANOGRAPHY, MODEL, IPV6 PROTOCOL, NETWORK STEGANOGRAPHY, ALGORITHMS.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

TCP – Transmission Control Protocol (Протокол керування передачею)

TCP/IP - Transmission Control Protocol (TCP)/Internet Protocol (IP) (Протокол керування передачею / Інтернет протокол)

IPv6 - Internet Protocol version 6 (Інтернет протокол шостої версії)

CBC-RC6 - Cipher block chaining - Rivest cipher 6 (Режим блочного шифрування – Rivest шифр 6)

RSA алгоритм - Rivest–Shamir–Adleman алгоритм

ECDH - Elliptic curve Diffie–Hellman (Протокол Діффі-Хеллмана на еліптичних кривих)

ECDSA - Elliptic Curve Digital Signature Algorithm (Алгоритм цифрового підпису на еліптичних кривих)

MAC - Message Authentication Code (Код автентифікації повідомлення)

ВСТУП

Актуальність роботи

Криптографія та стеганографія - інтерактивні методи для конфіденційного спілкування. Використання лише криптографії або стеганографії недостатньо для захисту секретних даних. Таким чином, для підвищення рівня захисту інформації та збереження секретності та конфіденційності даних обидва методи використовуються разом. Криптографія може використовуватися там, де стеганографія неефективна, а стеганографія - там, де неефективна криптографія. Обидва методи захищають по-своєму, але додавання декількох рівнів захисту завжди вважається гарною практикою для використання комбінації даних методів.

Мета і задачі дослідження

Метою бакалаврської роботи є підвищення надійності та стійкості процесу передачі даних по стеганографічному каналу в протоколі IPv6. Розроблена модель має використовувати новітні алгоритми шифрування та вбудовування даних для забезпечення стійкості стеганографічної моделі до атак на стегосистему. Комбінування новітнього підходу криптографії та стеганографії дозволить розробити надійну мережеву стеганографічну модель, за допомогою якої можна вбудовувати і передавати інформацію отримувачу таємно від усіх.

Досягнення мети включало розв’язання таких **задач**:

1) огляд існуючих моделей та алгоритмів;
2) вивчення існуючих підходів до розробки моделі в протоколах IPv6;
3) проектування стеганографічної моделі;
4) програмна реалізація мережевої моделі для вбудовування інформації;
5) порівняння розробленої моделі із існуючим підходом.

Об’єктом дослідження є мережева стеганографія.

Предметом дослідження є стеганографічна модель вбудовування інформації в проколі IPv6.

Методи дослідження

Для дослідження існуючих підходів до мережевої стеганографії використовувався метод аналізу та порівняння.

Наукова новизна отриманих результатів

Досліджено можливості використання мережевого протоколу IPv6 для вбудовування інформації. Запропоновано стеганографічну модель для вбудовування інформації. Виконана програмна реалізація розробленої моделі та проведено її аналіз із існуючою новітньою моделлю.

Практичне значення одержаних результатів

Одержаний підхід до розробки стеганографічної моделі може використовуватись для побудови реальних масштабних мережевих моделей для прихованим обміном інформацією.

Використано новітні подходи криптографії та стеганографії. Доведена ефективність використання модифікованої моделі у порівнянні з існуючими реалізаціями.

Особистий внесок студента

Основним результатом є:

1. запропонований автором підхід до розробки стеганографічної моделі із використанням протоколу IPv6;

2. приведена реалізація моделі, що забезпечує новітній підхід до таємного обміну інформацією.

Публікації

Івасенко М.В., Супрун О.М. (Ivasenko M., Suprun O.) Розроблення та програмна реалізація модифікованої стеганографічної моделі передачі даних в протоколі IPv6. MSTIoE 2021-8. 8-ма Східно-Європейська конференція “Математичні та програм технології Internet of Everything” (14.05.2021, Київ). Зб.матер., КНУ ім.Т.Шевченка

Структура та обсяг роботи

Робота викладена на 66 сторінках друкованого тексту, який складається із вступу, трьох розділів, висновків, списку використаних джерел (20 найменувань). Робота містить 3 таблиці, 17 рисунків та 3 додатки, обсягом 14 стор.

РОЗДІЛ 1

АНАЛІЗ ПІДХОДІВ ДО ПОБУДОВИ МОДЕЛЕЙ У МЕРЕЖЕВІЙ СТЕГАНОГРАФІЇ

1.1. Класифікація мережевої стаганографії

Захист інформації, що надсилається через мережу, може здійснюватися за допомогою принципів криптографії. Вважається, що використання шифрування достатньо для безпечного зв'язку в мережі. Однак зловмисник може виявити існування зашифрованого каналу між двома віддаленими вузлами зв’язку та розшифрувати захоплений трафік. Стеганографія усуває дану проблему, приховуючи існування повідомлень. Стеганографія - це здатність та вміння писати приховані повідомлення на носій таким чином, що ніхто, окрім відправника та запланованого одержувача, не підозрює про існування повідомлення. В більшості, стеганографічні програми в якості носія для приховування даних використовують файли звуку, зображення та відео. Відповідно до [1], [2] стеганографія може застосовуватися в цифрових водяних знаках для захисту авторських прав у різних цифрових аудіо-, відео- та програмних об'єктах. Приховування даних на рівні мережі, таких як протоколи, є відносно новим, але водночас це порушує важливе питання мережевої безпеки. Усі методи приховування інформації, які можуть бути використані для обміну секретними даними в комп'ютерних мережах, можна об’єднати під загальним терміном мережевої стеганографії. Відмінні від типових стеганографічних методів, які використовують цифрові носії інформації (зображення, аудіо- та відеофайли) як носій для приховування даних, мережева стеганографія використовує протоколи зв'язку, контрольні поля та їх основну заздалегідь визначену функціональність.

Типові мережеві методи стеганографії або мережеві приховані канали використовують певні властивості комунікаційного середовища таким чином, щоб передавати секретну інформацію через носій, не привертаючи уваги нікого, крім

суб'єктів, що експлуатують прихований канал. Мережева стеганографія є синонімом прихованих каналів, поділених на три широкі категорії (рис. 1.1.):

1. Методи модифікації заголовка або корисного навантаження мережевого пакета.
2. Методи модифікації структури пакетних потоків.
3. Гібридні схеми.

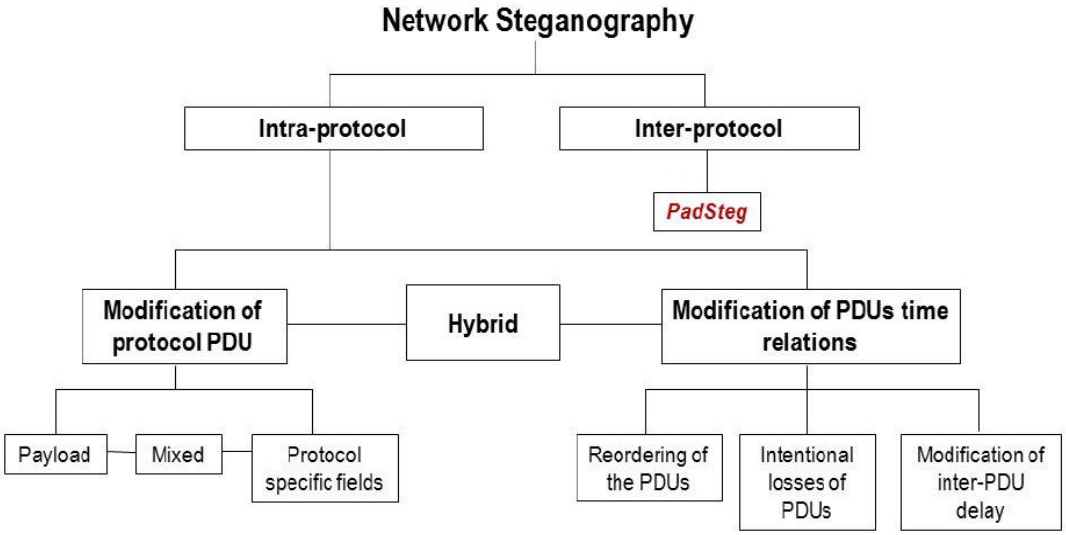

Рис. 1.1. Класифікація мережевої стеганографії

1.1.1. Методи модифікації заголовка мережевого пакета

У даному методі приховування даних здійснюється шляхом модифікації полів, специфічних для протоколу. Наприклад, заголовки TCP, IP або UDP модифіковані для вбудовування секретних повідомлень, як це зазначено в [1] та [4]. Усі стеганографічні методи за даним підходом мають високу стеганографічну ємність (здатність приховувати достатньо великий об’єм даних використовуючи одиницю носія). Деякі стеганографічні методи, засновані на прикладному рівні, змінюють корисне навантаження пакетів. Існує також метод, який передбачає приховування даних як у заголовку, так і в корисному навантаженні мережевого пакету, як зазначено в [3] та HICCUPS [3] (Прихована система зв'язку для пошкоджених мереж). Даний метод забезпечує високу стеганографічну здатність, але реалізація є складнішою, ніж будь-який інший спосіб. Це потребує перепрограмування мережевих карт інтерфейсу. Недоліками є збільшення частоти помилок у пакеті.

1.1.2. Методи модифікації структури пакетних потоків

Приховування даних також можна здійснити шляхом модифікації пакетних потоків мережі, як описано в [5]. Деякі з прикладів у цьому методі - це ті, що впливають на порядок послідовності пакетів [8], ті, що змінюють затримку між пакетами [9], та ті, що вносять навмисні втрати, пропускаючи номери послідовностей у відправника [10]. Основна проблема цих схем включає синхронізацію між відправником та одержувачем. Іншим недоліком є те, що затримки можуть вплинути на якість передачі.

1.1.3. Гібридні схеми

У гібридній схемі заголовок пакета та їхні часові залежності змінюються. Стеганографія втрачених аудіопакетів [6,7] та стеганографія ретрансляції є одним із прикладів, які підпадають під цю схему. У порівнянні з іншими методами, даний метод має більші стеганографічні можливості (можливість модифікувати підхід до приховування інформації за рахунок комбінування). Ще однією перевагою цього методу є те, що його важко виявити, тобто метод є стійкий до стегоаналізу та атак.

1.2. Прихований канал

Прихований канал - це носій, в якому може проходити інформація, але цей носій зазвичай не використовується для обміну інформацією. Вперше приховані канали введено Лампсоном [11]. Прихований канал визначається як будь-який канал зв'язку, який може використовуватися процесом для передачі інформації таким чином, що порушує правило захисту системи. Теоретично, майже будь-який процес або двійкові дані можуть бути прихованим каналом. Приховані канали можна розділити на дві категорії: прихований канал зберігання та прихований канал синхронізації.

Прихований канал зберігання: У прихованому каналі зберігання відправник та одержувач використовують спільну змінну, яка буде вставляти в нього приховані дані, а інша людина буде зчитувати з нього приховані дані. У мережевому середовищі поля заголовка будуть діяти як спільні змінні. Один із процесів безпосередньо або опосередковано записує до певного місця зберігання, який інший процес читає з цього

конкретного місця зберігання. Декілька інструментів використовують протоколи TCP, IP, ICMP та HTTP для встановлення прихованого каналу зберігання. У цих протоколах використовуються невикористані поля для передачі інформації, оскільки ці поля зазвичай не виявляються системами виявлення вторгнень та брандмауерами.

Приховані канали синхронізації: У каналі синхронізації одержувач та відправник за попередньою згодою узгоджують інтервал синхронізації та початковий протокол. Протягом кожного часового інтервалу відправник або передає один пакет, або зберігає мовчання. Приймач контролює кожен інтервал, щоб визначити, чи був прийнятий пакет. Варто звернути увагу, що вихідні дані, що передаються по каналу, є двійковими, але фактична інтерпретація двійкового потоку залежить від сторін, що спілкуються. Прихований канал синхронізації фокусується на передачі повідомлення через шаблон надходження пакетів, а не на вміст повідомлення. Крім того, прихований канал синхронізації не використовує заголовок пакета або корисне навантаження для кодування прихованих повідомлень. Прихований канал синхронізації поділяється на два канали: канали сортування пакетів та канали синхронізації, де інформація передається за порядком надходження пакетів у канал сортування пакетів. З іншого боку, інформація за тимчасовим каналом передається за наявністю або відсутністю пакета, заданого інтервалом часу.

1.3. Мережева стеганографія

Стек мережевого протоколу має різні рівні, що містить різні поля заголовків для належного зв'язку. Ці поля можна використовувати як приховані канали зберігання для таємного спілкування. Модель OSI - це стандартна модель мережі, з якою порівнюються майже всі поточні моделі мережі. Модель OSI включає протоколи TCP, IP та ICMP, які реалізовані на різних рівнях. Рис. 1.2. показує ієрархію мережевої стеганографії.

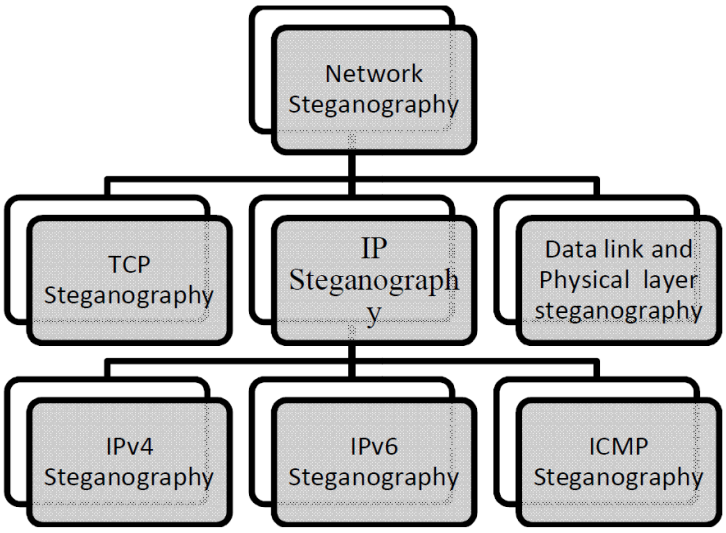

Рис. 1.2. Ієрархія мережевої стеганографії

1.3.1. TCP стеганографія

Заголовок протоколу управління передачею містить різні поля для зв'язку. Кожне поле має свої індивідуальні властивості і використання. Приховані поля можуть бути корисні для приховування інформації. Ці поля діють як носій для стеганографії. Початковий порядковий номер (ISN), що генерується ОС, варіюється від ОС до ОС. Автор [12] пояснив, як OPEN BSD і Linux будуть генерувати ISN. Для цілей приховування даних це поле служить ідеальним середовищем для передачі мережею Ітернет через його розмір і природу [13]. У TCP / IP є деякі поля, зарезервовані для майбутнього використання або невикористані, зокрема:

1. 4-бітне зарезервоване поле в заголовку TCP,
2. поля Padding і Options [17] в TCP/IP, як показано на рис. 1.3,
3. невикористовувані біти поля типу служби заголовка IP (TOS), які можуть використовуватися для кодування секретних даних.

Існують також деякі невикористовувані комбінації, які роблять можливими приховані канали. У 8-бітному полі прапора TCP ECE і CWR додаються RFC 3168 [22] для управління перевантаженням в мережі, включеної в ECN (явне повідомлення про перевантаження), а інші шість бітів використовуються для інтерпретації інших полів заголовка TCP.

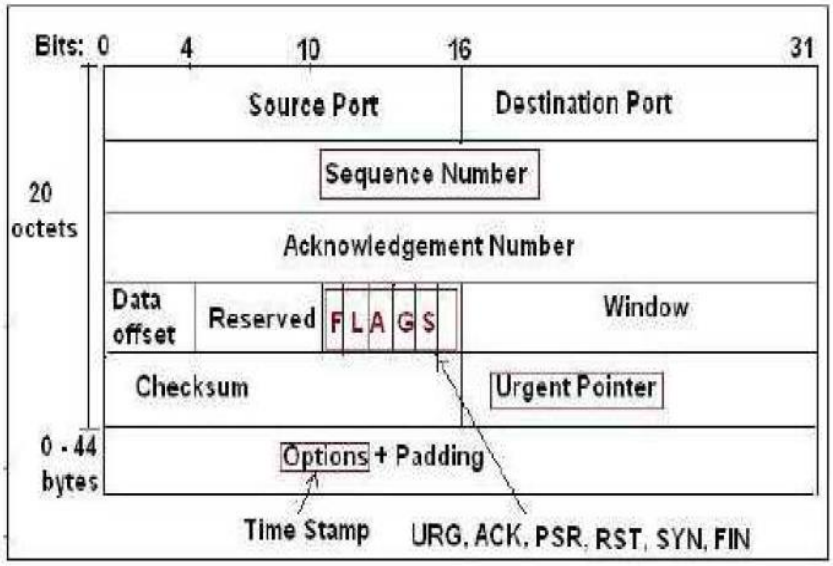

Рис. 1.3. TCP заголовок

Існують 64 можливі комбінації для 6-бітного поля прапора, з них 29 вважаються дійсними згідно з правилами, встановленими протоколом. Наприклад, поле термінового вказівника (16 біт) є суттєвим лише тоді, коли встановлено URG. Якщо URG не встановлено, поле Urg Point у заголовку TCP стає надлишковим, і тому його можна використовувати для прихованого зв'язку.

1.3.2. IPv4 стеганографія

Як і TCP, протокол IPv4 також містить ряд полів у заголовку, що допомагає приховати інформацію, як показано на рис. 1.4. Прапори IP, поле ідентифікації IP, зміщення фрагмента IP, поле параметрів IP і тип служби IP в заголовку IPv4 використовуються для приховування секретної інформації. Кілька дослідників зіткнулися з різними методами приховування, заснованими на вищевказаних каналах. Ахсан, Д. Кундур [14] згадав чотири джерела даних, засновані на ідентифікації IP і біті DF. У [14] ідея авторів полягає в маніпулюванні полем IP ID. Поле ідентифікації пакета призначається відправником. Це випадкове число, згенероване під час створення пакета.

При фрагментації використовується поле ідентифікації. Переконавшись, що фрагментація не відбудеться через розмір пакета; можна приховати дані в цьому полі без будь-яких наслідків при передачі. Перевага цієї роботи полягає в тому, що вона використовується для передачі інформації з точки в точку, але обмеження - це

кількість інформації, яку можливо відправити. Крім того, якщо за будь-яких обставин датаграма фрагментована, приймач буде спостерігати шум при передачі, тому що він буде отримувати одну і ту ж інформацію більше одного разу з кожним новим фрагментом датаграми.

У [16] проаналізували можливі приховані канали протоколів в TCP/IP і запропонували нову ефективну схему, звану методом відновлення фази (PRM), для ідентифікації прихованих каналів в полях ідентифікації TCP і IP. У [14] робота зосереджена на маніпулюванні бітом Do Not Fragment Bit. Там можна вказати, якщо не хочемо, щоб пакет був фрагментований маршрутизаторами в дорозі.

Дослідження Aclose [15] показує, що існує надмірність у стратегії фрагментації Інтернет-протоколу. Поле прапорів містить інформацію про фрагментацію. Перший біт зберігається, другий позначається DF (для представлення Do Not Fragment), а третій - MF (для представлення More Fragment). Нефрагментована датаграма містить всю інформацію про нульову фрагментацію (тобто MF = 0 і 13-бітний FragmentOffset = 0), що призводить до умови надмірності, тобто DF (не фрагментувати) може містити або "0", або "1" за умови знання максимального розміру датаграми. Цей аспект використовується в сценарії приховування даних. Деякі поля можуть бути змінені для кодування секретної інформації. TTL (Time to live). В IP-заголовку - це значення лічильника, яке зменшується при кожному стрибку. Коли TTL досягає нуля, пакет відкидається. Це призводить до того, що пакети в циклах маршрутизації в кінцевому підсумку перекриваються. Поточне рекомендоване значення TTL за замовчуванням для IPv4 (версія 4 протоколу інтернету) становить 64 або 128, залежно від різних операційних систем (ОС). Однак це можна змінити. Сандер [19] запропонував кодувати приховану інформацію за допомогою двох символів: сигнали низького TTL - "0", а сигнали високого TTL - "1".

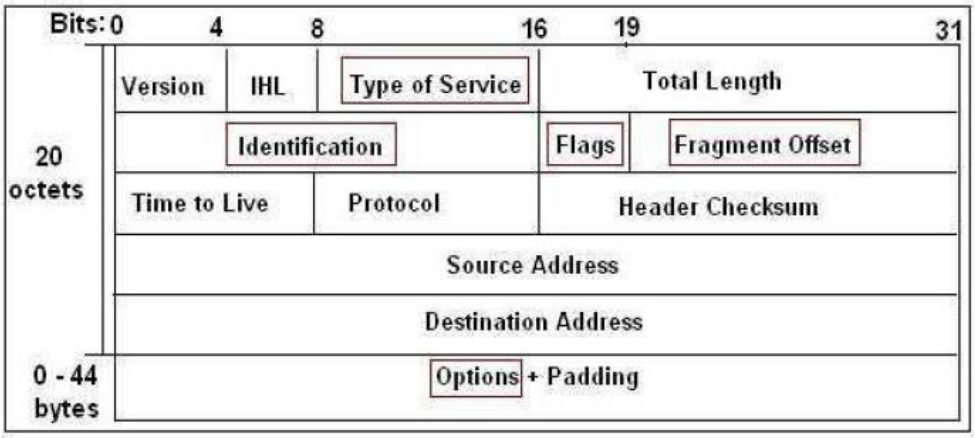

Рис. 1.4. IPv4 заголовок

1.3.3. IPv6 стеганографія

Спочатку представлений в [5] і [6], IPv6 був розроблений для поліпшення IPv4 в різних областях, наприклад, мобільності, безпеки та адресації. Однак розгортання IPv6 в основному обумовлено його 128-бітним адресним простором, що дозволяє йому відновлюватися після проблем, викликаних недостатньою кількістю адрес IPv4. Через повільне розгортання IPv4 очікується, що ці два протоколи будуть співіснувати протягом тривалого періоду, тому були запропоновані належні перехідні механізми [16]. Що стосується мережевих прихованих каналів, орієнтованих на IPv6, посилання [9] і [8] показують кілька стеганографічних методів, що вбудовують дані в заголовок або в додаткові розширення. Щоб оцінити доцільність використання прихованих каналів IPv6, слід розглянути 6 методів, націлених на заголовок, який зображений на рис 1.5. Використовувані поля і пов'язані з ними механізми приховування описані нижче.

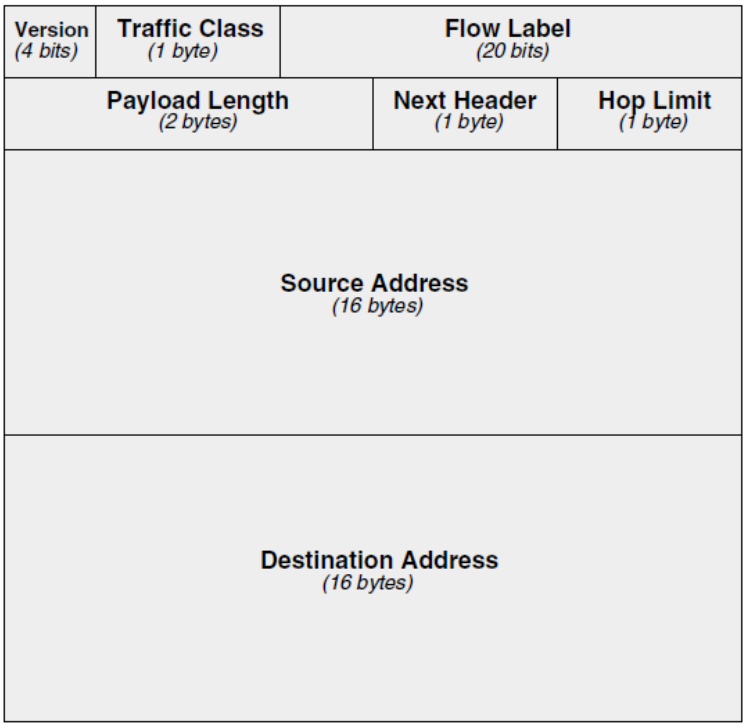

Рис. 1.5. IPv6 заголовок

1. Traffic Class (клас трафіку): це 8-бітне поле, що вказує очікувану від мережі послугу. Перші 6 біт визначають кодову точку диференційованих послуг (DSCP) і класифікують трафік відповідно до критеріїв якості. Решта 2 біта використовуються для явного повідомлення про перевантаження (ECN) для управління потоком в наскрізному режимі. Інформація, що міститься в класі трафіку, може бути замінена прихованими даними, щоб встановити прихований канал з пропускною здатністю 8 біт / пакет. Це поле може бути змінено проміжними вузлами, що призведе до порушення прихованого каналу.
2. Flow Label (мітка потоку): вона має довжину 20 біт і допомагає вузлам мережі направляти трафік по найбільш відповідному шляху [1]. У загальному випадку мітки повинні бути псевдовипадковими, а майбутні значення не повинні бути передбачуваними. Проміжні вузли не повинні перемикати мітки, щоб не порушувати потік. Частина бітів, що складають мітку потоку, може бути замінена прихованими даними, що ведуть до прихованого каналу з пропускною здатністю 20 біт/пакет.

3. Payload Length (довжина корисного навантаження): вона визначає розмір поля даних датаграми, який може становити до 65,536 байт. Інформація може бути прихована шляхом маніпулювання довжиною корисного навантаження, щоб додати довільні дані до корисного навантаження. Щоб уникнути неправильної поведінки IPv6protocol, контрольна сума повинна бути належним чином оновлена, щоб запобігти відкиданню пакетів проміжними вузлами. Крім того, прихована інформація повинна бути видалена до того, як датаграма буде доставлена одержувачу. Пропускна здатність прихованого каналу варіюється в залежності від обсягу вбудованих даних, який не може перевищувати максимальний розмір датаграми.
4. Next Header (наступний заголовок): він визначає наступний заголовок, який присутній в корисному навантаженні пакета. Типові значення: 6 - TCP, 58 - ICMPv6, 17 - UDP і 1 - ICMP. У разі розширень найбільш поширеними конкретними значеннями є 0 - перехід по переходу, 44 - фрагмент, 60 - призначення, 51 - автентифікація і 43 - маршрутизація. Інформацію можна приховати, змінивши наступний заголовок так, щоб він вказував на "фіктивний" додатковий заголовок, що містить дані. Як і в попередньому випадку, датаграма IPv6 повинна бути належним чином відновлена, перш ніж вона буде доставлена в пункт призначення. Результуюча пропускна здатність варіюється в залежності від розміру підроблених заголовків.
5. Hop Limit (межа переходу): він визначає максимальну кількість "переходів", тобто вузлів, які може пройти пакет. Оскільки він має довжину 8 біт, межа переходу може мати до 256 значень. Дані можуть бути приховані шляхом збільшення або зменшення значення поля для послідовних пакетів. Потім інформація декодується шляхом порівняння отриманих значень (якщо різні маршрути не порушили секрет). В результаті секретна інформація може передаватися зі швидкістю 1 біт / пакет.
6. Source Address (адреса джерела): містить мережеву адресу джерела. Прихована інформація вставляється шляхом заміни деяких бітів адреси довільними

даними. Максимальна пропускна здатність отриманого прихованого каналу становить 128 біт/пакет.

1.4. Аналіз полів заголовка протоколу IPv6

Як правило, ефективність методу приховування інформації в значній мірі залежить від наявності відповідного носія. Наприклад, введення даних в поле в заголовку призведе до прихованого каналу зі смугою пропускання, пропорційної швидкості передачі пакетів, тобто x біт/пакет, що вводиться в потік у пакетів/с. Крім того, стеганограма (тобто носій плюс вбудоване повідомлення) не повинна виглядати як аномалія. Наприклад, у випадку каналів, поля, що містять приховані дані, не повинні занадто сильно відхилятися від середніх значень, щоб не анулювати скритність прихованого каналу. Розуміння поведінки відкритого трафіку також має вирішальне значення для розробки відповідних методів виявлення або контрзаходів. На жаль, раніше розглянуті роботи не давали кількісної оцінки пропускної здатності з точки зору прихованих повідомлень або скритності реального трафіку IPv6. З цією метою було взято інформацію з роботи, в якій досліджували дані про трафік, зібрані по магістральному каналу (канал рівня 1) між Чикаго і Сіетлом протягом чотирьох різних днів і надані Центром прикладного аналізу інтернет-даних 1. Для обробки даних використовували користувацькі скрипти Python, бібліотеку Scapy і tshark.

1. Traffic Class (клас трафіку). Це поле є об'єднанням DSCP і ECN. Що стосується DSCP, спостерігали тільки три можливих значення в трасуваннях: 0 (0b000000), що є значенням за замовчуванням для багатьох мережевих пристроїв, спостерігалося в 5,5% пакетів, в той час як 2 (0b000010) і 3(0b000011) в 17,5% і 77,5% пакетів відповідно. Згідно [2], такі значення не вимагають будь-якої спеціальної обробки з боку мережі, тому датаграма може обслуговуватися найкращим чином. Потім, якщо DSCP маніпулюється, щоб містити секретні дані, будь-яке значення, відмінне від спостережуваних, буде представляти аномалію і може бути використано для виявлення прихованого каналу. Замість цього значення ECN були рівні 0 (0b00) в 99,99% зібраних пакетів, отже, поле недостатньо приховано, щоб містити секрети. Як наслідок,

поле класу трафіку може кодувати тільки 3 значення з 28 можливих, тим самим обмежуючи пропускну здатність прихованого каналу максимум до 2 Біт/пакет. Тому оцінка 8 біт / пакет [9] занадто оптимістична для реальних випадків використання.

2. Flow Label (мітка потоку). По-перше, підраховано, скільки пакетів мають мітку "нуль". Отриманий набір даних призвів до неоднозначних результатів. Зокрема, два трасування характеризувалися тим, що 96% пакетів мали нульову мітку потоку, в той час як два інші трасування мали нульове значення в 21% і 24% пакетів відповідно. Отримано недостатньо інформації, щоб пояснити таку поведінку. Згідно [1], нульові значення прийнятні, але не рекомендуються, оскільки вони можуть бути використані не за призначенням. Однак, з точки зору можливості використання мітки потоку з довільними значеннями для вбудовування даних, наявність незначної кількості пакетів з ненульовими значеннями все ще може привести до деякої стеганографічної ємності прихованого каналу мережі. Як правило, мітка потоку не змінюється в залежності від з'єднання, і величезна кількість пакетів з нульовим значенням може обмежити можливість використання в стеганографічних цілях.
3. Payload Length (довжина корисного навантаження). Навіть якщо максимальна довжина пакета для IPv6datagram становить 56,536 байта, значення, що спостерігаються, обмежують пропускну здатність методу, оскільки пакети з незвичайними розмірами можуть бути легко розпізнані як викиди за допомогою аналізатора трафіку або аналізатора протоколу. У використовуваному наборі даних максимальний розмір пакета дорівнював 1, 460 байтам, що є типовим розміром блоку максимальної передачі (MTU), підтримуваного інтерфейсом IEEE 802.3/Ethernet L2. Таке значення також було найбільш поширеним разом з невеликими пакетами по 32 байта, що містять TCP-ACK. Інші спостережувані розміри датаграми IPv6 були рівні 1, 240, 1, 400, 1, 420, і 1, 430 байт. Тому пропускна здатність прихованого каналу, що використовує підхід модуляції корисного навантаження, виявляється меншою за теоретичну [9]. Фактично, припускаючи MTU в 1 500 байт, максимальний

обсяг простору, доступного для транспортування прихованих даних, становить до 1 416 байт, тобто розмір MTU мінус 24 байти, 40 байт і 20 байт, необхідних для розміщення заголовків Ethernet, IPv6 і TCP відповідно.

4. Next Header (наступний заголовок). Це поле має довжину 8 біт, що дозволяє використовувати до 28 можливих значень. Однак зібраний трафік показує, що 99,15% вказали на наявність TCP, в той час як тільки 0,55% і 0,3% вказали на протоколи UDP і ICMP відповідно. Тому, як було запропоновано в [9], введення розширень для приховування даних для реалізації прихованого каналу мережі може бути легко виявлено, оскільки це може являти собою аномалію. Результуюча пропускна здатність може бути дуже обмеженою, оскільки тільки кілька пакетів можуть бути штучно модифіковані, щоб діяти в якості носіїв.
5. Hop Limit (межа переходу). Як і в попередньому випадку, це поле також може кодувати 256 значень. Пакети з обмеженням переходу в діапазоні 51-54 є найбільш поширеними, поряд з пакетами в діапазоні 242-245. Це можна пояснити тим фактом, що 64-це значення за замовчуванням, і більш високі значення можуть бути отримані протоколом виявлення сусідів, який враховує автоматичне налаштування і дозвіл мережевих адрес, не кажучи вже про найбільш важливих операціях. Тому, звертаючи увагу на те, щоб залишатися в таких діапазонах, модуляція HopLimit між сусідніми пакетами дозволяє реалізувати мережеві приховані канали з пропускною здатністю 1 біт/пакет, як це передбачено в [9].
6. Source Address (адреса джерела). Цей метод вкрай ненадійний, так як прихований канал, що змінює адресу джерела, може порушити мережеве з'єднання. У загальному випадку маніпулювання адресами має відбуватися тільки в тому випадку, якщо обидві секретні кінцеві точки розташовані спільно у відкритих вузлах. Тим не менш, широко поширений захист від підміни може легко виявити зміну адреси, тим самим блокуючи спроби прихованого зв'язку.

1.5. Висновок до розділу

Стек мережевого протоколу має різні рівні, що містить різні поля заголовків для належного зв'язку. Ці поля можна використовувати як приховані канали зберігання для таємного спілкування. Встановлено, що стеганограма (тобто носій разом із вбудованим повідомленням) не повинна виглядати як аномалія. Наприклад, у випадку каналів, поля, що містять приховані дані, не повинні занадто сильно відхилятися від середніх значень, щоб не анулювати скритність прихованого каналу. Розуміння поведінки відкритого трафіку також має вирішальне значення для розробки відповідних методів виявлення. Проаналізувавши можливі варіанти використання полів заголовка IPv6 було обрано поле Flow Label як одне з таких, що задовольняє умовам створення стеганографічного каналу та не призводить до виявлення модифікованих пакетів. У загальному випадку мітки потоку (Flow Label) повинні бути псевдовипадковими, а майбутні значення не повинні бути передбачуваними. Проміжні вузли не повинні перемикати мітки, щоб не порушувати потік. Виявлено, що теоретична ємність стеганографічного каналу становить 20 біт / пакет.

РОЗДІЛ 2

ПРОЕКТУВАННЯ СТЕГАНОГРАФІЧНОЇ МОДЕЛІ ПЕРЕДАЧІ ДАНИХ ПО ПРОТОКОЛУ IPv6

2.1. Обгрунтування вибору стеганографічного носія

Проаналізувавши можливі варіанти вибору носія для вбудовування секретної інформації, було вибрано частину заголовка IPv6 – Flow Label (мітка потоку). Довжина даного поля складає 20 біт, що може створювати прихований канал передачі даних із пропускною здатністю 20 біт/пакет. Також у загальному випадку мітки повинні бути псевдовипадковими, а майбутні значення не повинні бути передбачуваними. Дана особливість підходить до створення прихованого каналу, оскільки мітка потоку міститиме з кожним пакетом частину секретного повідомлення, передбачити наступне значення мітки для третьої сторони не є можливим. Проміжні вузли не повинні перемикати мітки, щоб не порушувати потік.

Наведено частину специфікації даної частини заголовка:

20-бітне поле мітки потоку в заголовку IPv6 [RFC2460] використовується вузлом для маркування пакетів потоку. Нульова мітка потоку використовується для позначення пакетів, які не були спеціально відмічені. Класифікатори пакетів можуть використовувати триплет полів мітки потоку, адреси джерела та адреси призначення для ідентифікації потоку, до якого належить конкретний пакет. Значення міток потоку слід вибирати таким чином, щоб їх біти демонстрували високу ступінь мінливості, що робить їх придатними для використання в якості частини вхідних даних для хеш-функції, використовуваної в схемі розподілу навантаження [18]. У той же час треті сторони навряд чи зможуть вгадати наступне значення, яке вибере джерело міток потоку. У статистиці дискретний рівномірний розподіл визначається як розподіл ймовірностей, в якому кожне значення в заданому діапазоні рівномірно розподілених значень (наприклад, послідовність цілих чисел) з рівною ймовірністю буде вибрано в якості наступного значення. Значення в такому розподілі демонструють як мінливість, так і неочевидність. Таким чином, в якості джерела

значень міток потоку переважно наближення до дискретного рівномірного розподілу. Немає точних математичних вимог, що пред'являються до розподілу або методу, використовуваного для досягнення такого розподілу [19].

Мітка потоку призначається потоку вузлом-джерелом потоку. Нові мітки потоку повинні бути обрані (псевдо-)випадковим чином і рівномірно в діапазоні від 1 до FFFFF hex. Загальний діапазон значень для мітки потоку становить 0-0xFFFFF.

Отже, дане поле власними характеристиками цілком задовольняє умови для створення прихованого каналу передачі даних та реалізації стеганографічної моделі.

2.2. Огляд існуючих моделей з використанням Flow Label як носія

2.2.1. Модель з хаотичним методом кодування та RSA шифруванням

Перший підхід, використаний авторами Sandip Bobade, Rajeshawari Goudar, публікація від грудня 2014 року у International Journal of Engineering and Advanced Technology (IJEAT) полягає у наступному (рис. 2.1):

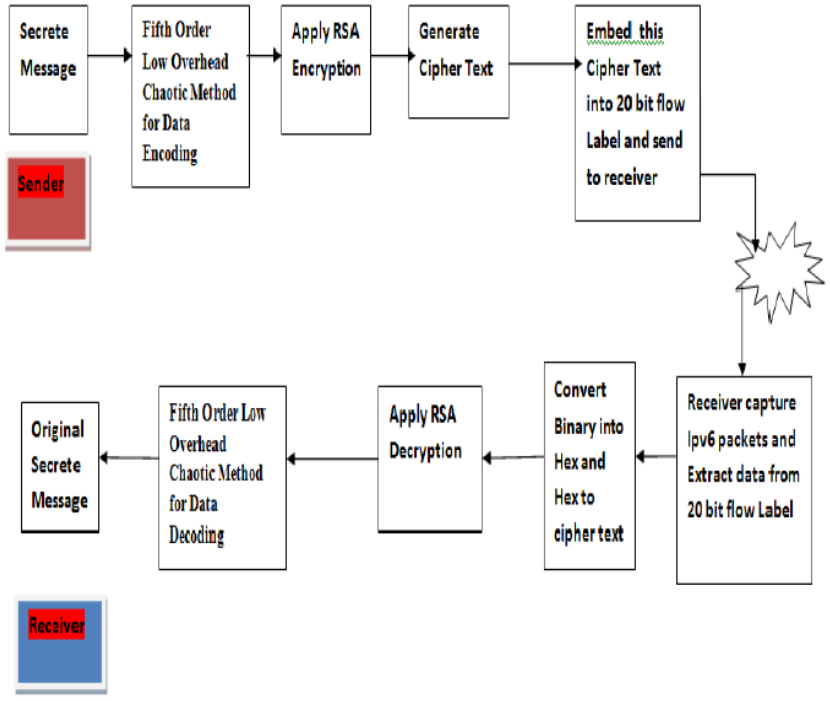

Рис 2.1. Стеганографічна модель (автори Sandip Bobade, Rajeshawari Goudar)

Для кодування використовувався алгоритм хаотичного методу з низькими накладними витратами п'ятого порядку з такими хаотичними картами: логістична хаотична карта, покращена логістична хаотична карта, хаотична карта Чебишева. Для

шифрування – алгоритм RSA. Послідовність обробки повідомлення від відправника до отримувача:

1. Отримання секретного повідомлення від відправника.
2. Застосування хаотичного методу для кодування інформації.
3. Застосування алгоритму шифрування RSA для отримання готового шифр-тексту.
4. Конвертація шифр-тексту в ASCII, після конвертація в HEX-значення.
5. Конвертація HEX-значень в бінарний (двійковий) вигляд.
6. Двійкові дані діляться на 20 біт і зберігаються по 20 біт двійкових даних в кожному пакеті IPv6 в полі Flow Label.
7. Відправка пакетів отримувачу.
8. Отримання пакетів, застосування алгоритмів дешифрування, декодування та виведення секретного повідомлення.

Даний алгоритм був запропонований авторами Sandip Bobade, Rajeshawari Goudar як альтернатива до існуючого підходу LME (Long Mac Encoding).

Серед переваг даного підходу можна виділити:

1. Швидкість кодування, шифрування алгоритму. Набагато швидше за підхід LME (див. рис 2.2.).
2. Стійкість до атак на стегосистему. За рахунок використання асиметричного шифрування RSA, дана модель є більш захищеною.

Серед недоліків даного підходу можна виділити:

1. Неможливо перевірити правильність порядку пакетів.
2. Не реалізований механізм валідації шифр-даних у полі Flow Label.

2.2.2. Модель з алгоритмом шифрування CBC-RC6 та MAC автентифікацією

Другий підхід, використаний авторами Ra’ad A. Muhajjar, Farah A. Badr, публікація від червня 2017 року у International Journal of Engineering & Technology полягає у наступному (рис. 2.2):

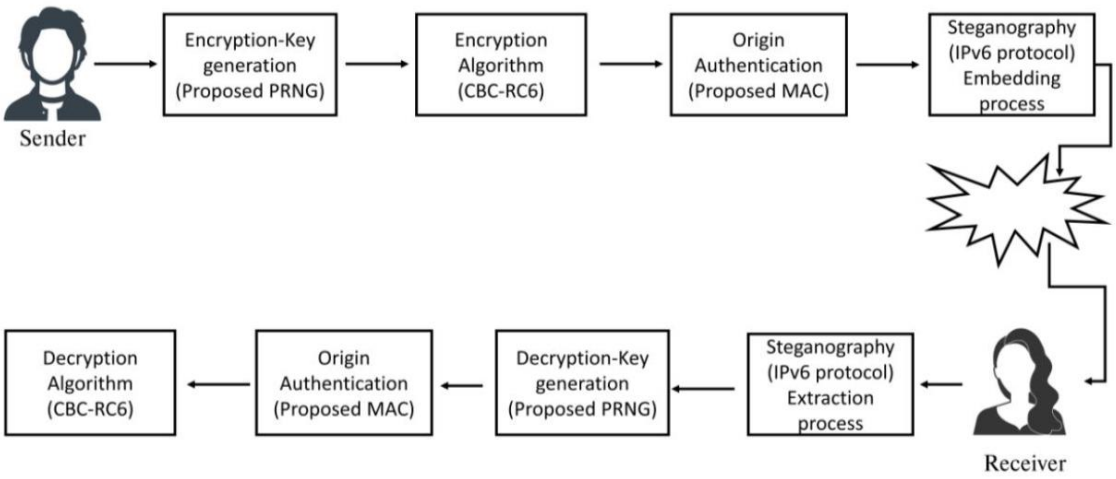

Рис. 2.2. Стеганографічна модель (автори Ra’ad A. Muhajjar, Farah A. Badr)

Для генерації ключа, який буде використовуватися в процесі шифрування/дешифрування, був застосований запропонований генератор псевдовипадкових чисел. Коли генерується ключ шифрування / дешифрування, реалізується алгоритм шифрування CBC-RC6, потім для автентифікації джерела використовується пропонований код автентифікації повідомлення для обчислення MAC, після отримання зашифрованого тексту і MAC обидва значення вбудовані в поле мітки потоку IPv6.

Запропонована система вимагає використання генераторів псевдовипадкових чисел, які здатні генерувати випадкові числа, для генерації ключів шифрування / дешифрування. Пропонований генератор являє собою простий і швидкий генератор псевдовипадкових чисел для генерації ключа шифрування / дешифрування. Таким чином, він повинен бути випадковим або псевдовипадковим, і він повинен бути безпечним.

Послідовність обробки повідомлення від відправника до отримувача:

1. Отримання секретного повідомлення від відправника.
2. Генерація ключа за допомогою генератора псевдовипадкових чисел.
3. Застосування алгоритму шифрування CBC-RC6 для отримання готового шифр-тексту.
4. Автентифікація MAC та розрахунок контрольної суми.
5. Вбудовування шифр-тексту в поле Flow Label.
6. Відправка пакетів отримувачу.

7. Отримання пакетів, застосування алгоритмів дешифрування, декодування та виведення секретного повідомлення.

Характерна особливість та перевага даного підходу полягає у наступному: після шифрування секретного повідомлення та обчислення MAC, повідомлення разом з MAC ховаються в полі мітки потоку (Flow Label). Номери (діапазон значень) міток потоку варіюються від 1 до шістнадцяткового числа FFFFF. 20 біт (тобто 5 шістнадцяткових символів) поля в кожному пакеті будуть використовуватися наступним чином: при вбудовуванні даних в поле мітки потоку перші 8 біт поля будуть використовуватися для ідентифікації послідовності кожного пакета, наступні 8 біт будуть використовуватися для приховування секретних бітів, а останні 4 біта будуть використовуватися для передачі MAC. Пропускна здатність пропонованого каналу складе 8 біт на пакет. Хоча пропозиція звужує пропускну здатність від 20 біт на пакет до 8 біт на пакет, воно забезпечить правильний порядок пакетів в приймачі (приймачах).

Після отримання пакетів від відправника одержувач впорядкує отримані пакети відповідно до їх порядкових номерів і переставляє MAC дані. Одержувач спочатку генерує ключ дешифрування за допомогою запропонованого PRNG (генератор псевдовипадкових чисел), а потім перевірить, чи були отримані дані недоторканими або змінені за допомогою запропонованого MAC. Якщо обчислений MAC збігається з отриманим, то одержувач реалізує розшифровку CBC-RC6; в іншому випадку зашифрований текст буде відхилений, і одержувач запросить повторну передачу пакетів.

Серед недоліків даного підходу можна виділити малу пропускну здатність прихованого каналу, лише 8 біт/пакет та використання алгоритму симетричного блочного шифрування CBC-RC6, що ставить під сумнів надійність даної стеганографічної системи до атак.

2.3. Запропонований підхід до побудови моделі

Враховуючи недоліки попередніх двох підходів, було запропоновано побудувати стеганографічну модель з використанням протоколу Діффі-Хеллмана на

еліптичних кривих для шифрування (Надалі - ECDH) та алгоритмом з відкритим ключем для створення цифрового підпису ECDSA (Надалі - ECDSA).

2.3.1. Протокол ECDH

Еліптична крива Діффі-Хеллмана (надалі - ECDH) - це протокол угоди про ключі, який дозволяє двом сторонам, кожна з яких має пару відкритих і закритих ключів з еліптичною кривою, встановити загальний секрет по відкритому (небезпечному) каналу зв'язку. Цей загальний секрет може бути безпосередньо використаний в якості ключа або для отримання іншого ключа. Потім ключ або похідний ключ можна використовувати для шифрування наступних повідомлень з використанням шифру з симетричним ключем. Це варіант протоколу Діффі-Хеллмана, що використовує криптографію з еліптичними кривими.

ECDH дуже схожий на класичний алгоритм DHKE (обмін ключами Діффі-Хеллмана), але він використовує множення точок ECC замість модульного зведення в ступінь. ECDH заснований на наступному властивості точок EC (2.1):

$$(a * G) * b = (b * G) * a \qquad (2.1)$$

Якщо є два секретні номери a і b (два закриті ключі, що належать Алісі і Бобу) і еліптична крива ECC з генераторною точкою G, можна обмінюватися по відкритому каналу зв'язку значеннями (a * G) і (b * G) (відкриті ключі Аліси і Боба), а потім можна отримати загальний секрет: secret = (a * G) * b = (b * G) * A. Наведене вище рівняння приймає наступний вид:

alicePubKey * bobPrivKey = bobPubKey * alicePrivKey = секрет

Алгоритм ECDH (еліптична крива обміну ключами Діффі–Хеллмана) тривіальний:

1. Аліса генерує випадкову пару ключів ECC:
   {alicePrivKey, alicePubKey = alicePrivKey * G}
2. Боб генерує випадкову пару ключів ECC:
   {bobPrivKey, bobPubKey = bobPrivKey * G}
3. Аліса і Боб обмінюються своїми відкритими ключами через відкритий (небезпечний) канал (наприклад, через Інтернет).

4. Аліса обчислює SharedKey = bobPubKey * alicePrivKey
5. Боб обчислює SharedKey = alicePubKey * bobPrivKey
6. Тепер і Аліса, і Боб мають один і той же

SharedKey == bobPubKey * alicePrivKey == alicePubKey * bobPrivKey

Слід зазначити, що ECDH не забезпечує автентифікацію. Таким чином, протокол вразливий проти атаки типу «Man in the middle» (людина посередині). Тому рішення полягає в тому, щоб використовувати додатково алгоритм цифрового підпису, такий як ECDSA.

2.3.2. Алгоритм ECDSA

ECDSA (алгоритм цифрового підпису з еліптичною кривою) – алгоритм з відкритим ключем для створення цифрового підпису, є наступником алгоритму цифрового підпису (DSA). ECDSA створився, коли два математика на ім'я Ніл Коблиц і Віктор С. Міллер запропонували використовувати еліптичні криві в криптографії. Однак знадобилося майже два десятиліття, щоб алгоритм ECDSA став стандартизованим.

ECDSA - це асиметричний криптографічний алгоритм, побудований навколо еліптичних кривих і базової функції, відомої як “функція люка". "Еліптична крива являє собою набір точок, що задовольняють математичному рівнянню:

$$y^2 = x^3 + ax + b \qquad (2.2)$$

Графік, що представляє еліптичну криву в криптографії еліптичних кривих (ECC) зображений на рис. 2.3.

Еліптична крива виглядає наступним чином:

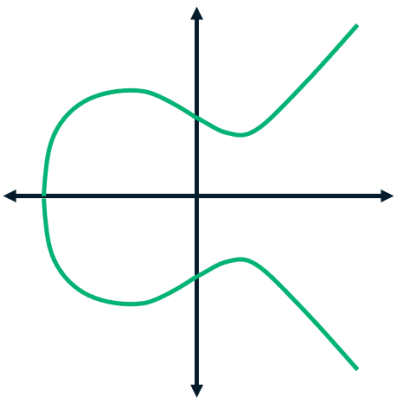

Рис. 2.3. Приклад еліптичної кривої, яка є частиною ECDSA.

Переваги ECDSA проти RSA:

1. Як і всі асиметричні алгоритми, ECDSA працює таким чином, що його легко обчислити в одному напрямку, але дуже важко в зворотному. У випадку ECDSA число на кривій множиться на інше число і, отже, створює точку на кривій. З'ясувати нову точку складно, навіть якщо відомо вихідну точку.
2. У порівнянні з RSA, було встановлено, що ECDSA більш захищений від сучасних методів злому `завдяки своїй складності. ECDSA забезпечує той же рівень безпеки, що і RSA, але робить це при використанні набагато коротших ключів. Тому для більш довгих ключів ECDSA потрібно значно більше часу, щоб зламати атаки з використанням грубої сили.
3. Ще однією великою перевагою, яку ECDSA пропонує в порівнянні з RSA, є перевага продуктивності і масштабованості. Оскільки ECC забезпечує оптимальну безпеку з коротшою довжиною ключа, він вимагає меншого навантаження на мережу і обчислювальну потужність. Це відмінно підходить для пристроїв з обмеженими можливостями зберігання та обробки даних. У сертифікатах SSL / TLS алгоритм ECC скорочує час, необхідний для виконання рукостискань SSL / TLS, і може допомогти швидше завантажити веб-сайт.

Як було зазначено вище, ECDSA вимагає набагато коротших довжин ключів, щоб забезпечити той же рівень безпеки, що забезпечується довгими ключами RSA. Ось як виглядає порівняння ECDSA з RSA:

Таблиця 2.1.

Порівняння необхідної довжини ключів для RSA та ECC

Дані (В бітах)	Необхідна довжина Ключа RSA (в бітах)	Необхідна довжина ключа ECC (в бітах)
80	1024	160-223
112	2048	224-255
128	3072	256-383
192	7680	384-511
256	15360	512+

2.4. Архітектура моделі

Пропонована система спрямована на підвищення рівня безпеки за рахунок використання комбінації двох методів захисту: криптографії та стеганографії. Тому що в той час як стеганографія приховує існування повідомлення, криптографія шифрує саме повідомлення. Для шифрування секретного повідомлення використано протокол ECDH. Після генерації зашифрованого повідомлення та вектора ініціалізації (IV), дані підписуються по алгоритму цифрового підпису ECDSA. Після отримання зашифрованого тексту, вектору ініціалізації, цифрового підпису , дані вбудовуються в поле мітки потоку IPv6, окрім цифрового підпису, який надсилається в payload протоколу IPv6.

Загальна архітектура побудованої моделі зображена на рис. 2.4:

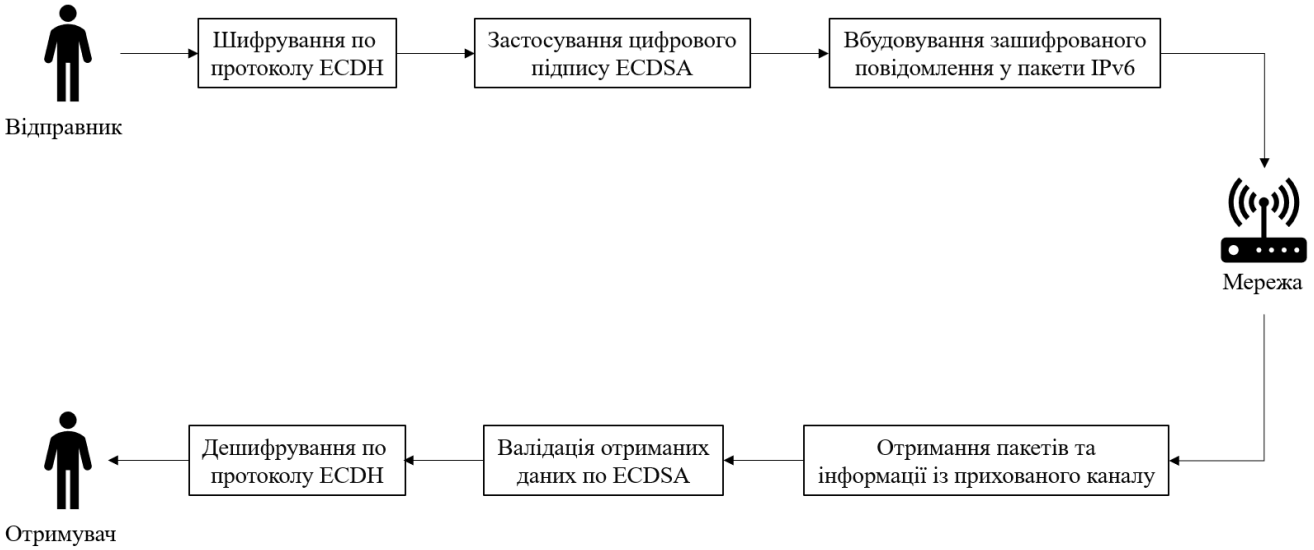

Рис. 2.4. Архітектура побудованої моделі

На першому етапі введене секретне повідомлення відправником шифрується за протоколом Діффі-Хеллмана на еліптичних кривих (ECDH). В результаті отримаємо зашифроване повідомлення та вектор ініціалізації (IV), який буде передаватися разом із готовим повідомленням.

На другому етапі зашифроване повідомлення та вектор ініціалізації оброблюється алгоритмом цифрового підпису ECDSA, що дозволить отримувачу валідувати дані та бути впевненим у тому, що повідомлення є цілісним та не було змінене / ушкоджене. В результаті отримаємо цифровий підпис, який передається отримувачу по протоколу.

На третьому етапі відбувається процес стеганографії – зашифроване повідомлення разом із вектором ініціалізації вбудовуються у поле мітки потоку Flow Label (відбувається процес створення секретного стеганографічного каналу зв’язку). Мітка потоку має розмір 20 біт із діапазоном значень 0 – 0xFFFFF. Визначені наступні правила, за якими відбувається процес вбудовування інформації у носії:

1. Для пакетів №0 – 9 (дані пакети передають вектор ініціалізації для дешифрування):
    1.1. Перші 8 біт (1 байт) використовуються для позначення порядкового номеру пакету. Це гарантує, що отримувач проведе процес обробки пакетів в правильному порядку.

1.2. Останні 12 біт (1.5 байт) використовуються для вбудовування вектора ініціалізації (IV). Вектор необхідний для дешифрування даних по протоколу ECDH.

Створення прихованого каналу у носії Flow Label, пакети 0-10

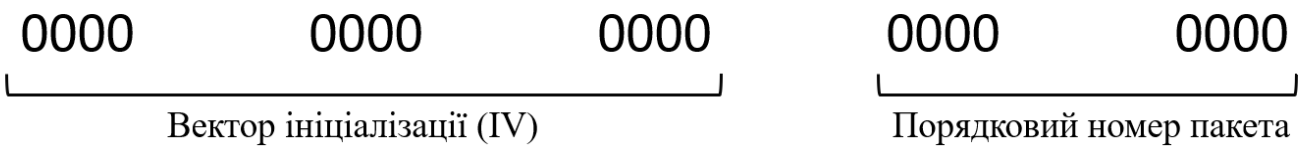

Рис. 2.5. Позначення заповнення стеганографічного носія (пакети №0-9)

2. Для пакету № 10 (вбудовування останнього байту вектору ініціалізації та початок вбудовування секретних даних):
   2.1. Перші 8 біт (1 байт) використовуються для позначення порядкового номеру пакету. Це гарантує, що отримувач проведе процес обробки пакетів в правильному порядку.
   2.2. Наступні 8 біт (1 байт) використовуються для вбудовування вектора ініціалізації (IV).
   2.3. Останні 4 біт використовуються для вбудовування зашифрованого повідомлення.

Створення прихованого каналу у носії Flow Label, пакет 11

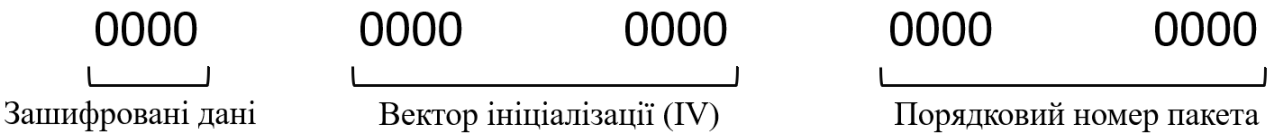

Рис. 2.6. Позначення заповнення стеганографічного носія (пакет №10)

3. Для пакетів № 11-N, N<256 (вбудовування секретних даних):
   3.1. Перші 8 біт (1 байт) використовуються для позначення порядкового номеру пакету. Це гарантує, що отримувач проведе процес обробки пакетів в правильному порядку.
   3.2. Останні 12 біт використовуються для вбудовування зашифрованого повідомлення.

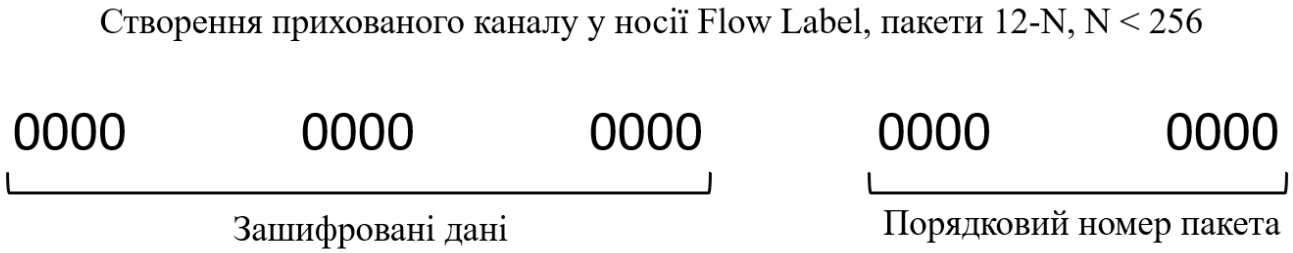

Рис. 2.7. Позначення заповнення стеганографічного носія (пакет №11-N, N<256)

Відповідно до даного підходу, пропускна здатність стеганографічного каналу становить 12 біт / пакет. Слід зазначити, що веткор ініціалізації (IV) завжди має довжину 16 байт і даний розмір не залежить від вхідних даних. Максимальна кількість даних, які можуть передаватися по прихованому каналу становить 384 байт, 16 з яких завжди становлять вектор ініціалізації, інші – користувацькі зашифровані дані.

Візуально третій етап моделі можна зобразити таким чином (рис. 2.8):

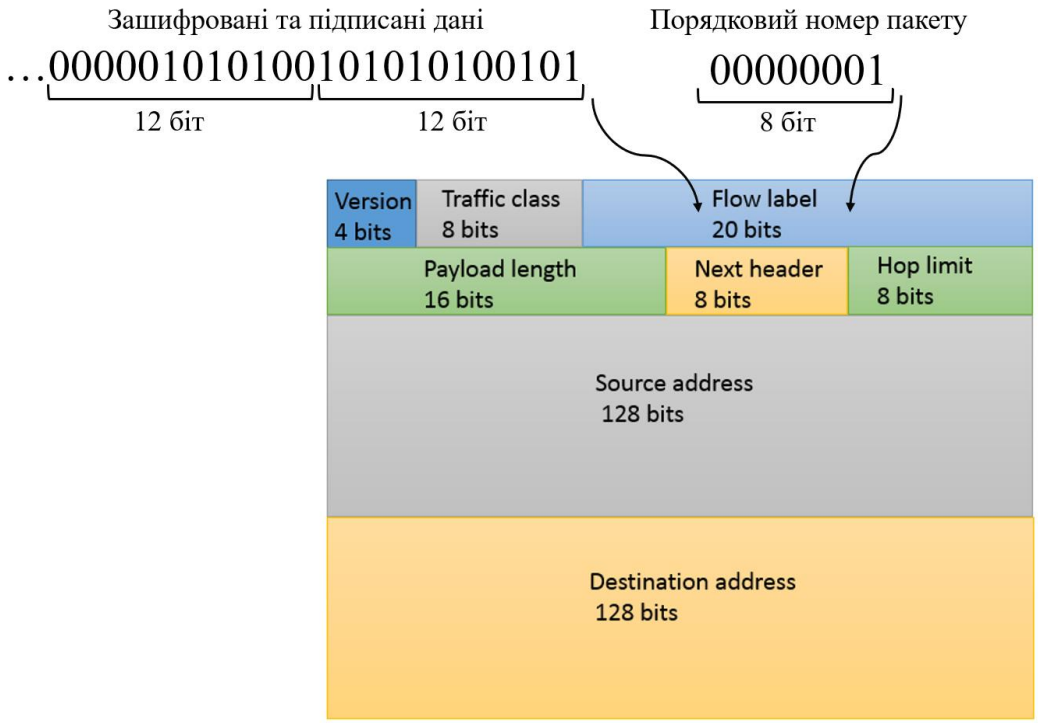

Рис. 2.8. Стеганографія. Вбудовування інформації у прихований канал

На четвертому етапі створені та модифіковані пакети відправляються по мережі до отримувача.

На п’ятому етапі відбувається отримання пакетів та всі значення із Flow Label заголовка пакету. Слід зазначити, що отримувач обробляє вхідні пакети за правилами, описаними на третьому етапі.

На шостому етапі відбувається процес валідації отриманих даних відповідно до алгоритму ECDSA. Якщо валідація не успішна – пакети відкидаються і подальший процес зупиняється. Якщо валідація успішна – процес переходить до наступного етапу дешифрування.

На останньому сьомому етапі відбувається процес дешифрування даних відповідно до протоколу Діффі-Хеллмана на еліптичних кривих (ECDH). Після успішного дешифрування даних, в результаті отримувач має початкове повідомлення.

Розроблена модель має значні переваги у порівнянні з існуючими реалізаціями (пункт 2.3), а саме:

1. Шифрування забезпечується одним із найбільш стійких та складних до злому асиметричних методів шифрування, а саме - протокол Діффі-Хеллмана на еліптичних кривих (ECDH) на противагу RSA та CBC-RC6 у інших моделях.
2. Забезпечується новітній метод цифрового підпису ECDSA для забезпечення цілісності даних та перевірки отримувачем на коректність та неушкодженість отриманих даних. Лише в одній з розглянутих вище моделей відбувалась перевірка цілісності даних (за допомогою MAC – коду автентифікації повідомлення).
3. Збільшена пропускна здатність стеганографічного каналу – до 12 біт / пакет. Це дозволить значно прискорити процес передачі даних до отримувача та зменшить кількість необхідних носіїв (пакетів).

2.5. Висновок до розділу

Проаналізовано існуючі підходи до побудови стеганографічних моделей із використанням поля Flow Label в якості прихованого каналу. Розглянуті моделі мають значні недоліки, серед яких в одній з моделей (розділ 2.2.1.) не відбувається перевірка валідації пакетів на стороні одержувача, також не реалізований підхід до впорядкування пакетів. В іншій моделі (розділ 2.2.2.) серед недоліків виділено використання симетричного шифрування та малу пропускну здатність каналу (8 біт /

пакет). Запропоновано модель, в якій використовується один із найбільш стійких асиметричних методів шифрування – протокол Діффі-Хеллмана на еліптичних кривих та алгоритм валідації – ECDSA. Запропонований підхід має пропускну здатність 12 біт / пакет та дозволяє забезпечити валідацію пакетів на стороні одержувача. За рахунок цього дана модель є стійкою до атаки MITM. Слід зазначити, що запропонований підхід дозволяє також забезпечити перевірку надходження пакетів в правильному порядку.

РОЗДІЛ 3
ПРОГРАМНА РЕАЛІЗАЦІЯ МОДЕЛІ ТА АНАЛІЗ РЕЗУЛЬТАТІВ

3.1. Технологія WPF

Для створення програмного продукту, який реалізує локально розроблену стаганографічну модель, було використано технологію WPF від Microsoft на базі .NET 5.

WPF, розшифровується як Windows Presentation Foundation, є платформою розробки і підсистемою .NET 5. WPF використовується для створення клієнтських додатків Windows, що працюють в операційній системі Windows. WPF використовує XAML як мову інтерфейсу та C# як мову програмування.

WPF був представлений як частина. NET Framework 3.0 в якості бібліотеки Windows для створення клієнтських додатків Windows і наступного покоління Windows Forms. Поточна версія WPF – 5.0.

WPF - це механізм, який відповідає за створення, відображення і управління користувацькими інтерфейсами, документами, зображеннями, фільмами і носіями в операційних системах Windows 7 і пізніших версій. WPF - це набір бібліотек, які мають всі функції, необхідні для створення, запуску, виконання та управління клієнтськими додатками Windows.

XAML - це нова описова мова програмування, розроблена Microsoft для написання користувацьких інтерфейсів для керованих додатків наступного покоління. XAML використовується для створення користувальницьких інтерфейсів для Windows і мобільних додатків, що використовують форми Windows Presentation Foundation (WPF), UWP і Xamarin.

Мета XAML проста - створювати користувацькі інтерфейси з використанням мови розмітки, яка виглядає як XML. XAML використовує формат XML для елементів і атрибутів. Кожен елемент в XAML представляє об'єкт, який є екземпляром типу. Область дії типу (класу, перерахування і т.д.) визначається простором імен, яке фізично знаходиться в збірці (DLL) бібліотеки. NET.

Хоча XAML використовується для створення користувацьких інтерфейсів у WPF, C# (9.0) використовується як мова програмування в WPF. Незважаючи на те, що вікна та їх елементи керування створюються в XAML під час розробки, вони також можуть бути створені під час виконання за допомогою мови C#.

C# також використовується для програмування всіх подій і бізнес-логіки. Всі дії, події та рендеринг виконуються в коді, що лежить в основі коду C#.

WPF підтримує два типи ресурсів: статичні і динамічні.

Статичні ресурси: статичний ресурс буде дозволений і призначений властивості (полю) під час завантаження XAML, яке відбувається до фактичного запуску програми. Він буде призначений тільки один раз, і будь-які зміни в словнику ресурсів ігноруються.

Динамічні ресурси: динамічний ресурс призначає об'єкту виразу властивість під час завантаження, але фактично не шукає ресурс до часу виконання, коли об'єкт виразу запитує значення. Це відкладає пошук ресурсу до тих пір, поки він не знадобиться під час виконання. Хорошим прикладом може служити пряме посилання на ресурс, визначений пізніше в XAML. Інший приклад - ресурс, який навіть не буде існувати до виконання. Він оновить ціль, якщо словник ресурсів джерела буде змінений.

3.2. Технологія WinPcap

Для фізичного створення та відправки пакетів між хостами використано технологію WinPcap.

WinPcap – це стандартний інструмент, що забезпечує доступ до з'єднань між рівнями мережі (підключення і вибір між двома хост-системами) в середовищах Windows. Він дозволяє захоплювати і передавати мережеві пакети, які обходять стек протоколів, включаючи фільтрацію пакетів на рівні ядра, механізм мережевої статистики і підтримку віддаленого захоплення пакетів.

WinPcap має драйвер, який розширює операційну систему для забезпечення низькорівневого доступу до мережі. Він також має бібліотеку, яка забезпечує легкий доступ до мережевих рівнів низького рівня. Ця бібліотека має версію популярного

UNIX libpcap API для Windows. Слід зазначити, що WinPcap є механізмом захоплення і фільтрації пакетів для багатьох інструментів з відкритим вихідним кодом і комерційних мереж. Цей інструмент має аналізатори протоколів, мережеві монітори, системи виявлення мережевих вторгнень, сніффери, генератори трафіку і тестери мережі. Деякі з цих інструментів, такі як Wireshark, Nmap або Snort, широко використовуються в управлінні мережею.

Відповідно для того, щоб використовувати дану технологію в середовищі .NET, було взято вихідних код бібліотеки Pcap.Net [13] та модифіковано для застосування у розробленій стеганографічній моделі. Pcap.Net – це .NET оболочка для WinPcap, написана на C++/CLI і C#, яка включає в себе майже всі функції WinPcap і включає в себе структуру інтерпретації пакетів [13].

3.3. Архітектура програмної системи

На рис. 3.1 зображено загальну архітектуру стеганографічної моделі:

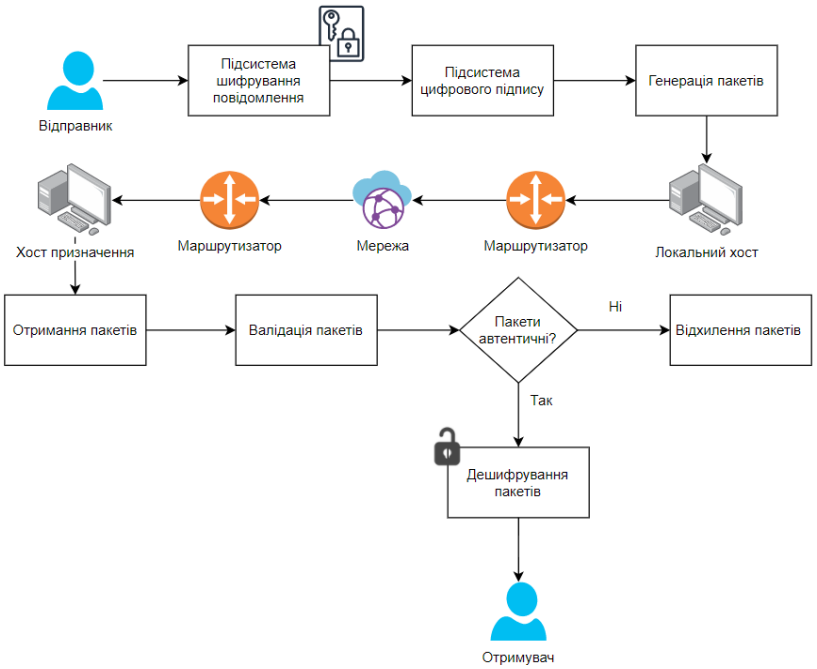

Рис. 3.1. Загальна архітектура стеганографічної моделі

В рамках дипломної роботи розроблено 2 настільних додатки:

1. Програма для відправки пакетів (сторона відправника).
2. Програма для отримання пакетів (сторона отримувача).

Слід зазначити, що для імітації сторони отримувача, на локальний хост було підключену розроблену програму - сніффер для отримання всіх відправлених пакетів.

Аналізатор пакетів, або «сніффер» пакетів, - це програмне або апаратне забезпечення, яке може перехоплювати і відстежувати пакети в міру їх проходження по мережі. Таким чином, IT-фахівці та кіберзлочинці можуть ефективно перевіряти вміст файлів і повідомлень, що передаються в мережу, з мережі або всередині мережі.

Сніффери пакетів можна використовувати в двох режимах: фільтрований і нефільтрований. Фільтроване зчитування пакетів означає, що аналізатори будуть шукати певні дані і будуть захоплювати або копіювати тільки ті пакети, які містять ці дані. Зчитування нефільтрованих пакетів означає, що всі пакети захоплюються та/або копіюються, незалежно від даних, що містяться в них. Вони також можуть збирати широкий спектр інформації, в тому числі про те, які веб-сайти відвідує конкретний користувач, що він переглядає, місця призначення і вміст будь-яких електронних листів або повідомлень, які вони відправляють, а також будь-які файли, які вони завантажують.

Сніффер було розроблено як консольний додаток на платформі .NET 5 з використанням бібліотеки Pcap.Net та технології WinPcap. Дане програмне забезпечення має здатність прослуховувати всі мережеві інтерфейси, які існують на локальній машині.

Для прикладу було обрано віртуальний мережевий адаптер vEthernet (Default switch) через який пакети відправлялись на шлюз за замовчуванням. Сніффер прослуховував та отримував пакети, які проходили по шляху vEthernet (Default switch) – Default Gateway.

На рис. 3.2 зображена фактично реалізована архітектура моделі:

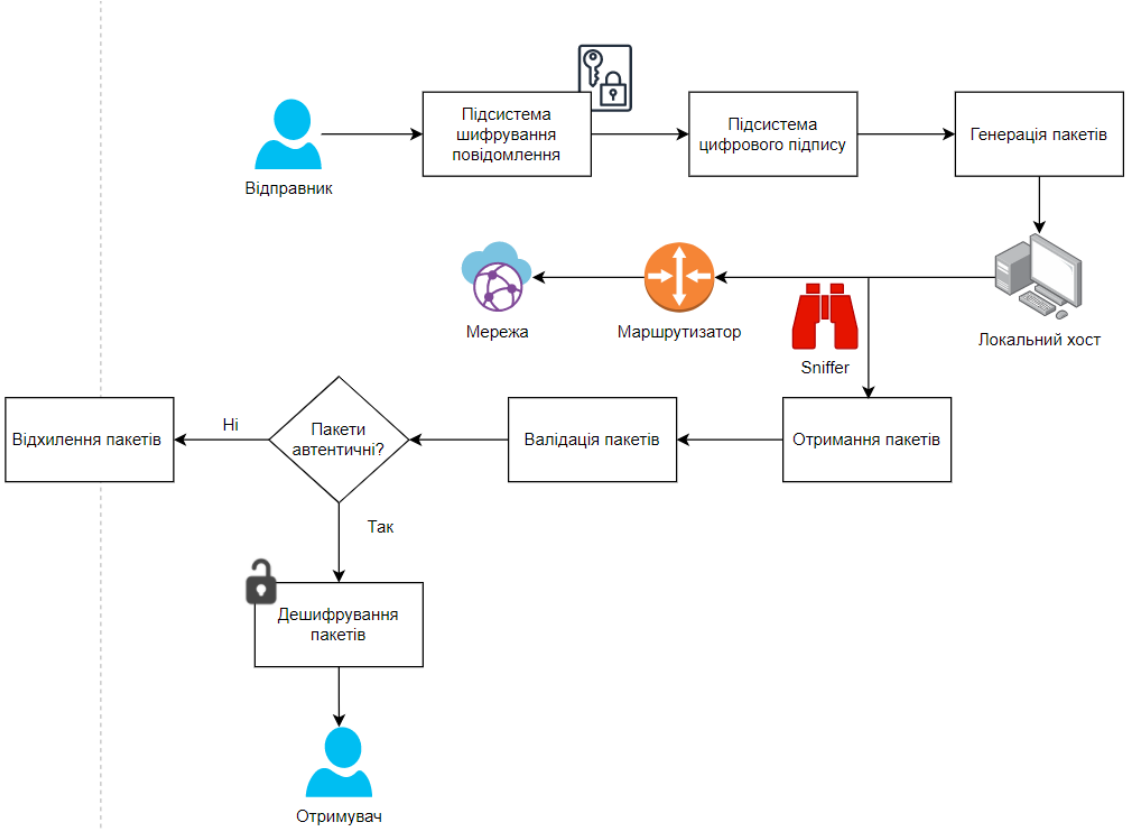

Рис. 3.2. Реалізована архітектура стеганографічної моделі

Загалом було реалізовано такі програмні компоненти:

1. Підсистема для шифрування та дешифрування по протоколу EDCH.
2. Підсистема для створення цифрового підпису та його валідацію за алгоритмом ECDSA.
3. WPF-додаток в ролі відправника пакетів.
4. WPF-додаток в ролі отримувача пакетів (разом із сніффером).

3.4. Опис програмних компонентів

Підсистема для шифрування та дешифрування по протоколу EDCH. Для реалізації даної підсистеми було використано клас ECDiffieHellmanCng у бібліотеці System.Security.Cryptography.Cng.dll (провайдер - Microsoft). Дана бібліотека забезпечує криптографічну реалізацію наступного покоління (CNG) алгоритму Діффі-Хеллмана з еліптичною кривою (ECDH). Цей клас використовується для виконання криптографічних операцій. Використовуючи за основу реалізацію компанії Microsoft була створена власна бібліотека (dll-файл), яка отримувала на

вході секретне повідомлення та на виході віддавала зашифроване повідомлення разом із вектором ініціалізації.

Підсистема для створення цифрового підпису та його валідацію за алгоритмом ECDSA. Для реалізації даної підсистеми було використано клас ECDsaCng у бібліотеці System.Security.Cryptography.Cng.dll (провайдер - Microsoft). Дана бібліотека забезпечує криптографічну реалізацію наступного покоління (CNG) алгоритму цифрового підпису еліптичної кривої (ECDSA). Використовуючи за основу реалізацію компанії Microsoft була створена власна бібліотека (dll-файл), яка отримувала на вході зашифроване повідомлення разом із вектором ініціалізації та на виході віддавала цифровий підпис (масив байтів).

WPF-додаток в ролі відправника пакетів. Даний додаток має графічний інтерфейс (Додаток А), за допомогою якого користувач обирає мережевий адаптер відправника, вводить секретне повідомлення та запускає процес генерації та відправки пакетів. Користувач поетапно бачить виконання кожного процесу, відповідні повідомлення виводяться в консолі даного додатку (спеціально відведене місце для виводу текстової інформації).

WPF-додаток в ролі отримувача пакетів. Даний додаток має графічний інтерфейс (Додаток Б), за допомогою якого користувач обирає мережевий адаптер для зчитування пакетів (запускає сніффер), та очікує на надходження та обробку пакетів, отриманих від відправника. Користувач поетапно бачить виконання кожного процесу, відповідні повідомлення виводяться в консолі даного додатку (спеціально відведене місце для виводу текстової інформації).

3.5. Аналіз отриманих результатів

Було проведено дослідження етапів шифрування, застосування цифрового підпису та валідації цифрового підпису, дешифрування. Проведено порівняння за швидкістю виконання даних етапів із попередньою реалізацією даної моделі (Розділ 2.2.1). Дослідження проводились на комп’ютері із такими технічними характеристиками: Intel Core i7-8565U, 16 GB RAM, 1 TB SSD. Для кожного з тест

кейсів було отримано середнє значення із вибірки 100 тестувань. Отримані такі результати:

Таблиця 3.1.

Порівняння швидкості виконання окремих підсистем на стороні відправника.

		Запропонована модель			Процент зміни продуктивності (+%)	Попередня реалізація		
№	Розмір вхідних даних (байт)	Протокол ECDH (с)	Алгоритм ECDSA (с)	Разом (с)		Хаотичний метод кодування (с)	RSA-шифрування (с)	Разом (с)
1	1	0.1538	0.4593	0.6131	55.4	0.7158	0.658	1.3738
2	10	0.2981	0.4869	0.7850	46.3	0.8113	0.6512	1.4625
3	20	0.3471	0.5112	0.8583	46.4	0.9240	0.6785	1.6025
4	30	0.4119	0.5396	0.9515	41.0	0.9512	0.6621	1.6133
5	40	0.4378	0.5503	0.9881	40.6	0.9921	0.6703	1.6624
6	50	0.5193	0.5784	1.0977	45.0	1.2036	0.7919	1.9955
7	100	0.6875	0.6141	1.3016	46.9	1.6321	0.8192	2.4513
8	150	0.7102	0.6399	1.3501	54.9	2.1943	0.7998	2.9941
9	200	0.8583	0.6521	1.5104	54.0	2.3957	0.8894	3.2851
10	250	0.9343	0.6895	1.6238	57.5	2.8974	0.9215	3.8189

Графічні дані порівняння продуктивності роботи двох підходів до побудови моделі (сторона відправника) представлено на рис. 3.3.

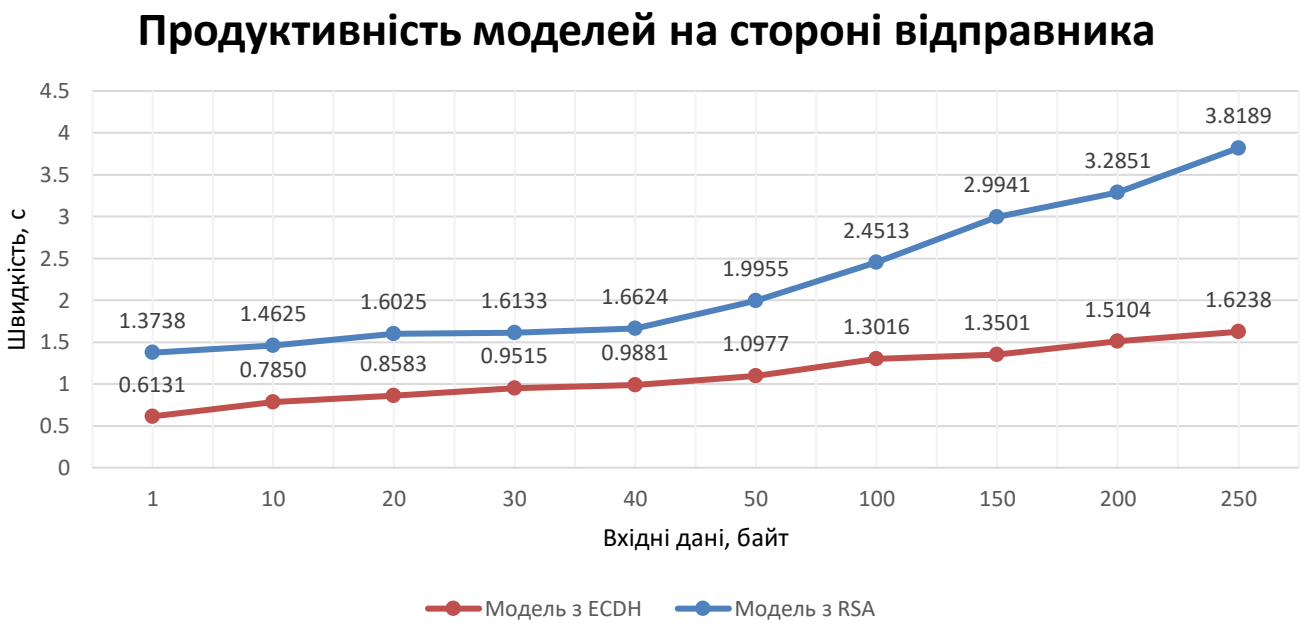

Рис. 3.3. Графічне відображення порівняння моделей на стороні відправника

Проаналізувавши отримані дані, можна зробити висновок, що на етапі відправника, запропонована модель працює в середньому на 48.8% швидше за попередню реалізацію.

Таблиця 3.2.

Порівняння швидкості виконання окремих підсистем на стороні отримувача.

		Запропонована модель			Процент зміни продуктивності (+%)	Попередня реалізація		
№	Розмір вхідних даних (байт)	Протокол ECDH (с)	Валідація ECDSA (с)	Разом (с)		Хаотичний метод декодування (с)	RSA-дешифрування (с)	Разом (с)
1	1	0.1621	0.4587	0.6208	46.1	0.5699	0.5824	1.1523
2	10	0.2348	0.4694	0.7042	42.7	0.6325	0.5963	1.2288
3	20	0.2864	0.5296	0.816	37.1	0.6491	0.6487	1.2978
4	30	0.3159	0.5347	0.8506	35.6	0.6692	0.6509	1.3201
5	40	0.3328	0.5546	0.8874	36.5	0.7249	0.6716	1.3965
6	50	0.3622	0.5617	0.9239	41.7	0.8794	0.7058	1.5852
7	100	0.4593	0.6079	1.0672	41.5	1.0873	0.7356	1.8229
8	150	0.4967	0.6125	1.1092	41.9	1.1587	0.7492	1.9079
9	200	0.5388	0.6347	1.1735	43.7	1.3187	0.7654	2.0841
10	250	0.6259	0.6796	1.3055	44.3	1.5497	0.7931	2.3428

Графічні дані порівняння продуктивності роботи двох підходів до побудови моделі (сторона отримувача) представлено на рис. 3.4.

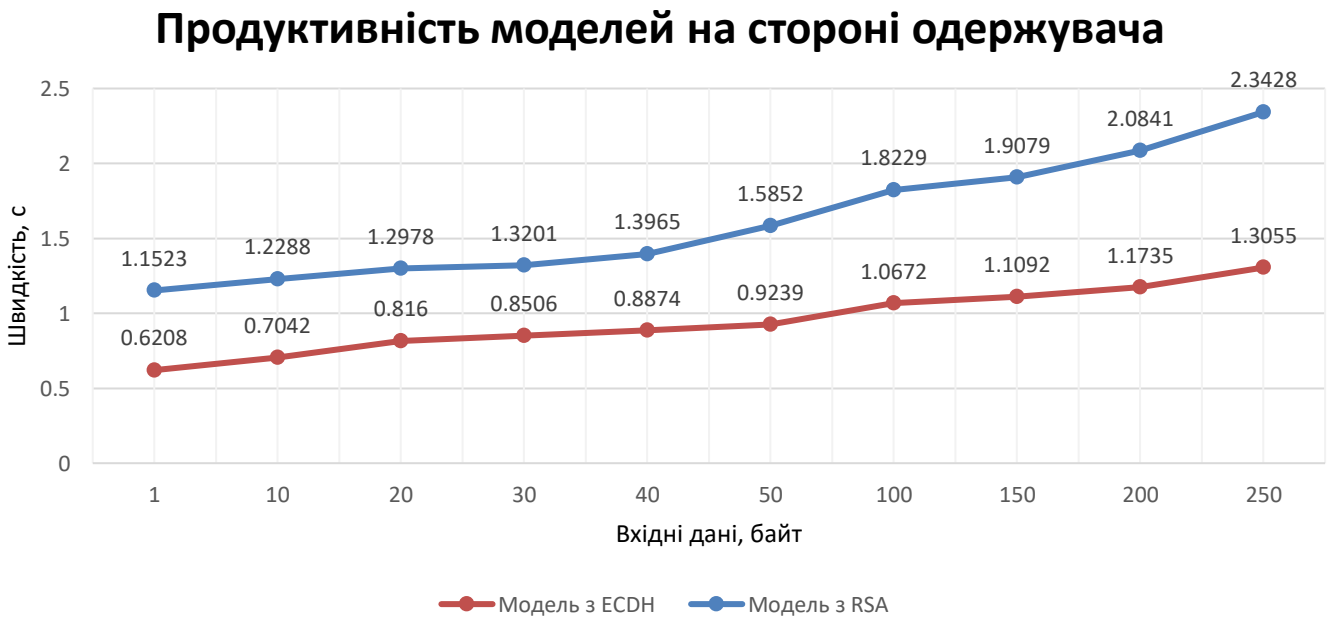

Рис. 3.4. Графічне відображення порівняння моделей на стороні отримувача

Проаналізувавши отримані дані, можна зробити висновок, що на етапі відправника, запропонована модель працює в середньому на 41.1% швидше за попередню реалізацію.

3.6. Висновок до розділу

Для створення програмного продукту, який реалізує локально розроблену стаганографічну модель, було використано технологію WPF від Microsoft на базі .NET 5. Для фізичного створення та відправки пакетів між хостами використано технологію WinPcap. Для імітації сторони отримувача, на локальний хост було підключену розроблену програму - сніффер для отримання всіх відправлених пакетів. Аналізатор пакетів, або «сніффер» пакетів, - це програмне або апаратне забезпечення, яке може перехоплювати і відстежувати пакети в міру їх проходження по мережі. Було проведено дослідження етапів шифрування, застосування цифрового підпису та валідації цифрового підпису, дешифрування. Встановлено значний приріст у продуктивності роботи моделі при використанні розробленого підходу.

ВИСНОВКИ

Проаналізовано існуючи підходи до побудови стеганографічної моделі передачі даних по прихованому каналу в протоколі IPv6 та зроблено висновок, що серед запропонованих не існує жодного, який би пропонував одночасно обробку даних стійкими алгоритмами та перевірку цілісності пакетів на стороні отримувача.

Розроблено WPF-додатки (для операційної системи Windows) на платформі .NET 5 для демонстрації роботи стеганографічної моделі з використанням мови програмування C# 9.0 та мови розмітки XAML.

Встановлено, що запропонований підхід значно підвищує надійність та стійкість моделі за рахунок використанням протоколу Діффі-Хеллмана на еліптичних кривих (ECDH) для шифрування та методу цифрового підпису ECDSA. В запропонованый моделі забезпечується перевірка цілісності та неушкодженості пакетів на стороні отримувача. Лише в одній з попередньо розглянутих моделей відбувалась перевірка цілісності даних (за допомогою MAC – коду автентифікації повідомлення).

Збільшено пропускну здатність стеганографічного каналу – до 12 біт / пакет. Це дозволило значно прискорити процес передачі даних до отримувача та зменшити кількість необхідних носіїв (пакетів).

Виявлено, що використання протоколу ECDH разом з алгоритмом ECDSA значно підвищують продуктивність системи у порівнянні з попередньою існуючою реалізацією моделі з використання хаотичного методу п’ятого порядку та асиметричного алгоритму шифрування RSA. На стороні відправника швидкість обробки пакетів в середньому збільшилась на 48.8%, в той час на стороні отримувача – на 41.1% відповідно. Використання даної моделі підвищує безпеку передачі даних по прихованому каналу та надає можливість отримувачу валідувати пакети. Таким чином дана модель може бути використана для забезпечення безпечного прихованого зв’язку у реальних мережевих системах.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ra’ad A. Muhajjar, Farah A. Badr, «Secure Data Communications using Cryptography and IPv6 Steganography», публікація в International Journal of Engineering & Technology, 2018. – 624 - 628 с.
2. Sandip Bobade, Rajeshawari Goudar, «Secure Data Communication using Protocol Steganography in IPv6», публікація в International Journal of Engineering & Advanced Technology, 2014. – 104 - 109 с.
3. Wojciech Mazurczyk, Krystian Powójski, and Luca Caviglione, «IPv6 Covert Channels in the Wild», CECC ’19, Nov. 14–15, 2019, Munich, DE.
4. Song Fei, Cui Zhe, «A Method for Low-overhead Secure Network Coding», публікація в Appl. Math. Inf. Sci. 7, No. 5, 1699-1703 (2013).
5. R. K. Yadav and M. Kushwaha, "Message Hiding Using Steganography and Cryptography," 2018.
6. E. Cauich, R. G. Cárdenas, and R. Watanabe, "Data hiding in identification and offset IP fields," публікація в International Symposium and School on Advancex Distributed Systems, 2005, pp. 118-125.
7. Lazaros Moysis, Aleksandra Tutueva, Christos Volos, Denis Butusov, Jesus M. Munoz-Pacheco, Hector Nistazakis, «A Two-Parameter Modified Logistic Map and Its Application to Random Bit Generation», публікація в Symmetry, 2020.
8. Steven Gianvecchio, Haining Wang «Detecting Covert Timing Channels: An Entropy-Based Approach», публікація в CCS’07, October 29–November 2, 2007, Alexandria, Virginia, USA.
9. N. Singh, J. Bhardwaj, and G. Raghav, «Network Steganography and its Techniques: A Survey» vol. 174, 2017.
10. S. Zander, G. Armitage, and P. Branch, "A survey of covert channels and countermeasures in computer network protocols," публікація в IEEE Communications Surveys & Tutorials, vol. 9, pp. 44-57, 2007.
11. Szczypiorski K., «Steganography in TCP/IP Networks.State of the Artand a Proposal of a New System – HICCUPS, In Institute of Telecommunications' seminar,

Warsaw University of Technology, Poland, November, 2003 http://krzysiek.tele.pw.edu.pl/pdf/steg-seminar 2003.pdf

12. D. D. DhobaJe, V. R. Ghorpade B. S. Patjj, S. B. Patil, «Steganography by hiding data in TCP/IP headers", публікація в 3rd International Conference on Advanced Computer Theory and Engineering (ICACTE), 2010.

13. https://en.wikipedia.org/wiki/Elliptic-curve_Diffie%E2%80%93Hellman

14. https://docs.microsoft.com/en-us/dotnet/api/system.security.cryptography.ecdiffiehellmancng?view=net-5.0

15. https://sectigostore.com/blog/ecdsa-vs-rsa-everything-you-need-to-know/

16. https://en.wikipedia.org/wiki/Steganography

17. https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation

18. https://datatracker.ietf.org/doc/html/rfc6437

19. https://docs.oracle.com/cd/E19683-01/817-0573/chapter1-26/index.html

20. https://networkengineering.stackexchange.com/questions/28723/usage-of-flow-label-in-ipv6-header

Додаток А

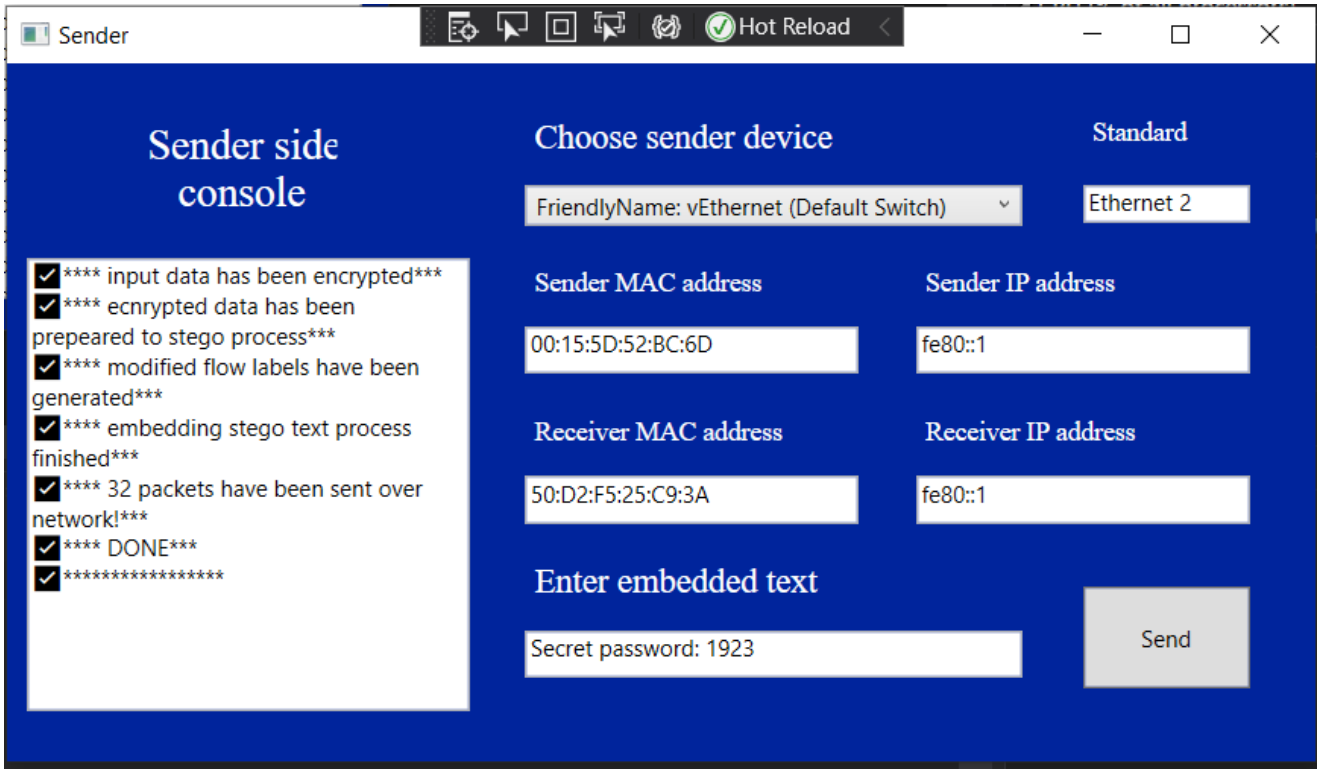

Рис. Д.1 Зображення інтерфейсу сторони відправника

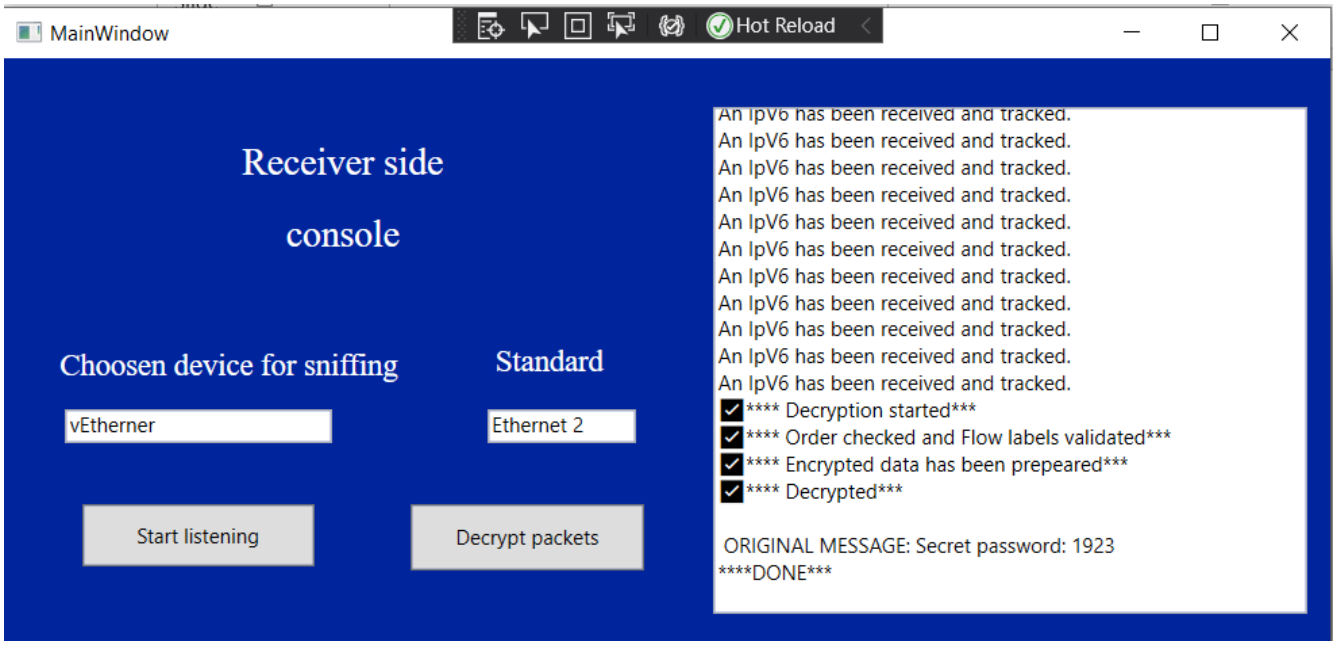

Рис. Д.2 Зображення інтерфейсу сторони отримувача

Додаток Б

Документ архітектури програмного забезпечення (Software Architecture Document)

Розроблення та програмна реалізація модифікованої стеганографічної моделі передачі даних в протоколі IPv6

Максим Івасенко, Ольга Супрун
Версія 1.5

Київ 2021

Історія змін

Примітка: Цикл історії змін починається після завершення початкової версії документа архітектури програмного забезпечення.

Дата	Версія	Опис	Автор
2.04.2021	1.0	Створення бібліотеки з використанням протоколу ECDH.	Максим Івасенко
18.04.2021	1.1	Створення бібліотеки з використанням алгоритму ECDSA.	Максим Івасенко
5.05.2021	1.2	Розробка WPF-додатку для формування пакетів.	Максим Івасенко
11.05.2021	1.3	Підключення розроблених раніше бібліотек до WFP-додатку.	Максим Івасенко
16.05.2021	1.4	Розробка WPF-додатку для зчитування пакетів (сніффер).	Максим Івасенко
21.05.2021	1.5	Виправлення багів та проблем в розроблених додатках. Перевірка системи.	Максим Івасенко

Зміст

Документ архітектури програмного забезпечення

1. Вступ

Даний документ містить огляд розробленого програмного забезпечення і пояснює всю архітектуру інструменту специфікації процесів (PST).

Документ містить високорівневий опис цілей архітектури, варіантів використання, підтримуваних системою, а також архітектурних стилів і компонентів, які були обрані під час розробки програмного забезпечення. Дана структура дозволяє розробити критерії проектування і документи, які детально визначають технічні стандарти і стандарти предметної області.

1.1. Мета

Документ з архітектури програмного забезпечення (SAD) містить повний архітектурний огляд розробленого додатку. У ньому представлений ряд різних архітектурних підходів, що зображують різні аспекти системи. Документ призначений для документування і передачі важливих архітектурних рішень, які були прийняті під час розробки системі.

Для того, щоб максимально точно описати програмне забезпечення, структура документа заснована на поданні архітектурної моделі "4+1" [KRU41] (рис. 1.1).

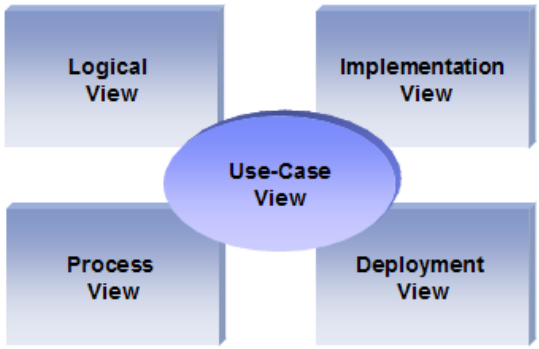

Рис. 1.1. Модель «4+1»

Модель подання "4+1" дозволяє різним зацікавленим сторонам знайти те, що їм потрібно в архітектурі програмного забезпечення.

1.2. Сфера застосування

Обсяг даної роботи полягає в тому, щоб описати архітектуру додатку для демонстрації працездатності стеганографічної моделі за допомогою передачі пакетів по мережі.

Даний документ описує аспекти проектування інструменту специфікації процесу (PST), які вважаються архітектурно значущими; тобто ті елементи і поведінки системи, які є найбільш фундаментальними для розуміння проекту в цілому. Зацікавленим сторонам, яким потрібне технічне розуміння інструменту специфікації процесу, рекомендується почати з читання даного документа, потім переглянути UML-моделі інструменту специфікації процесу, а потім переглянути вихідний код.

1.3. Визначення, абревіатури та скорочення

.NET - платформа Microsoft

IP – інтернет протокол

C# - мова програмування від Microsoft

SAD - документ з архітектури програмного забезпечення

UML - уніфікована мова моделювання

Відправник - це будь-який користувач, який відправляє пакети

Отримувач – це будь-який користувач, який приймає пакети

2. Архітектурне представлення

У цьому документі детально описана архітектура з використанням різних представлень, визначених у моделі "4+1" [KRU41]:

1. Перегляд варіантів використання (Use Case view)

Аудиторія: всі зацікавлені сторони системи, включаючи кінцевих користувачів.

Область: описує набір сценаріїв та / або варіантів використання, які представляють деякі важливі, центральні функціональні можливості системи. Описує учасників і варіанти використання системи. Дане представлення представляє потреби користувача і додатково розробляється на рівні проектування для більш докладного опису дискретних потоків і обмежень.

2. Логічне представлення (Logical view)

Аудиторія: Дизайнери, розробники.

Область - функціональні вимоги: описує об'єктну модель проекту. Також описані найбільш важливі реалізації варіантів використання і бізнес-вимоги системи.

3. Представлення процесу (Process view)

Аудиторія: Інтегратори.

Область - нефункціональні вимоги: описує аспекти процесів та синхронізації проекту.

4. Представлення декомпозиції модуля (Module Decomposition View)

Аудиторія: Програмісти.

Область – програмні компоненти: описує модулі та підсистеми програми.

5. Представлення даних (Data view)

Аудиторія: фахівці з даних, адміністратори баз даних.

Область: описує архітектурно значущі елементи в моделі даних.

6. Представлення розгортання (Deployment view)

Аудиторія: менеджери з розгортання.

Область - топологія: описує відображення програмного забезпечення на апаратне забезпечення і показує розподілені аспекти системи.

3. Архітектурні цілі та обмеження

Серверна сторона

Даний додаток працює на локальній машині користувача та не потребує додаткових зовнішніх ресурсів.

Клієнтська сторона

Користувачі зможуть отримати доступ до додатку, відправляти дані та отримувати їх.

3.1. Безпека

Права читача будуть надані будь-якому користувачеві, який отримав доступ до цільової сторінки програми.

Додаткові правила безпеки не потрібні.

3.2. Збереження даних

Додаток не зберігає жодну інформацію у системах бази даних. Всі процеси відбуваються лише in-memory (під час роботи додатку).

3.3. Продуктивність

Немає ніяких особливих обмежень, пов'язаних з продуктивністю системи.

Очікується, що система повинна відповідати на будь-який запит відповідно до стандартних тайм-аутів сценаріїв роботи системи, також продуктивність системи може залежати від доступного обладнання, мережі блоку живлення і можливостей

підключення до Інтернету. Крім того, час обробки пакетів може залежати від розміру даних, який, у свою чергу, залежить від введення даних Користувачем. Таким чином, фактична продуктивність може бути визначена тільки після розгортання і тестування системи.

4. Перегляд варіантів використання (Use-Case View)

Це список варіантів використання, які представляють основні функціональні можливості кінцевої системи [SRS] (рис 4.1):

1. Підключення до мережевого інтерфейсу
2. Відправка повідомлення
3. Отримання повідомлення

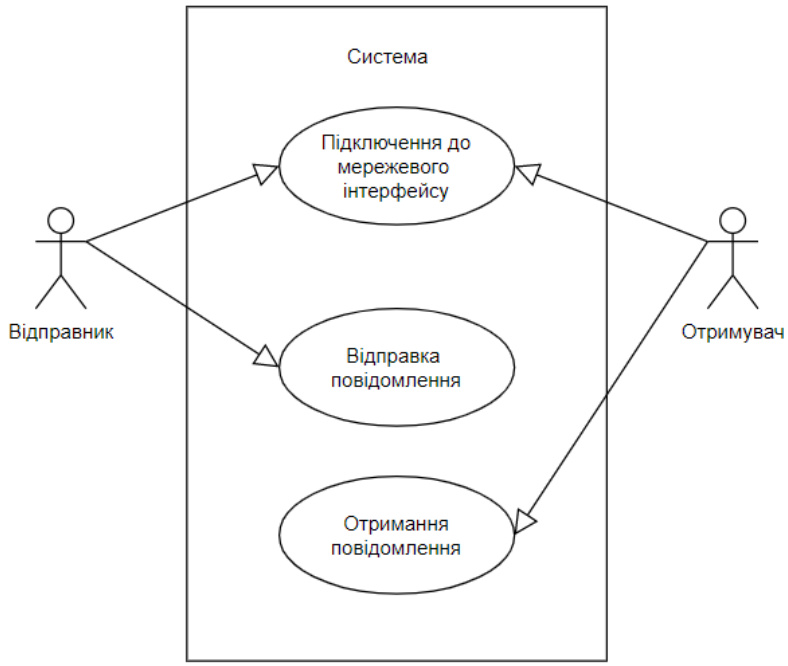

Рис. 4.1. Use-Case діаграма

4.1. Дійові особи

Як описано на діаграмі варіантів використання, користувач може бути одним з двох типів акторів:

1. Відправник: має права на підключення до мережевого інтерфейсу та відправку даних.

2. Отримувач: має права на підключення до мережевого інтерфейсу та отримання даних.

4.2. Реалізація варіантів використання

На наведеній нижче діаграмі функціональних можливостей (рис. 4.2) варіантів використання описується, як елементи системи забезпечують функціональні можливості, визначені у варіантах використання. Варіанти використання відображаються як функціональні можливості системи. Функціональність може включати в себе кілька варіантів використання.

Рис. 4.2. Реалізація варіантів використання

5. Логічне Представлення

5.1. Огляд

Даний інструмент розділений на шари на основі N-рівневої архітектури [KRU41] (рис. 5.1).

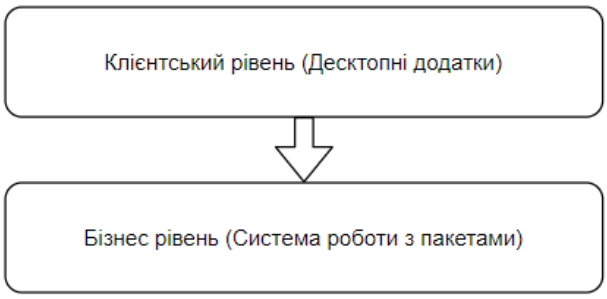

Рис. 5.1. Розподіл системи на шари

Дана стратегія була обрана тому, що вона ізолює різні системні обов'язки один від одного, щоб поліпшити як розробку, так і обслуговування системи.

6. Представлення процесу

На рис. 6.1 зображено представлення процесів, які відбуваються у системі. Серед зазначених процесів можна виділити:

1. Процес шифрування. Відправник надсилає в систему роботи з пакетами вхідне повідомлення та на виході отримує зашифровані дані разом з вектором ініціалізації. Даний процес також включає етап цифрового підпису даних.
2. Процес дешифрування. Отримувач надсилає в систему роботи з пакетами отримані дані разом з цифровим підписом та вектором ініціалізації та на виході отримує розшифроване повідомлення.
3. Процес відправки Відправник – Мережа та Мережа – Отримувач. Дані процеси означають процес передачі пакетів по мережі.

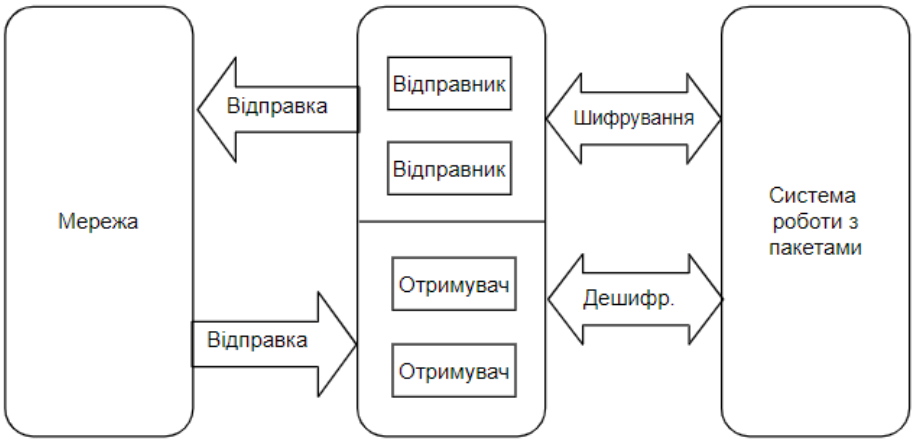

Рис. 6.1. Представлення процесів

7. Представлення декомпозиції модуля

Представлення декомпозиції модуля засноване на принципах поділу частин системи та абстракції та підтримує цілі модифікації системи та зручності її використання.

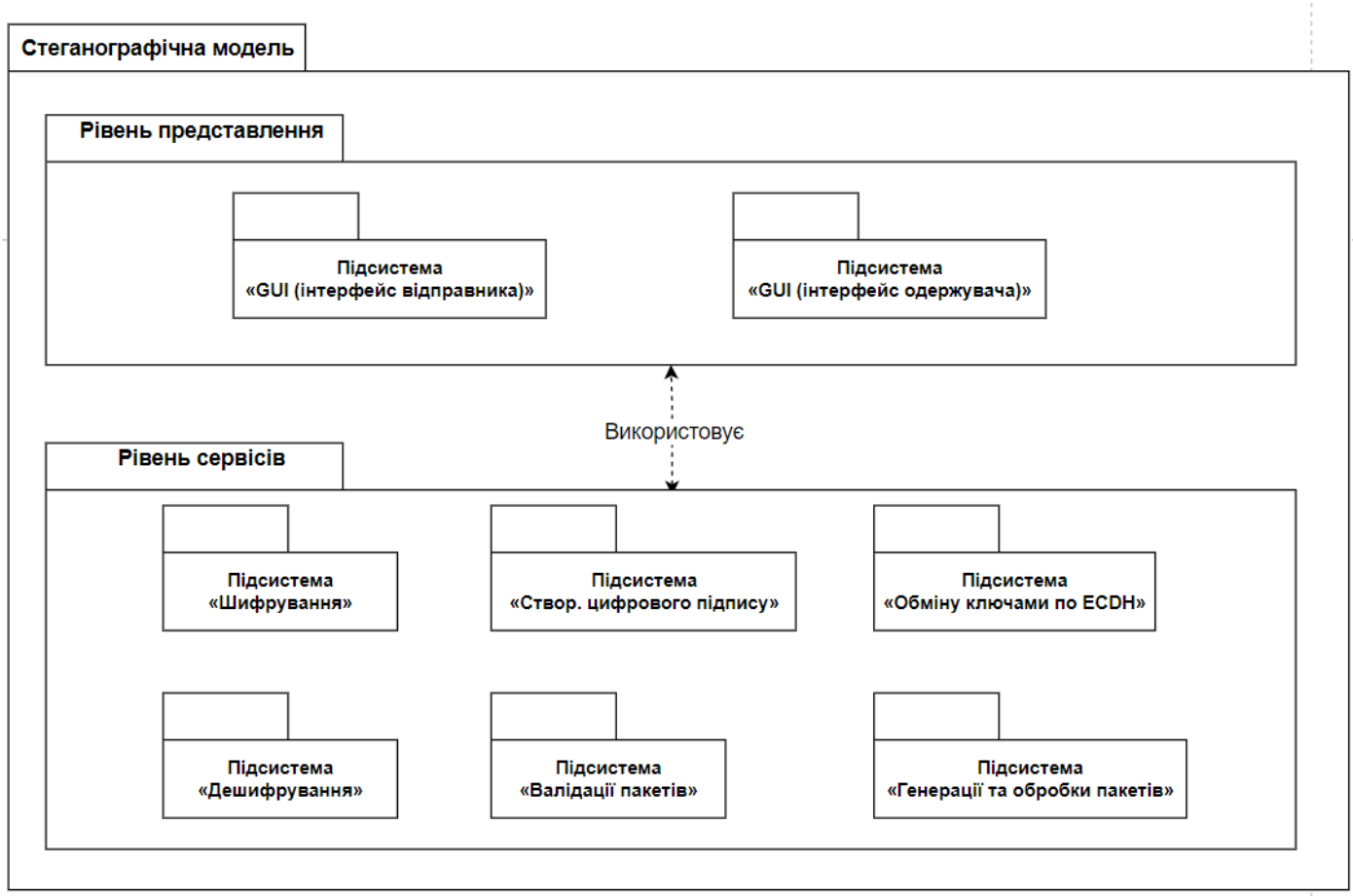

Рис. 7.1. Представлення декомпозиції модуля

Рівень представлення містить 2 підсистеми: графічний інтерфейс одержувача та графічний інтерфейс відправника.

Рівень сервісів містить 6 підсистем: підсистема шифрування по протоколу ECDH, підсистема дешифрування по протоколу ECDH, підсистема створення цифрового підпису по алгоритму ECDSA, підсистема валідації цифрового підпису та пакетів по алгоритму ECDSA, підсистема локального обміну ключами, підсистема генерації, отримання та обробки пакетів.

8. Перегляд даних

Даний додаток не потребує створення спеціальних умов для збереження, відтворення та обробки даних, оскільки всі дані зберігаються лише in-memory, тобто під час роботи додатку.

9. Представлення розгортання

Розгортання даного додатку здійснювалось з використання одного хосту та маршрутизатора з виходом в мережу Інтернет (рис. 8.1).

Рис. 8.1. Графічне представлення фізичних компонентів системи

10. Розмір і продуктивність

Умови до використання:

На одному хосту кількість можливих користувачів (як відправників, так і отримувачів) залежить від параметрів даного хоста та кількості доступних мережевих інтерфейсів.

Продуктивність роботи додатку залежить від кількості введеної допустимої інформації (відповідно і к-ть згенерованих пакетів-носіїв для передачі даної інформації відповідно).