Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра обчислювальної техніки

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Методи виявлення передавання прихованої інформації в службових заголовках протокольних блоків даних комп'ютерних мереж»

Виконав: студент 2-го курсу, групи 1КІ-20м
спеціальності 123 — Комп’ютерна інженерія
_____ Сирцов С. Р.
(прізвище та ініціали)

Керівник: к.т.н., проф. каф. ОТ
_____ Захарченко С. М.
(прізвище та ініціали)
«____» _____ 2021 р.

Опонент: к.т.н., професор каф. ЗІ
_____ Кондратенко Н.Р.
(прізвище та ініціали)
«____» _____ 2021 р.

Допущено до захисту
Завідувач кафедри ОТ
_____ д.т.н., проф. Азаров О.Д.
(прізвище та ініціали)
«____» _____ 2021 р

Вінниця 2021

АНОТАЦІЯ

УДК 004.4

Сирцов С. Р. Методи виявлення передавання прихованої інформації в службових заголовках протокольних блоків даних комп'ютерних мереж. Магістерська кваліфікаційна робота зі спеціальності 123 — комп'ютерна інженерія, освітня програма — комп'ютерна інженерія. Вінниця: ВНТУ, 2021. 116 с.

В магістерській дипломній роботі було досліджено та проаналізовано можливі методи приховування даних в службових заголовках протокольних блоків даних та розроблено методи для їх виявлення. У теоретичній частині розглянуто основні технології та інструменти, які дозволяють вбудовувати приховані повідомлення в пакети комп’ютерної мережі. Розроблено методи виявлення та протидії пакетів з прихованою інформацію, що потрапляють у внутрішню локальну мережу. Розроблено програмну систему для виявлення підозрілих пакетів, що можуть бути модифіковані шляхом зміни полів службових заголовків протокольних блоків даних. Було проведено тестування та розроблено інструкцію користувача.

Програма виявлення прихованих повідомлень розроблена для використання в операційній системі Windows на платформі .NET Framework за допомогою середовища розробки Visual Studio та мов програмування C# та C++.

Ключові слова: стеганографія, стеганоконтейнер, приховані повідомлення, службові заголовки, мережеві пакети, перехоплення пакетів, сніфер, ідентифікатор пакету, тести на псевдовипадковість.

ABSTRACT

UDC 004.4

Syrtsov S. R. Methods for detecting the transmission of hidden information in the service headers of the protocol data blocks of computer networks. Master's thesis in the specialty 123 — computer engineering, educational program — computer engineering. Vinnytsia: VNTU, 2021. 116 p.

In the master's thesis, possible methods of hiding data in the service headers of protocol data blocks were investigated and analyzed and methods for their detection were developed. The theoretical part discusses the basic technologies and tools that allow you to embed hidden messages in computer network packets. Methods for detecting and counteracting packets with hidden information that enter the internal local area network have been developed. A software system has been developed to detect suspicious packets that can be modified by changing the service header fields of the protocol data blocks. Testing and user manual were developed.

The hidden message detection program is designed for use in the Windows operating system on the .NET Framework using the Visual Studio development environment and the C # and C ++ programming languages.

Keywords: steganography, steganocontainer, hidden messages, service headers, network packets, packet interception, sniffer, packet identifier, pseudo-randomness tests.

Вінницький національний технічний університет

Факультет інформаційних технологій та комп’ютерної інженерії

Кафедра обчислювальної техніки

Освітньо-кваліфікаційний рівень — магістр

Спеціальність 123 — «Комп’ютерна інженерія»

ЗАТВЕРДЖУЮ
Завідувач кафедри ОТ
Азаров О. Д.
«____» _____2021 р

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

Сирцову Сергію Романовичу
(прізвище, ім’я, по-батькові)

1 Тема роботи: «Методи виявлення передавання прихованої інформації в службових заголовках протокольних блоків даних комп'ютерних мереж»

Керівник роботи: к.т.н., професор кафедри ОТ Захарченко Сергій Михайлович затверджені наказом Вінницького національного технічного університету від 24.09.2021 року № 277.

2 Строк подання студентом роботи 10.12.2021

3 Вихідні дані до роботи: проаналізувати основні технології захисту інформації в комп’ютерних мережах, існуючі методи приховування інформації, розробити механізм виявлення прихованих даних в службових заголовках протокольних блоків даних.

4 Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити): огляд та аналіз технологій прихованої передачі інформації, аналіз методів прихованої передачі інформації в комп’ютерних мережах, порівняння мережевих стеганографічних методів, розробка методів виявлення прихованих даних, програмна реалізація обраного методу передачі прихованої інформації.

5 Перелік ілюстративного матеріалу (з точним зазначенням обов'язкових креслень): загальний вигляд стеганосистеми, класифікація методів мережевої стеганографії, макет інтерфейсу користувача, блок-схема алгоритму, лістинг програми.

6 Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1,2,3,4	**Захарченко С. М., к.т.н., професор каф. ОТ**		
5	**Кавецький В. В., к.т.н., доцент каф. ЕПВМ**		

7 Дата видачі завдання_____

8 Календарний план

№ з/п	Назва етапів виконання бакалаврської дипломної роботи	Строк виконання етапів роботи	Примітка
1	Постановка задачі роботи	04.10.2021	Виконано
2	Аналіз сучасних технологій захисту інформації.	15.10.2021	Виконано
3	Аналіз та дослідження технологій захисту інформації в комп'ютерних мережах.	29.10.2021	Виконано
4	Розгляд існуючих методів приховування інформації в комп'ютерних мережах.	04.11.2021	Виконано
5	Розробка методів перевірки полів службових заголовків на сторонній вміст	17.11.2021	Виконано
6	Програмна реалізація обраного стеганографічного методу.	21.11.2021	Виконано
7	Підготовка матеріалів та опис розробки механізму передачі прихованої інформації.	30.11.2021	Виконано
8	Оформлення пояснювальної записки	03.12.2021	Виконано
9	Перевірка якості виконання магістерської роботи	10.12.2021	Виконано

Студент _____ Сирцов С. Р.
(підпис) (прізвище та ініціали)

Керівник магістерської кваліфікаційної роботи _____ Захарченко С. М.
(підпис) (прізвище та ініціали)

Консультант з економічної частини _____ Кавецький В. В.
(підпис) (прізвище та ініціали)

ЗМІСТ

ВСТУП

З кожним днем все гостріше стають питання, що стосуються сфери інформаційної безпеки. І з кожним днем їм приділяється все більше уваги. Відбувається це внаслідок того, що персональних даних, які зберігаються на комп'ютерах та передаються в комп'ютерних мережах стає дедалі більше. Процес діджиталізації є неминучим, оскільки використання комп'ютерних технологій в усіх сферах життя в рази спрощує створення, передачу, обробку та накопичення інформації. Проте так само стрімко зростає і рівень загрози конфіденційності та цілісності персональних даних, якщо не забезпечити їм необхідний рівень захисту.

Сучасні системи безпеки повинні захищати дані від різного роду загроз, таких як шпигунство, видалення файлів та інших несанкціонованих дій. Кожен із цих факторів може негативно вплинути на коректне функціонування локальної чи глобальної мережі, що в свою чергу, нерідко приводить до втрати та розголошення конфіденційної інформації. Однією з найпоширеніших загроз в комп'ютерних мережах є несанкціонований доступ ззовні, який проводиться з метою перехоплення та заволодіння певною персональною інформацією.

Безсумнівно сьогодні існує велика кількість комплексних систем безпеки, навіть з використанням штучного інтелекту, які моніторять дані мережі на наявність незвичайної поведінки. Проте хорошу стенографічну роботу помітити зазвичай неможливо.

Беручи все до уваги, можна дійти висновку, що дослідження методів протидії подібних загроз в комп'ютерних мережах є доцільним.

Метою роботи є дослідження та вдосконалення методів виявлення передавання прихованої інформації в комп'ютерних мережах за допомогою протокольних блоків даних.

Задачі дослідження:

— проведення аналізу сучасних методів захисту інформації в комп'ютерних мережах;

— розгляд існуючих стеганографічних методів;

— порівняння та вибір найперспективнішого методу виявлення прихованих даних;

— аналіз службових заголовків протокольних блоків даних на можливість використання їх у якості контейнера для прихованих повідомлень;

— аналіз та розробка методів виявлення стеганографічних контейнерів в службових заголовках протокольних блоків даних;

— реалізація найоптимальніших методів.

Методи дослідження що використовувались у магістерській роботі: методи шифрування даних, методи статистичного аналізу, методи стеганографічного захисту, методи аналізу протокольних блоків даних.

Наукова новизна отриманих результатів магістерської роботи полягає у вдосконаленні методів виявлення прихованої інформації в службових заголовках протокольних блоків даних за рахунок контролю вмісту псевдовипадкових полів IP заголовку, що дозволяє виявляти стеганоконтейнери на пристроях мережевого рівня.

Об’єктом дослідження є процес захисту інформації в комп’ютерних мережах.

Предметом дослідження є методи виявлення стеганографічних контейнерів та вбудованої в них інформації в службових заголовках протокольних блоків даних.

Практичне значення отриманих результатів магістерської роботи полягає в розробці спеціалізованого програмного забезпечення для виявлення передавання прихованих даних завдяки перехопленню та перевірці службових полів протокольних блоків даних мережевого рівня.

Отримані результати магістерської кваліфікаційної роботи **апробовано** на науково-технічній конференції підрозділів Вінницького національного технічного університету «Молодь в науці: дослідження, проблеми, перспективи (МН-2022)» [1].

1 ОГЛЯД ТА АНАЛІЗ ТЕХНОЛОГІЙ ПРИХОВАНОЇ ПЕРЕДАЧІ ІНФОРМАЦІЇ

Реалії сьогодення змусили людство переходити до дистанційного формату взаємодії один з одним. Через це більшість гравців зі сфери надання послуг вимушені були перейти в режим онлайн, що спровокувало стрімкий розвиток комп'ютерних технологій, в тому числі і мережевих. Це був той самий поштовх, який дав змогу відійти від звичного усім ритму життя до ритму, де інформаційні технології займають провідну позицію.

Через різке збільшення трафіку в мережі, великі компанії вимушені були створювати нові та розширювати існуючі датацентри, оскільки їх потужності не були готові до такого. Багатьом підприємствам та звичайним людям довелося модернізуватися, аби іти в ногу з часом. Як наслідок насьогодні ми маємо в мережі інтернет безліч персональних даних, які нажаль є недостатньо захищеними.

Стрімкий розвиток комп'ютерних технологій дозволяє будувати мережі розподіленої архітектури, які об'єднують в собі велику кількість сегментів, що розташовані на значному віддаленні один від одного. Все це викликає збільшення кількості вузлів мереж і різного роду ліній зв'язку між ними. При такому стрімкому впровадженні інформаційних систем, в них залишаються численні вразливості в плані безпеки. А сучасні системи захисту не встигають адаптуватись до все нових і нових методів несанкціонованих проникнень в мережу з метою виведення її з ладу або крадіжки конфіденційної інформації [3].

Захист інформації в комп'ютерних мережах має певні особливості, які пов'язані з тим, що інформація не пов'язана чітко з певним носієм. Вона має змогу швидко і легко копіюватись і передаватись по каналам зв'язку. Загрози конфіденційної інформації можуть при як ззовні, так і зсередини мережі, в якій вона знаходиться. Для того, щоб забезпечити найвищий рівень захисту, необхідно якомога раніше виявити шкідливу програму, яка потрапила до мережі.

Більшість сучасних систем орієнтується на сам вміст пакетів, в яких знаходиться певна інформація, що передається. Або на підозрілу поведінку, яка неможлива в звичайних умовах. Проте злоякісний код, який прихований за допомогою криптографії або стеганографії виявити практично неможливо, якщо спеціально не виявляти його.

Щоб боротись з подібними хакерськими атаками необхідно дослідити методи передавання прихованої інформації, щоб знати де її шукати і що може виступати контейнером.

1.1 Загальні поняття криптографії та стеганографії

1.1.1 Криптографія

Криптографія — це набір методів та алгоритмів для передавання для зберігання інформації на основі шифрування. В його основі завжди лежить алгоритм, за яким буде відбуватись шифрування, та секретний ключ [4].

Для сучасної криптографії притаманне шифрування на основі відкритих алгоритмів, що використовують для своєї роботи обчислювальні засоби. Основною вимогою для всіх алгоритмів шифрування є криптографічна стійкість. Криптографічна стійкість характеризує алгоритм як такий, що здатний протистояти криптографічному аналізу. При умові, що реалізація самого алгоритму шифрування є правильною та використовується ключ достатньої довжини, знайдеться десятки перевірених алгоритмів, які є крипто стійкими.

Серед цих алгоритмів зазвичай зустрічаються ті, що відносяться до однієї з трьох груп [5]:

— симетричні (DES, AES, Camellia, Twofish, Blowfish, IDEA, RC4 та ін.);

— асиметричні (RSA та Elgamal);

— хеш-функції (MD4, MD5, MD6, SHA-1).

Головною особливістю алгоритмів симетричного шифрування є використання спільного секретного ключа між учасниками, що здійснюють обмін даними. З його допомогою відправник виконує шифрування даних та

відправляє їх разом з ключем отримувачу. Отримувач при отриманні повідомлення дешифрує його зміст, використовуючи ключ, який йому надав відправник. Секретний ключ повинен бути відомий лише учасникам обміну і передаватись бажано по захищеному каналу зв'язку (кур'єр, пошта, особиста зустріч тощо).

До найпопулярніших алгоритмів симетричного шифрування відносяться DES (Data Encryption Standard) та IDEA (International Data Encryption Algorithm). В алгоритмі DES використовується комбінація підстановок та перестановок при шифруванні 64-бітних блоків даних 64-бітним секретним ключем. Дешифрування відбувається шляхом виконання тих самих операцій, що використовувались при шифрування, але вже в зворотному порядку. При перевищенні довжини 64 біта повідомлення розбивається на блоки. В такому випадку операції шифрування виконуються над кожним таким блоком окремо, після чого вони з'єднуються послідовно, утворюючи зашифрований файл. Алгоритм IDE також працює з 64-бітними блоками даних, але при цьому використовує ключ довжиною 128 біт, що робить його більше крипто стійким і відповідно безпечнішим [6].

Асиметричні алгоритми також відносяться до систем шифрування з відкритим ключем. В них використовуються пара з двох різних ключів, один із яких є відкритим, а інший секретним. Відкритий ключ використовується для шифрування надісланих даних і залишається у вільному доступі. Будь-який користувач у системі, в якій він використовується, може використовувати його під час надсилання зашифрованого повідомлення. Однак розшифрувати дані з його допомогою неможливо. Одержувач повідомлення може виконати дешифрування, лише якщо у нього є секретний ключ. Генерування ключів відбувається в отримувача для того, щоб не відправляти секретний ключ по відкритих каналам зв'язку. Тобто схема роботи асиметричного шифрування виглядає так. Отримувач генерує секретний та відкритий ключі і відправляє останній відправнику. На основі отриманого ключа відправник шифрує дані і

відправляє їх отримувачу, який за допомогою секретного ключа виконує дешифрування. У порівнянні із симетричними методами шифрування, асиметричні є набагато безпечнішими, але мають меншу швидкодію [7].

Третій метод — хешування. Хешуванням називають перетворення масиву вхідних даних будь-якої довжини у рядок вихідних бітів фіксованої довжини. Такі перетворення також називають хеш-функціями або функціями згортки, а результат називають хеш-кодом, контрольною сумою або дайджестом повідомлення. Результати хешування статистично унікальні. Послідовності, що відрізняються принаймні на один байт, не перетворюються в одне й те саме значення.

Для шифрування в комп'ютерних мережах можуть використовуватись кілька методів, кожен з яких працює на різних рівнях еталонної моделі OSI. Метод шифрування даних на прикладному рівні є одним із найпростіших у реалізації. При його використанні шифруються лише дані, що надсилаються. Додаткова інформація, наприклад, що міститься в службових заголовках протокольних блоків даних, не може бути зашифрованою. Найбільш поширеними криптографічними алгоритмами для цього методу є: DES, CAST, IDEA, SHA, DSS. Такий спосіб швидко та без додаткових затрат дозволяє забезпечити конфіденційність персональних даних.

Для забезпечення прозорості процедури шифрування перед користувачем існує мережевий протокол SSL (Secure Socket Layer), що працює на представницькому рівні. Його незалежність від прикладного рівня надає йому перевагу перед іншими технологіями. Це дає змогу протоколам прикладного рівня, як TELNET, FTP, HTTP, працювати безперешкодно поверх SSL. У даному протоколі використовуються алгоритми шифрування MD5, DSA, DES, RC2, RC4 [8].

Для захисту конфіденційної інформації мережевому рівні, який є більш низьким, застосовується технологія Virtual Private Network, або ж просто VPN. В даній технології засоби криптографії використовуються для того, щоб

забезпечити конфіденційність та цілісність даних, які передаються через мережу загального користування. В даній технології засоби криптографії використовуються для того, щоб забезпечити конфіденційність та цілісність даних, які передаються через мережу загального користування.

В загальному випадку така технологія як криптографія дає гарантію на конфіденційність надісланого повідомлення, проте не захищає дані від можливого перехоплення та знищення. Цей недолік дає нам варіант поглянути в сторону іншої технології.

1.1.2 Стеганографія

Наука, що займається дослідженням та створенням способів передавання та зберігання повідомлень, факт передачі яких приховується, називається стеганографією. Стеганографія суттєво відрізняється від криптографії тим, що приховує саме повідомлення, а не лише його зміст. Її перевагою є те, що передавання таких прихованих повідомлень взагалі не повинно привертати уваги . Криптографічні повідомлення, факт передачі яких не приховується, зазвичай викликають підозри. В деяких країнах криптографія є взагалі забороненою. Стенографічні методи дають можливість вбудовувати в звичайні повідомлення секретний текст, який буде приховано від усіх сторонніх. Запідозрити чи виявити факт існування такого вбудованого таємного повідомлення стає дуже важко. Проте якщо це повідомлення все ж буде розкрито, то прочитати його вміст буде просто, що є недоліком стеганографії.

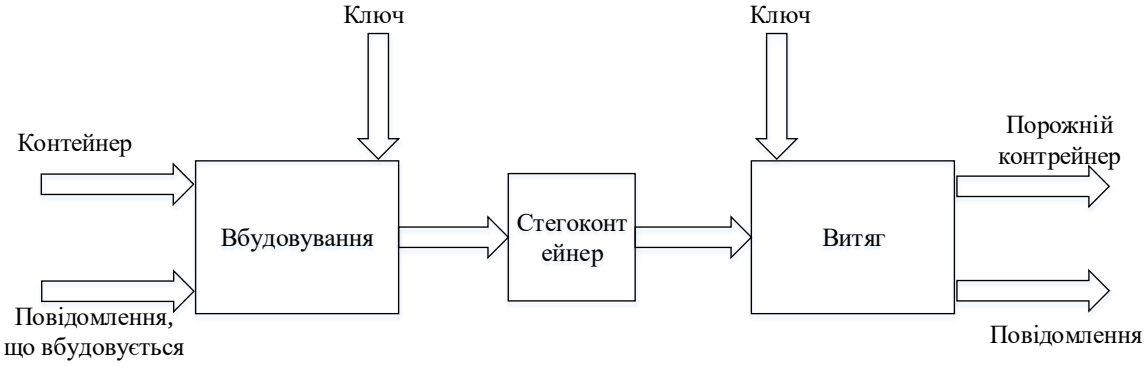

Рисунок 1.1 — Загальний вигляд стеганосистем

З кожним днем все стрімкіше розвивається такий напрямок стеганографії як мережева. Її метою є приховування секретних повідомлень безпосередньо в мережевих потоках, на відміну від решти стеганографічних технологій, які зазвичай працюють з мультимедійними даними. Різке збільшення мережевого трафіку останнім часом створює ідеальне прикриття для зв'язку на основі стеганографії. Є дуже багато різних методів в мережевій стеганографії що дозволяють приховувати дані. Наприклад, можна використовувати надлишкові біти в заголовках протоколів, які залишаються невикористаними при передачі пакетів в мережі, або шляхом зміни порядку і часу їх надходження, щоб приховати додаткову інформацію.

Мережева стеганографія — це напрямок класичної стеганографії, що є заснованим на передачі прихованих повідомлень в комп'ютерних мережах за допомогою особливостей роботи протоколів передачі даних. Носіями секретних даних в ній виступають протоколи еталонної моделі OSI. Мережева стеганографія є сукупністю методів модифікації даних в заголовках мережевих протоколів або структури передачі пакетів в тому чи іншому протоколі. Сімейство цих методів може використовувати один або декілька протоколів одночасно або їх взаємодію між собою для вбудовування прихованих даних за допомогою зміни їх внутрішніх властивостей [9].

Мережева стеганографія стає все більш впливовішою за цифрову і її розвиток є більш стрімким, що зумовлено насамперед двома факторами. По-перше, на відміну від мультимедійних об'єктів таких, як відео, цифрові зображення, аудіо, об'єм прихованого повідомлення, що передається, є необмеженим. В комп'ютерних мережах нові дані можуть передаватись в будь-який час за вимогою, в той час як розмір мультимедійних об'єктів обмежений. По-друге, передавання прихованого повідомлення може здійснюватися частинами напротязі великого проміжку часу, хоч і дуже повільно. Виявити і проаналізувати носії з прихованою інформацією серед безлічі подібних пакетів насправді є дуже трудомісткою задачею, що є безсумнівною перевагою.

Головна мета мережевої стеганографії — приховування певних даних в звичайних процедурах обміну інформацією між користувачами без суттєвої зміни контейнера, в який ці дані вбудовуються. Для більшості стеганографічних методів можна виділити спільні риси які притаманні їм:

— використання мережевого трафіку для приховування обміну даними шляхом створення прихованих каналів зв’язку;

— нерозривно пов’язані з процесом передачі даних (контейнер);

— не вносять суттєвих змін в контейнер.

Між мережевою та класичною стеганографією є чітка відмінність. Класична стеганографія спрямована на живих людей і її метою є введення в оману людських органів чуття. Мережева ж стеганографія спрямована на різного роду мережеві пристрої, серед яких кінцеві системи або проміжні вузли. Для стороннього спостерігача, який не знає про процедуру передачі прихованих повідомлень, обмін стеганограмами залишається прихованим. Це зумовлено тим, що процес вбудовування в носій прихованих даних є «невидимим» для сторін, які не беруть участі в стеганографічній комунікації. Таким чином, секретні дані зазвичай приховуються всередині контейнера. Сам факт передачі такого повідомлення теж відповідно є прихованим.

Контейнер, в який буде вбудовуватись приховане повідомлення, займає особливе місце при створенні та відправці стеганограм. Контейнером називають один або декілька неприхованих потоків трафіку, які встановлюють зв’язок між відправником стеганограми та її отримувачем. Контейнери, що мають певну множину різноманітних місць куди можна вбудувати дані, називаються багатовимірними, а ці місця вважаються підконтейнерами. Підконтейнером можуть виступати поля заголовків чи деяка послідовність пакетів, куди може бути вбудована секретна інформація за допомогою єдиного стеганографічного методу.

Ідеальні стеганографічні контейнери в комп’ютерних мережах повинні відповідати двом основним критеріям:

— вони повинні бути популярні, тобто використання таких контейнерів не повинно сприйматись як щось незвичайне, чим більше таких носіїв присутні в мережі, тим легше буде замаскувати приховане повідомлення і не викликати зайвої підозри;

— модифікація контейнера, яка по суті є процедурою вбудовування в нього стеганограми, не повинна бути помітною та підозрілою особам, які не знають про можливу передачу прихованих даних.

В комп'ютерних мережах існує три основних положення, на яких базується мережева стеганографія. Першим є те, що в мережі завжди можливі різного роду помилки та аномалії, тобто канал зв'язку зазвичай не ідеальний. До подібних явищ можна віднести пошкоджені чи втрачені пакети, неправильний порядок їх надходження тощо. Наслідування їх поведінки дозволить замаскувати процес передачі прихованих даних. По-друге, більшістю мережевих протоколів використовуються не всі наявні поля чи повідомлення, що створює певну надлишковість. Її можна використовувати для вбудовування стеганограм, при умові що це ніяк не пошкодить сам контейнер. По-третє, не кожен протокол є цілком визначеним і допускає «семантичне перевантаження». Певна свобода при їх реалізації може бути використана в стенографічних цілях. Наприклад, поля заголовку HTTP-протоколу можуть знаходитись як в нижньому, так і верхньому регістрах. В такому випадку секретні дані можуть кодуватись шляхом маніпуляцій регістром. На сьогоднішній день практично не існує протоколів, які б виключали можливість їх використання в стеганографічних цілях. Розробка таких протоколів є дуже складною, а частіше навіть неможливою, оскільки доводиться жертвувати функціоналом задля безпеки. Тому мережева стеганографія успішно реалізується навіть в найпростіших протоколах. Проте складніші протоколи є більш багатовимірними і надають більше можливостей для використання витонченіших стеганографічних методів.

Методи мережевої стеганографії діляться на такі групи (рис. 1.2):

а) методи модифікації пакетів;

— методи, що змінюють дані в полях заголовків мережевих протоколів;

— методи, що змінюють дані в полях корисного навантаження пакетів — це різного роду алгоритми водяних знаків, мовних кодеків та інших стеганографічних технік, що приховують дані;

— методи, що об’єднують в собі обидва попередні класи;

б) методи стеганографії, що змінюють структуру і параметри передачі пакетів;

— методи, в яких змінюється порядок надходження пакетів;

— методи, які змінюють затримку між пакетами;

— методи, що роблять навмисні втрати пакетів шляхом втрати порядкових номерів у відправника;

в) змішані (гібридні) методи стеганографії — змінюють і вміст пакетів, і терміни їх доставки, і порядок їх передачі.

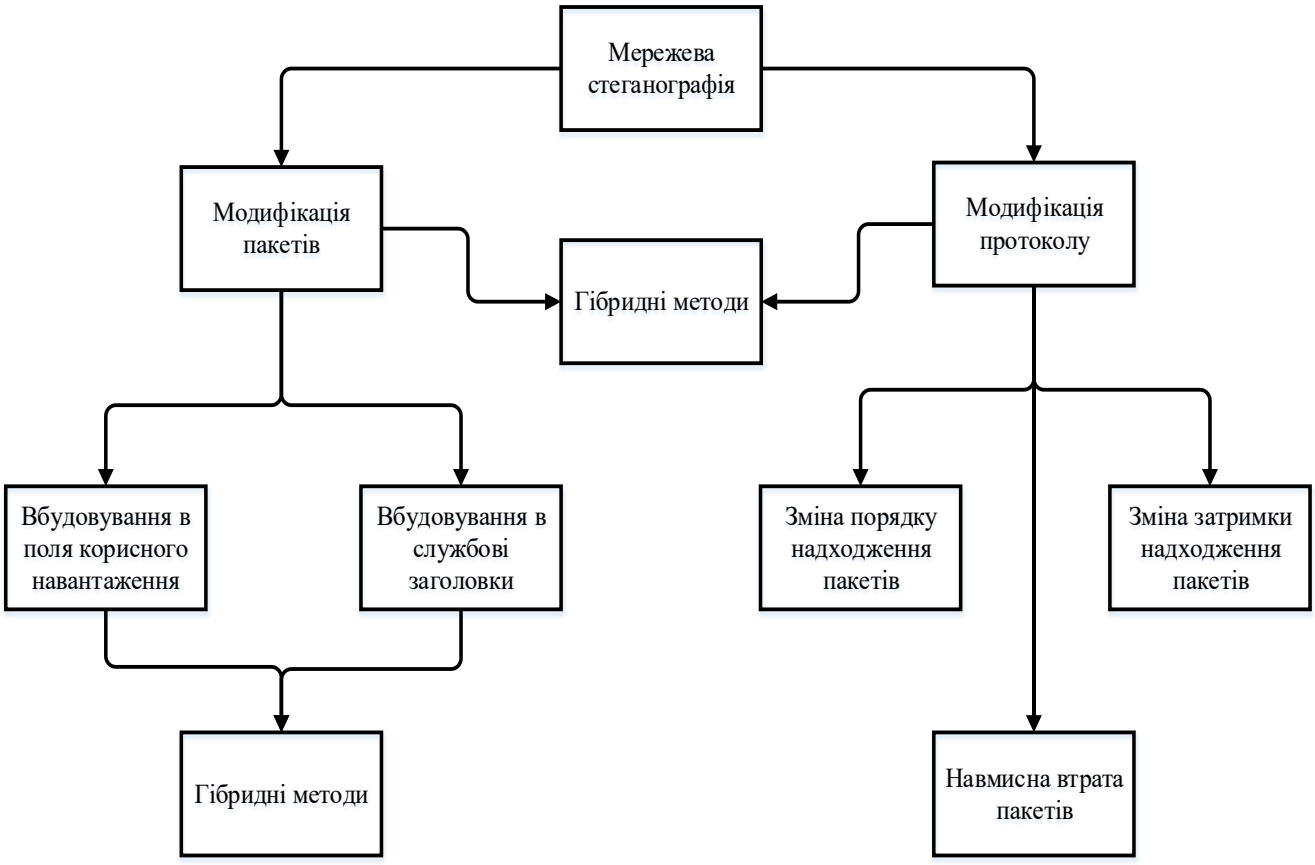

Рис. 1.2. — Класифікація методів мережевої стеганографії

Головна ідея методів модифікації мережевих пакетів – це використання протокольних блоків даних (protocol data unit або PDU) у якості контейнера для передачі стеганограми. Вони є досить поширеними, оскільки їх реалізація є більш простою, а дані що надходять від відправника до отримувача ніяк при цьому не змінюються. Даний спосіб надає непогану пропускну здатність, яка досягається завдяки відправці великої кількості модифікованих пакетів. Що важливо, використання цих методів не вимагає синхронізації в часі між користувачами, які обмінюються даними. Це приводить і до зменшення вартості стеганографії при використанні цих методів.

Хоч і попередній варіант приховування даних є досить простим, та найпростішим все ж є метод вбудовування стеганограм в поля корисного навантаження. Ці поля не містять жодної службової інформації, лише дані, що передаються, та інкапсульовані фрейми нижчих рівнів. Їх модифікацію відслідкувати дуже легко, якщо є пасивний спостерігач, який аналізує пакети всередині мережі. При вдалому перехопленні такого пакету прочитати приховані в ньому дані простіше простого.

Існують методи, що дають змогу передавати стеганограми в службових заголовках протокольних блоків даних. Оскільки в процесі обміну пакетами в мережі не завжди є необхідність у використанні всіх полів службових заголовків, часто в них виникає певна надлишковість. Тобто існують певні умови, коли певні поля не використовуються і теоретично їх можна використовувати для вбудовування в них секретних даних. Найчастіше використовуються поля заголовків TCP та IP протоколів.

1.2 Методи модифікації пакетів для приховування інформації

Є досить багато методів мережевої стеганографії, кожен з яких пропонує унікальний підхід в передачі прихованої інформації. Зрозуміло, що всі вони мають свої переваги та недоліки. На їх основі створюються методи, що об’єднують в собі досягнення попередніх з метою досягнення максимальної

ефективності й уникнення відомих недоліків. Щоправда такі гібридні методи зазвичай складніші в реалізації і потребують додаткового часу для їх впровадження.

Побудувати стеганографічний тунель в комп'ютерній мережі можна на будь-якому рівні моделі OSI. Щоб обрати найбільш оптимальний метод необхідно розібрати найпоширеніші та найперспективніші з них.

Оскільки при переході між рівнями в стеці протоколів відбувається інкапсуляція і декапсуляція, то на кожному рівні відповідно додається або вилучається заголовок цього рівня. Для доставки повідомлення в місце призначення або повідомлення при невиконанні дії в заголовках міститься необхідна службова інформація. Щоб передавати приховану інформацію, можна використовувати стек протоколів транспортного та мережевого рівня TCP/IP.

Транспортний рівень мережі відповідає за управління доставкою пакетів від відправника до отримувача і гарантує правильну послідовність їх надходження.

1.2.1 Заголовки протоколів транспортного рівня

При використанні протоколу UDP неможливо пересвідчитись чи надійшло повідомлення адресату чи ні. За це його часто називають «ненадійним». Проте він зайняв свою нішу, в якій справляється краще всіх. UDP використовується для передачі потокового відео та в комп'ютерних онлайн іграх, де допускається втрата пакетів.

Рисунок 1.1 — Структура заголовку UDP

Оскільки протокол не може забезпечити надійної передачі даних, то розглядати його заголовок у якості стегоконтейнера не є доцільним. При передачі стеганограми у вигляді множини прихованих пакетів будь-яка, навіть найменша втрата, приведе до її спотворення і неможливості прочитати.

Більш перспективним в цьому плані виглядає протокол TCP. На відміну від UDP він забезпечує надійний потік, шляхом попереднього встановлення з’єднання. TCP гарантує, що дані надійдуть безпомилково, оскільки при виникненні яких-небудь втрат, він повторно робить запит на отримання даних для уникнення їх втрати та дублювання. Також він відрізняється від UDP тим, що дані завжди отримуються в правильній послідовності.

Проаналізувавши структуру службового заголовку протоколу TCP (рис.1.2), можна визначити поля які будуть підходити на роль стегоконтейнера якнайкраще.

Рисунок 1.2 — Структура заголовку TCP

Порти відправника та отримувача змінювати не можна тому що, повідомлення на своєму шляху може не дійти до місця призначення, бо деякі порти заблоковані.

Порядковий номер використовується для збору сегментів у єдине ціле і його використання не є можливим оскільки порушення цього поля не дозволить зібрати повідомлення в потік. Але збір сегментів в потік не є обов'язковим, оскільки приховане повідомлення може бути вилучено раніше, тому його можна модифікувати при умові, що пакет буде опрацьовуватись приймачем і його доставка до певного додатку не є обов'язковою.

Номер підтвердження цілком підходить для використання в якості контейнера, якщо ініціюється передача потоку в певну кількість сегментів і підтвердження відправляється один раз після завершення передачі потоку.

Як контейнер для прихованого повідомлення не можливо буде використати довжину заголовку та прапори тому, що при зміні цих значень повідомлення може ідентифікуватись як пошкоджене і буде відкинуте. Також не можна змінювати значення полів розмір вікна, контрольна сума і вказівник важливості. Їх зміна може привести до порушення передачі сегмента.

Отже у якості контейнера на постійній основі можна використовувати тільки поле «Номер підтвердження» та за певних умов «Порядковий номер». В сумі їх використання дає 8 байт.

1.2.2 Заголовок протоколу IPv4

На рівні мережі працює протокол IP, який забезпечує передачу даних між мережами. Передача пакетів, на відміну від фреймів, відбувається на значно більші відстані.

Якщо звернути увагу на структуру IP-пакету протоколу IPv4 (рис. 1.3), можна побачити, що поле «Ідентифікатор пакету» потенційно може бути використане в якості стегоконтейнера. Зумовлено це тим, що його значення може генеруватись з боку відправники і бути псевдовипадковим. Тобто це поле містить випадкову числову послідовність, яка генерується разом зі створенням пакету. Проте коли виконується фрагментація пакету, то значення цього поля несе в собі певну інформацію і генерується не псевдовипадково. Для використання цього поля як контейнера необхідно спочатку дізнатись значення

MTU в мережі, в якій відбуватиметься передача. MTU (Maximum transmission unit) — це максимальний розмір корисного блоку даних одного пакету, який може бути переданий протоколом без фрагментації. Отож, щоб пакет не фрагментувався для передачі, достатньо просто не перевищувати цього значення. Породжується певна надлишковість і в полі «Прапори», якщо передача даних відбувається без фрагментації пакету. Це надає нам ще один підконтейнер, який ми можемо використовувати разом з ідентифікатором пакету і отримати 19 біт для приховування даних.

Також поля «Час життя» та «Зсув фрагмента» додатково надають до 49 біт, які ми можемо використати для створення стегоконтейнера.

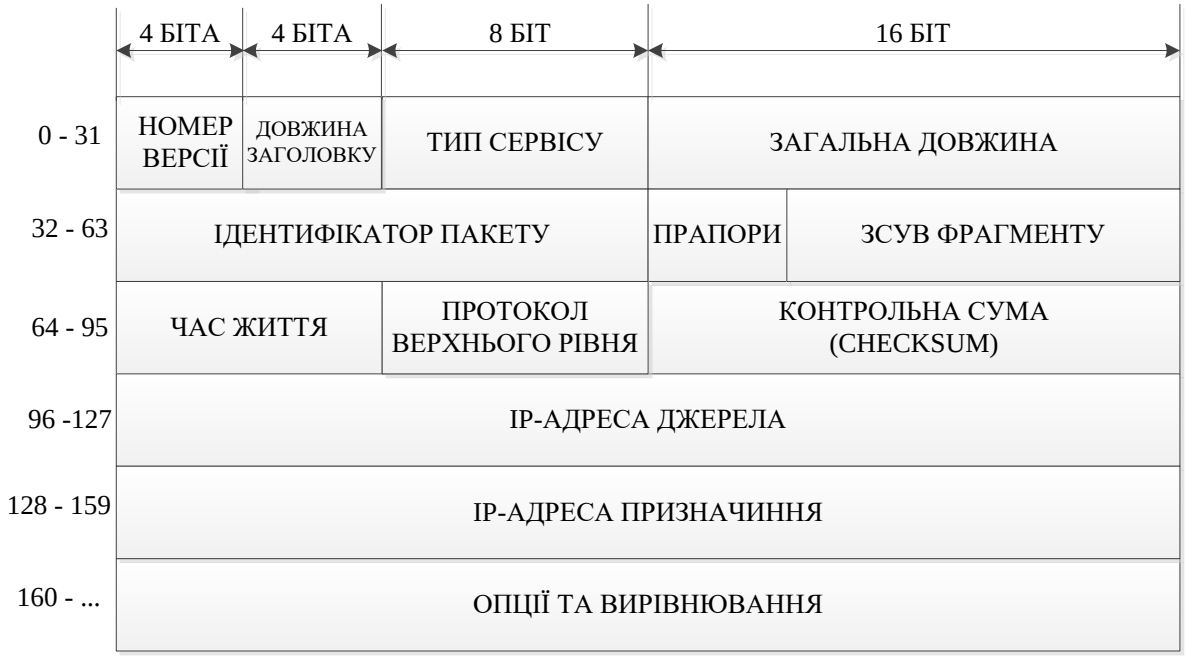

Рисунок 1.3 — Структура заголовку протоколу IPv4

Стеганографія на основі цього методу приховує інформацію без зміни користувацьких даних, має хорошу пропускну здатність та низьку вартість, не порушує цілісність та функціонал пакету, що є безперечними перевагами. Основним недоліком залишається лише, те що при виявленні цих прихованих даних їх без труднощів можна прочитати. Проте якщо застосувати один із криптографічних методів, то дані вдасться захистити.

1.2.3 Заголовок протоколу IPv6

Протокол версії IPv6 розроблявся спеціально з метою заміни застарілого протоколу IPv4. Він повинен стати кращим за попередника в сферах мобільності, безпеці та адресації тощо. Головна причина, через яку вирішили замінити IPv4 — це вичерпування його адрес. У протоколу оновленого стандарту розмір складає 128 біт, що повинно вирішити проблему на довгий час. Перехід від IPv4 до IPv6 виявився не із легких, оскільки згортання старого протоколу займає певний час. Тому досі активно використовуються обидва. Але без сумнівів IPv6 — це протокол майбутнього і рано чи пізно відбудеться повний перехід на нього.

При розробці нового протоколу за основу було взято IPv4, внаслідок чого IPv6 успадкував немало полів заголовків, якими можна скористатись при створенні стегоконтейнера. Проте не всі механізми приховування даних, що були розроблені для IPv4, будуть працювати і для нього, оскільки новий протокол зазнав значних змін та нововведень.

Для того, щоб розібратись в яких же полях заголовку IPv6 є певна надлишковість, яку можна використати для приховування даних, необхідно поглянути на його структуру (рис. 1.4). В протоколі нового стандарту залишились як старі поля, так і з’явились підходящі нові, які варто розглянути на роль стегоконтейнера [10].

Поле «Клас трафіку» вказує на пріоритет пакету. Має довжину 8 біт. Перші 6 біт використовуються DSCP (Differentiated Services Code Point) для класифікації пакетів. Решта 2 біти використовуються ECN (Explicit Congestion Notification) для керування потоком передачі між кінцевими пристроями. Слабо підходить для вбудовування в нього даних, оскільки розмір має невеликий, а на проміжних вузлах взагалі може змінюватись, що руйнує прихований канал.

Поле «Мітка потоку» дозволяє спростити процедуру маршрутизації однорідного потоку пакетів. Його довжина становить 20 біт і аналогічно до поля «Ідентифікатор пакету» в протоколі IPv4 його вміст генерується

псевдовипадково і не може бути переписаний проміжними пристроями, що дозволяє приховати в ньому секретну інформацію.

Поле «Довжина корисного навантаження» вказує на розмір поля даних датаграми, який може складати 65536 байт. За допомогою маніпуляцій з цим полем, ми можемо збільшити розмір корисного навантаження пакету і приховати в ньому нашу стеганограму. Потрібно пам’ятати, що контрольну суму потрібно оновлювати правильним чином, аби уникнути порушення роботи протоколу. Насамперед це потрібно для того, щоб мережеві проміжні пристрої не відкинули пакет як пошкоджений. Прихована інформація повинна бути прихована і зчитана до моменту надходження даних користувачу. Пропускна здатність такого стеганографічного тунелю обмежена лише максимальним розміром датаграми і може як завгодно варіюватись [11].

Рисунок 1.4 — Структура заголовку протоколу IPv6

Поле «Наступний заголовок» визначає наступний заголовок, що міститься в полі корисного навантаження пакету. Типовими значеннями є: 6 — TCP, 58 — ICMPv6, 17 — UDP, 1 — ICMP. Інформацію можна приховати, налаштувавши «Наступний заголовок» так, що він буде вказувати на фіктивний додатковий

заголовок, в якому будуть знаходитись дані. Як і в попередньому способі, датаграма повинна бути відновлена до того як дійте до місця призначення. Пропускна здатність залежить від розміру фіктивних заголовків, що вводяться.

Також може бути створений прихований канал шляхом заміни деяких бітів в полі «Адреса відправника» з пропускною здатністю аж у 128 біт. Проте надійність цього способу є сумнівною. Лише у випадку, коли обидві кінцеві точки розміщені в межах відкритих вузлів, може відбуватись маніпулювання адресами. Інакше загальнодоступні інструменти для захисту від спуфінгу можуть легко виявити модифікацію і обірвати з'єднання.

1.2.4 DNS-тунелювання

Domain Name System (DNS) — це важлива ієрархічна розподілена система, яка дозволяє перетворювати доменне ім'я хоста в IP-адресу. Простіше кажучи, DNS допомагає знайти правильну адресу за допомогою доменного імені. Зараз важко уявити стек мережевих технологій без DNS, чим залюбки користуються зловмисники.

Широке використання цієї технології привело до виникнення такого явище як DNS-тунелювання. Існує воно з початку 2000-х років, коли NSTX — простий у використанні інструмент був опублікований для мас. З тих пір з'явилася чітка тенденція – посилення безпеки міжмережевого екрану призвело до ширшого поширення тунелювання DNS. До 2011 року він уже використовувався такими шкідливими програмами, як Morto і Feederbot для управління та контролю, а також популярним шкідливим ПЗ для торгових точок - FrameworkPOS для ексфільтрації кредитних карт.

Спочатку DNS був створений для дозволу імен, а не для передачі даних, тому часто не розглядається як загроза зловмисного обміну даними та крадіжки даних. Оскільки DNS — це протокол, що добре зарекомендував себе і користується довірою, хакери знають, що організації рідко аналізують пакети DNS на предмет зловмисних дій. DNS приділяють менше уваги і більшість

організацій зосереджують ресурси на аналізі веб-трафіку або трафіку електронної пошти, де, на їхню думку, найчастіше відбуваються атаки. Насправді, для виявлення та запобігання DNS-тунелюванню потрібен ретельний моніторинг кінцевих точок.

Більше того, набори інструментів для тунелювання перетворилися на індустрію та широко доступні в Інтернеті, тому хакерам справді не потрібно вдаватись в технічні складнощі для реалізації атак із тунелюванням DNS. Нещодавнє дослідження показало, що кількість атак DNS лише у Великій Британії за останній рік збільшилася на 105%. DNS-тунелювання привабливе тим, що хакери можуть отримувати будь-які дані у вашу внутрішню мережу та з неї, минаючи більшість брандмауерів. Чи використовується він для управління та контролю (C&C) скомпрометованих систем, витоку конфіденційних даних за межі або для тунелювання всередині вашої закритої мережі, DNS-туннелювання залишається серйозним ризиком для вашої організації [12].

Серед найпоширеніших випадків зловживання DNS та інструментів, за допомогою яких вони здійснюються можна виділити три основних. Першим є управління та контроль шкідливими програмами(C&C). Шкідливі програми можуть використовувати DNS-туннелювання для отримання команд від своїх керуючих серверів та завантаження даних в Інтернет, не відкриваючи жодного TCP/UDP-з'єднання із зовнішнім сервером. Такі інструменти, як Dnscat2, створені спеціально для використання з метою управління та контролю.

Другий спосіб — це створення тунелю «обходу фаерволу». DNS-тунелювання дозволяє зловмиснику проникнути у внутрішню мережу, створивши повноцінний тунель. Такі інструменти, як Iodine, дозволяють створювати спільну мережу між пристроями, створюючи повноцінний тунель IPv4.

Третім є обхід авторизованих порталів для платного Wi-Fi. Багато систем з порталами, що адмініструються, пропускають весь трафік DNS, тому можна тунелювати IP-трафік без плати. Деякі комерційні послуги навіть надають

серверний тунель як послугу. Такі інструменти, як Your-Freedom створені спеціально для виходу із захоплених порталів.

Щоб зрозуміти в повній мірі як працює DNS-тунелювання необхідно розглянути покроково дії, які робить зловмисник для створення прихованого тунелю. Спочатку зловмисник отримує домен, наприклад, evilsite.com. Далі він налаштовує сервери доменного імені на свій власний DNS-сервер. Потім зловмисник делегує піддомен, наприклад tun.evilsite.com, і налаштовує власний комп'ютер в якості повноважного DNS-сервера піддомену. Після цих маніпуляцій будь-який DNS-запит, який робиться жертвою до «{data}.tun.evilsite.com», зрештою досягає машини зловмисника. Машина зловмисника кодує відповідь, яка надходить назад на машину жертви. У результаті двонаправлений канал для передачі даних за допомогою DNS-тунелювання створено. Детальніше процес його створення можна розглянути на рисунку 1.5.

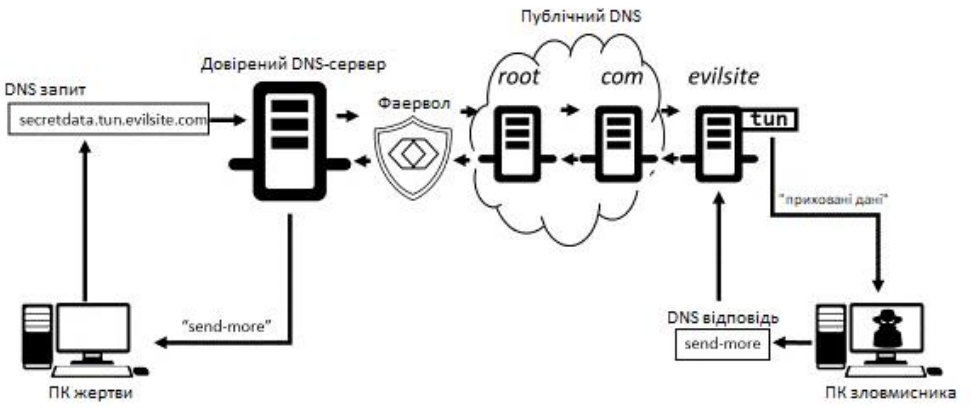

Рисунок 1.5 — Процес створення прихованого каналу за допомогою DNS

Брандмауера вже недостатньо для ізоляції внутрішньої мережі та забезпечення її захисту, а DNS-туннелювання – лише один із творчих прийомів, які кіберзлочинці використовують для виходу із внутрішніх мереж. Для боротьби з тунелюванням DNS тягар перемістився з мережі на кінцеву точку.

2 РОЗРОБКА МЕТОДІВ ВИЯВЛЕННЯ ТА ПРОТИДІЇ ПЕРЕДАВАННЯ ПРИХОВАНОЇ ІНФОРМАЦІЇ В КОМП'ЮТЕРНИХ МЕРЕЖАХ

З початку XX століття і до сьогодні дуже активно розвивалась як сама стеганографія, так і суміжна їй наука — стегоаналіз. Стегоаналіз — це наука про виявлення факту вбудовування прихованої інформації в контейнер.

Сьогодні ми спостерігаємо за невтішними і загрозливими тенденціями, що все більше розробників шкідливого програмного забезпечення і засобів кібершпигунства надають перевагу використанню стеганографії. Більшість сучасних антивірусних систем не можуть гарантувати надійного захисту від подібного ПЗ, або не захищають взагалі. Хоча потрібно розуміти, що кожен такий заповнений стегоконтейнер становить загрозу для будь-якої локальної мережі. Він може містити приховані дані, які перехоплюються і витягуються шпигунським програмним забезпеченням, для зв'язку центральним сервером або оновлення для модулів шкідливої програми.

Виявлення і усунення зловмисного використання стеганографії, що відбувається в комп'ютерній мережі, є дуже важким. Згідно з Барвайзом(2018): «Якщо зловмисник може успішно проникнути в мережу і, не викликаючи підозри, встановити шкідливе ПЗ, що використовує цифрову стеганографію для приховування своєї присутності, тоді і мережа і всі дані, які містяться в ній, можна вважати повністю скомпрометованими.(Теоретична основа)» [13]. Це хороший опис того, наскільки важко виявити і відреагувати на використання методів, що приховують дані проти ваших інформаційних ресурсів.

Можна явно виділити три основні причини чому автори шкідливого програмного забезпечення використовують стеганографічні методи в своїх розробках:

— стеганографія дозволяє приховати сам факт завантаження/ вивантаження даних, а не лише самі дані;

— дозволяє обійти DPI-системи, що зазвичай використовують в корпоративних мережах, де це актуальним;

— антивірусні системи взагалі мало що можуть зробити із заповненими контейнерами, їх практично нереально виявити, оскільки вони виглядають як звичайні пакети.

Коли діло доходить до засобів протидії стеганографії, то виникає серйозна проблема у вигляді великої різноманітності існуючих методів приховування інформації. Мережева стеганографія може застосовуватись не лише до різноманітних протоколів і мультимедійних даних, що передаються, а й з використанням різного роду методів приховування таких як: RSTEG, LACK, TranSteg, HICCUPS тощо. На даний момент відомо понад декілька сотень методів приховування інформації, включаючи їх комбінації. Навіть якщо не брати до уваги методи, що використовують корисне навантаження пакетів, то залишається більше сотні методів, які передають секретні дані з використанням метаінформації, такої як елементи службових заголовків прокольних блоків даних.

З іншого боку, контрзаходи не можуть покривати усі доступні стеганографічні методи одночасно через високу складність та різноманітність протоколів і сервісів, і впливають тільки на один чи декілька методів приховування кожен. Водночас, якщо засіб протидії спрямований на декілька методів приховування одночасно, то забезпечення високої точності залишається невирішеною задачею. Якщо ж створювати комплексні системи на основі найоптимальніших методів, то виявлення, обмеження та запобігання мережевої стеганографії стане ще більш складною задачею.

Тому перед тим як створювати контрзаходи для складних сценаріїв, необхідно для початку провести дослідження і запропонувати фундаментальні підходи для протидії стеганографічним методам. Після чого можна буде розробляти продукти, що будуть забезпечувати захист в організованих мережевих середовищах.

Для того, щоб розробити методи виявлення факту передачі прихованої інформації, необхідно розглянути основні етапи створення пакетів із вбудованим в них стегоконтейнером.

2.1 Процес створення пакетів зі стегоконтейнером

Як відомо, всі інструменти, що використовуються системними адміністраторами для оцінки безпеки своїх мереж, є як благом, так і напастю. Тож всі ці інструменти можуть використовуватись хакерами для виявлення вразливостей мереж та їх експлуатації у власних цілях. Техніка створення пакетів зі стегоконтейнером не є виключенням, і оскільки вона лежить в основі технічно важкого методу експлуатації вразливостей, то її застосування важко виявити і встановити.

Ethernet-фрейм містить велику кількість полів, які зазвичай використовуються для підтримання з’єднання другого рівня моделі OSI, в той час, як пакети TCP та IP містять дані більш високих рівнів. Частина пакету, що відноситься до протоколу TCP, надає гарантію успішної передачі, а пакет протоколу IP містить адреси та номера портів відправника та отримувача[14].

Значна частина з полів цих пакетів може піддаватись модифікації з боку хакерів при проведенні атаки на певну мережу. IP-адреси та номери портів модифікуються найчастіше при проведенні атак відмови в обслуговуванні або при використанні спуфінгу пакетів. Фактично кожен прапорець и значення кожного поля TCP-пакету може бути модифіковано з ціллю проникнути в систему.

Варто звернути увагу на те, що методи створення пакетів та спуфінгу пакетів чимось схожі, проте цілковито відрізняються в плані результату. Спуфінг використовується зловмисниками для приховування ідентифікаційних даних та факту присутності в мережі. Спуфінг пакетів використовується зазвичай для отримання інформації про мережу, яка включає в себе список відкритих портів,

служб що виконуються на вузлах, активних вузлів тощо. А протягом проведення атаки цільовий вузол не в змозі відслідкувати атакуючого.

З іншого боку, атаки на основі створення пакетів можуть заходити далі, що дозволяє визначити наявність, функціональність і точність задання правил міжмережевого екрану і систем виявлення проникнень. Створення пакетів вимагає глибоких знань в області TCP-протоколу та принципів його роботи, що робить атаку з його використанням заснованою більше на основі ручної модифікації, аніж програмної.

Сам процес створення пакета можна умовно розділити на чотири основних етапи: складання пакету, редагування, відправка та аналіз (рис. 2.1).

Рисунок 2.1 — Етапи створення модифікованого TCP-пакету

Складання пакету — це перший етап створення пакетів, на якому атакуючий вирішує яку мережу потрібно зламати, намагається отримати максимальну кількість інформації про можливі вразливості і генерує пакети для відправлення. Ці пакети ретельно перевіряються, щоб впевнитись, атака буде максимально непомітна в мережі і не буде ідентифікована. Наприклад, пакет може мати підмінену вихідну адресу і випадковий номер послідовності TCP. При

цьому збірка пакета не обов’язково повинна проходити з нуля. Для вбудовування прихованої інформації можна перехопити пакет, що передається по мережі.

Редагування пакету є наступним кроком. На цьому етапі проводиться перша спроба відправлення модифікованого пакету для тестування та виправлення можливих помилок на с основі отриманих даних для переходу до наступного кроку. В процесі редагування пакету головним завданням є отримання якомога більшої кількості інформації при відправці мінімальної кількості пакетів в мережу. Наприклад, для тестування того як міжмережевий екран реагує на модифіковані пакети, може бути створений простий пакет з неіснуючою IP-адресою та встановленим бітом в полі ACK. В ідеальному випадку екран повинен відкинути такий пакет.

Як тільки вдається створити коректний пакет або набір пакетів, відбувається перехід до етапу відправлення. Сформовані пакети відправляються в мережу і відбувається складання пакетів, що відправляються у відповідь для проведення наступного дослідження та встановлення залежностей. На цьому етапі і проводиться атака. Якщо не вдалося досягти бажаного результату, доводиться повертатись до попереднього кроку, допоки не вдасться вдало провести приховану атакау.

На останньому етапі, етапі аналізу пакету, відбувається створення та декодування пакетів, що були відправленні у відповідь цільовою мережею. Зловмисники можуть використовувати для цієї цілі прості сніфери пакетів або просто виконувати запис прийнятих пакетів у файл для подальшого дослідження. На цьому етапі, аналізуючи результати, стає відомо чи вдалось проникнення успішно. Якщо результати невтішні, то хоча б отримуються вихідні дані для внесення коригувань та зміни методів атаки, що допоможуть надалі покращити результати.

Стандартними техніками атак зі створенням пакетів є маніпуляція прапорцями пакетів, дублювання пакетів, маніпуляція полем протоколу, створення напіввідкритих з’єднань тощо.

2.2 Методи виявлення прихованої інформації в пакетах

2.2.1 Перевірка змісту псевдовипадкових полів

Як було розглянуто вище при створенні та відправці пакетів в мережі частина полів службових заголовків може не використовуватись. Наприклад, поле «Ідентифікатор пакету», що знаходиться в заголовку протоколу IPv4, не використовується, якщо не виконується фрагментація пакетів. Зазвичай це відбувається коли розмір корисного навантаження не перевищує максимально допустимий в мережі і пакет не потрібно ділити на окремі частини для передачі. В цьому випадку вміст поля «Ідентифікатор пакету» заповнюється за допомогою генерації псевдовипадкової послідовності.

Усі генератори чисел діляться на два типи: генератор справжніх випадкових чисел (ГСВЧ) та генератор псевдовипадкових чисел (ГПВЧ). Основна відмінність їх відмінність полягає в тому, що ГСВЧ створює числа на основі непередбачуваних фізичних явищ, наприклад, таких як шуми атмосфери, радіоактивний розпад або космічне випромінювання. Проте такі системи зазвичай дуже затратні в установці та експлуатації. Вигіднішою, але менш точною, альтернативою є ГПВЧ, які генерують числа за допомогою використання математичних алгоритмів, які повністю залежать від комп’ютера. Оскільки генератори справжніх випадкових чисел мають ряд недоліків, частіше використовуються генератори псевдовипадкових чисел [15]. Головними недоліками ГСВЧ є:

— час та трудовитрати при установці та налаштуванні;

— дороговизна;

— генерація випадкових чисел відбувається повільніше, ніж при програмній реалізації ГПВЧ;

— неможливість відтворення послідовності випадкових чисел, що була згенерована раніше.

Водночас ГСВЧ залишається затребуваним, оскільки ГПВЧ не може генерувати настільки ж рівномірні послідовності з «більш випадковими» бітами.

Послідовні значення в генераторі псевдовипадкових чисел все ж не є незалежними. Для порівняння цих двох систем можна поглянути на рисунок 2.2, де за допомогою растрового генератора показано різницю між ними. Чітко видно, що ГПВЧ в результатах видає числа, які формують деякий закономірний візерунок. Чого не можна сказати про систему, яка базується на основі фізичних процесів.

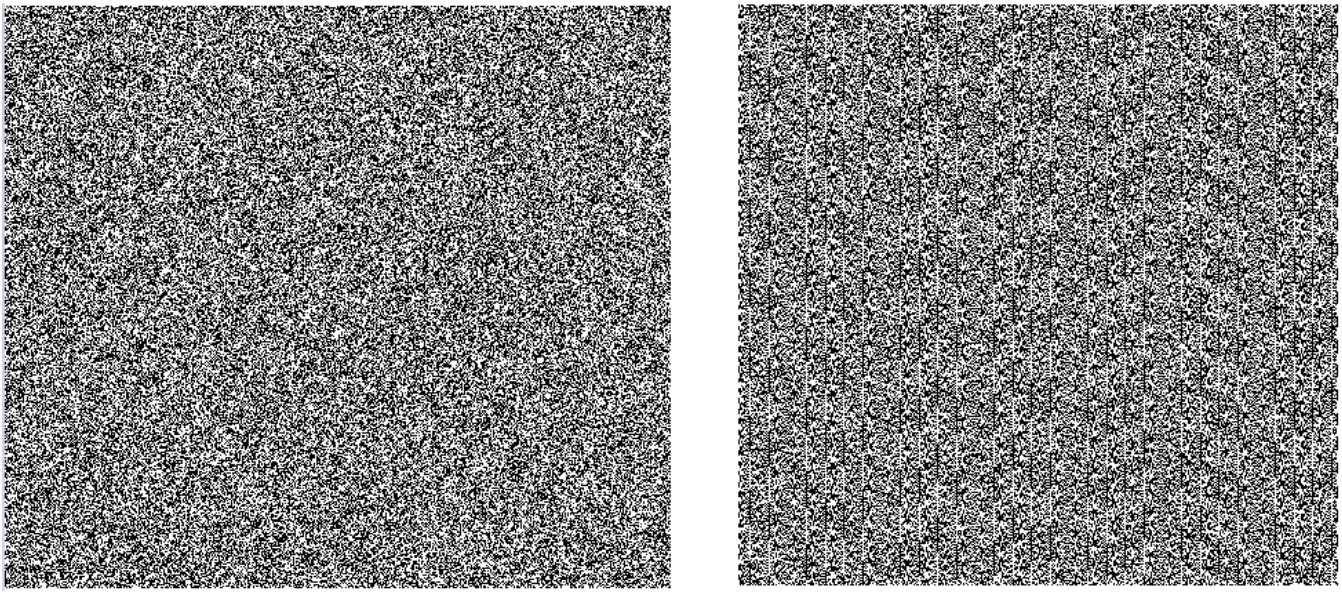

Рисунок 2.2 — Растр згенерованих чисел ГСВЧ та ГПВЧ

Проте тут показаний не найкращий випадок генерації псевдовипадкових чисел. Велику роль грає операційна система, мова програмування та математичний алгоритм, за якими будувалась числова послідовність. Більшість ГПВЧ добре справляються зі своєю задачею, деякі навіть відмінно, проте жоден з них не є істинним.

Головною особливістю справжніх випадково згенерованих чисел є те, що отримана послідовність буде максимально рівномірно розподілена. Тобто такою, де кількість бітів одиниць буде рівною кількості бітів нулів, при чому за кожною одиницею зазвичай буде слідувати нуль. Для перевірки псевдовипадкових чисел на їх істинність існують спеціально розроблені набори методів.

Тестування псевдовипадкових послідовностей — це сукупність методів визначення тотожності заданої псевдовипадкової послідовності до випадкової. В

якості міри зазвичай виступає наявність рівномірного розподілу, великого періоду, рівної частоти появи однакових підрядків тощо. Існує дві основних категорії тестів: графічні та статистичні. До графічних відносяться тести, результати яких відображаються у вигляді графіків, що характеризують властивості досліджуваної послідовності. Їх результати інтерпретуються людиною, тому висновки на їхній основі можуть бути неоднозначними. Статистичні тести, на відміну від графічних, видають чисельну характеристику послідовності і дозволяють сказати однозначно, чи пройдений тест. Серед найвідоміших можна виділити тести Д. Кнута, DIEHARD, NIST.

Найбільш підходящі методи для тестування службових полів на псевдовипадковість містить набір статистичних тестів NIST.

2.2.2 Статистичні тести NIST

Статистичні тести NIST — це пакет статистичних тестів, розроблений Лабораторією інформаційних технологій, що є головною дослідницькою організацією Національного інституту стандартів і технологій (NIST). До складу пакету входять 15 статистичних тестів, метою яких є визначення міри випадковості двійкових послідовностей, що були згенеровані або апаратно, бо програмно. Ці тести засновані на різного роду статистичних властивостях, які притаманні тільки випадковим послідовностям. Кожен із тестів отримує вхідні дані у вигляді скінченної послідовності. Потім обчислюється статистика, яка характеризує деяку властивість даної послідовності — це може бути як одне значення, так і декілька. Після цього отримана статистика порівнюється з еталонною, яку дасть ідеальна випадкова послідовність. Еталонна послідовність виводиться математично, чому присвячено багато теорем та наукових робіт.

В основі тестів лежить поняття нульової гіпотези. Допустимо, ми набрали якусь статистичну інформацію. Наприклад, нехай це буде кількість людей, які захворіли на рак легень у групі з 1000 людей. І нехай відомо, що деякі люди з цієї групи є курцями, а інші ні, причому відомо які конкретно. Стоїть таке завдання:

зрозуміти, чи є взаємозв'язок між курінням та захворюванням. Нульова гіпотеза — це припущення, що між двома фактами немає будь-якого взаємозв'язку. У нашому прикладі це припущення, що куріння не викликає раку легень. Існує також альтернативна гіпотеза, яка спростовує нульову гіпотезу: тобто між явищами взаємозв'язок існує (куріння спричиняє рак легень). Якщо переходити до термінів випадкових чисел, то за нульову гіпотезу приймається припущення, що послідовність є справді випадковою (знаки якої з'являються рівноймовірно та незалежно один від одного). Отже, якщо нульова гіпотеза правильна, наш генератор виробляє досить «хороші» випадкові числа.

Як перевіряється гіпотеза? З одного боку, ми маємо статистику, підраховану на основі фактично зібраних даних (тобто за послідовністю, що вимірюється). З іншого боку, є еталонна статистика, що одержується математичними методами (теоретично обчислена), яку мала справді випадкова послідовність. Вочевидь, що зібрана статистика неспроможна зрівнятися з еталонною — хоч би як не був хорошим наш генератор, він все одно не ідеальний. Тому вводять певну похибку, наприклад 5%. Вона означає, що якщо, наприклад, зібрана статистика відхиляється від еталонної більш ніж на 5%, то робиться висновок про те, що нульова гіпотеза не вірна з великою надійністю.

Оскільки ми маємо справу з гіпотезами, існує 4 варіанти розвитку подій:

— зроблено висновок у тому, що послідовність випадкова, і це правильний висновок;

— зроблено висновок, що послідовність не випадкова, хоча вона була насправді випадкова. Такі помилки називають помилками першого роду;

— послідовність визнана випадковою, хоча насправді такою не є. Такі помилки називають помилками другого роду;

— послідовність справедливо відбракована.

Імовірність помилки першого роду називають рівнем статистичної значущості та позначають як α. Тобто α — це імовірність відбракувати «хорошу»

випадкову послідовність. Це значення визначається сферою застосування. У криптографії прийнято брати від 0.001 до 0.01.

У кожному тесті обчислюється так зване Р-значення: це імовірність того, що піддослідний генератор зробить послідовність не гірше, ніж істинний гіпотетичний. Якщо значення = 1, то наша послідовність ідеально випадкова, а якщо воно = 0, то послідовність цілком передбачувана. Далі Р-значення порівнюється з α, і якщо воно більше за α, то нульова гіпотеза приймається і послідовність визнається випадковою. В іншому випадку - відбраковується.

В даних тестах береться α = 0.01, чого слідує що:

— якщо Р-значення ≥ 0.01, послідовність визнається випадковою з рівнем довіри 99%;

— якщо Р-значення < 0.01, послідовність відбраковується з рівнем довіри 99%.

Надалі буде розглянуто найбільш оптимальні та підходящі методи, за допомогою яких можна перевірити на псевдовипадковість.

Частотний побітовий тест.

Даний тест заснований на визначенні співвідношення між нулями та одиницями у вхідній двійковій послідовності. Його метою є з'ясування, чи має задана послідовність приблизно однакову кількість нулів та одиниць, що зазвичай притаманно випадковій двійковій послідовності. Тобто частка одиниць в послідовності має бути 0,5, як і нулів відповідно. Очевидно, що чим більш випадкова послідовність, тим ближче це відношення буде до 1. Даний тест оцінює наскільки близьким це відношення є до 1.

Для обчислення кожна «1» приймається за +1, а кожен «0» за -1. Після цього вираховуємо суму всієї послідовності за наступною формулою 2.1:

$$S_n = X_1 + X_2 + \cdots + X_n, \qquad (2.1)$$

де $Xi = 2xi — 1$.

Для прикладу візьмемо таку послідовність: 1011010101.

Тоді *S = 1 + (-1) + 1 + 1 + (-1) + 1 + (-1) + 1 + (-1) + 1 = 2.*

Обчислюємо статистику за наступним виразом:

$$S_{obs} = \frac{|S|}{\sqrt{n}} = \frac{2}{\sqrt{10}} = 0.632455532$$

Обчислюємо значення P через додаткову функцію помилок:

$$P_{value} = er\ fc\left(\frac{S_{obs}}{\sqrt{2}}\right) = er\ fc\left(\frac{0.632455532}{\sqrt{2}}\right) = 0.527089$$

Додаткова функція помилок визначається за формулою 2.2:

$$er\ fc(x) = \frac{2}{\sqrt{\pi}} \int_x^{\infty} e^{-t^2}\, dt \qquad (2.2)$$

Як бачимо, результат > 0.01, це означає, що задана послідовність пройшла тест. Даним методом рекомендується тестувати послідовності не довжиною не менше 100 біт. Після проходження цього тесту можна переходити до всіх інших.

Першим тестом, що повинна пройти двійкова псевдовипадкова послідовність є частотний побітовий тест. Суть тесту полягає у визначенні кількості одиниць, що знаходяться в блоці довжиною M біт. Аналогічно до частотного побітового тесту він повинен виявити чи дійсно кількість одиниць в блоці довжиною M приблизно рівна M/2, тобто чи діляться значення приблизно порівну.

Цей тест робиться на основі попереднього, тільки тепер значення пропорції «1»/«0» для кожного блоку аналізуються методом Хі-квадрат. Зрозуміло, що це співвідношення має бути приблизно рівним 1.

Наприклад, нехай дана послідовність 0110011010. Розіб'єм її на блоки по 3 біти (нулем вкінці можна знехтувати): 011 001 101

Порахуємо пропорції πi для кожного блоку: π1 = 2/3, π2 = 1/3, π3 = 1/3. Далі обчислюємо статистику за методом Хі-квадрат з N ступенями свободи (тут N - кількість блоків):

$$\chi^2_{obs} = 4M \sum_{i=1}^{N} (\pi_i - 1/2)^2 = 4 \cdot 3 \cdot \left[\left(\frac{2}{3} - \frac{1}{2}\right)^2 + \left(\frac{1}{3} - \frac{1}{2}\right)^2 + \left(\frac{2}{3} - \frac{1}{2}\right)^2\right] = 1$$

Обчислимо P-значення через спеціальну функцію Q:

$$P_{value} = Q\left(\frac{N}{2}, \frac{\chi^2_{obs}}{2}\right) = Q\left(\frac{3}{2}, \frac{1}{2}\right) = 0.801252$$

Q — це так звана неповна верхня гамма-функція, яка визначається формулою (2.3):

$$Q(a,x) = \frac{1}{\Gamma(a)} \int_x^\infty e^{-t} t^{a-1}\, dt \qquad (2.3)$$

При цьому функція Г – стандартна гамма-функція (формула 2.4)

$$\Gamma(z) = \int_0^\infty t^{z-1} e^{-t}\, dt \qquad (2.4)$$

Послідовність вважається випадковою, якщо P-значення >0.01. Рекомендується аналізувати послідовності довжиною не менше 100 біт, а також повинні виконуватися співвідношення M≥20, M>0.01n та N<100.

Тест на одинакові біти, що ідуть поряд

У тесті знаходяться всі послідовності однакових бітів, а потім аналізується, наскільки кількість і розміри цих послідовностей відповідають кількості та розмірам випадкової послідовності. Сенс у цьому, що якщо зміна 0 на 1 (і назад) відбувається занадто рідко, то таку послідовність не можна вважати випадковою.

Нехай дана вхідна послідовність 1001101011. Спочатку обчислюємо частку одиниць у загальній масі:

$$\pi = \frac{\sum_i X_i}{n} = \frac{6}{10}$$

Далі перевіряється умова:

$$\left|\pi - \frac{1}{2}\right| < \frac{2}{\sqrt{n}}$$

Якщо воно не задовольняє умови, тоді весь тест вважається неуспішним і на цьому все закінчується. У даному випадку 0.63246 > 0.1, отже йдемо далі.

Обчислюємо сумарне число знакозмін V за формулою (2.5):

$$V_n = \sum_{k=1}^{n-1} r(k) + 1, \quad (2.5)$$

де $r(k) = 0$ якщо $X_i = X_{i+1}$, або $r(k) = 1$ в іншому випадку.

$V_{10} = (1 + 0 + 1 + 0 + 1 + 1 + 1 + 1 + 0) + 1 = 7$

Обчислюємо P-значення через функцію помилок:

$$P_{value} = er\ fc\left(\frac{|V_n - 2n\pi(1-\pi)|}{2\sqrt{2n}\pi(1-\pi)}\right) = er\ fc\left(\frac{7 - (2 \cdot 10 \cdot \frac{3}{5} \cdot \left(1 - \frac{3}{5}\right))}{2 \cdot \sqrt{2 \cdot 10} \cdot \frac{3}{5} \cdot (1 - \frac{3}{5})}\right)$$

$$= 0.147232$$

Якщо результат ≥ 0.01 (як і в даному прикладі), то послідовність визнається випадковою.

Наступним розглянемо тест кумулятивних сум. Приймемо кожен нульовий біт вихідної послідовності за -1, а кожен одиничний — за +1, після чого порахуємо суму. Інтуїтивно зрозуміло, що чим більш «випадкова» послідовність, тим швидше ця сума буде наближатись до нуля. З іншого боку, уявімо, що дана послідовність складається зі 100 нулів і 100 одиниць, що йдуть поспіль: 00000...001111...11. Тут сума вийде рівною 0, однак очевидно, що назвати таку послідовність випадковою не можливо. Отже, необхідний глибший критерій. І цим критерієм є часткові суми. Будемо поступово рахувати суми, починаючи від першого елемента:

S1 = x1

S2 = x1 + x2 S3 = x1 + x2 + x3…

Sn = x1 + x2 + x3 +… + xn

Далі знаходиться число z = максимум серед цих сум.

Нарешті, обчислюється P-значення за такою формулою (2.6):

$$P_{value} = 1 - \Sigma_1 + \Sigma_2, \quad (2.6)$$

Тут Ф — функція розподілу стандартної нормальної випадкової величини. Стандартний нормальний розподіл — це всім відомий гаусовий

розподіл (у формі дзвона), у якого математичне очікування 0 та дисперсія 1. Виглядає так (формула 2.7):

$$\Phi(z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} e \frac{-u^2}{2} du \quad (2.7)$$

Якщо P-значення > 0.01, то послідовність визнається випадковою. До речі, у цього тесту є 2 режими: перший розглянутий вище, а у другому суми обчислюються, починаючи з останнього елемента.

Тест на непересічні шаблони, що зустрічаються.

Досліджувана послідовність розбивається блоки однакової довжини. Наприклад: 1010010010 1110010110

У кожному блоці шукається якийсь певний шаблон, наприклад «001». Слово непересічні означає, що у разі знаходження шаблону всередині послідовності, наступне порівняння не захоплюватиме жодного біта знайденого шаблону. В результаті пошуку для кожного i-го блоку буде знайдено число Wi, що дорівнює кількості знайдених випадків.

Отже, для заданих блоків W1 = 2 та W2 = 1:

101 <u>001</u> <u>001</u> 0

111 <u>001</u> 0110

Далі обчислюються математичні очікування та дисперсія, так ніби вихідна послідовність була справді випадкова. Нижче наведено формули. Тут N = 2 (кількість блоків), M = 10 (довжина блоку), m = 3 (довжина зразка).

$$\mu = \frac{M - m + 1}{2^m} = \frac{10 - 3 + 1}{2^3} = 1$$

$$\sigma^2 = M \cdot \left(\frac{1}{2^m} - \frac{2m - 1}{2^{2m}}\right) = 10 \cdot \left(\frac{1}{2^3} - \frac{2 \cdot 3 - 1}{2^{2 \cdot 3}}\right)$$

Обчислення Хі-квадрат:

$$\chi^2_{obs} = \sum_{j=1}^{N} \frac{(W_j - \sigma)^2}{\sigma^2} = \frac{(2-1)^2 + (1-1)^2}{0.46785} = \frac{1+0}{0.46785} = 2.13333$$

Обчислюється підсумкове P-значення через неповну гамма-функцію:

$$Q(a,x) = Q(\frac{N}{2}, \frac{\chi^2_{obs}}{2}) = Q(\frac{2}{2}, \frac{2.13333}{2}) = 0.344154$$

Видно, що P-значення > 0.1, отже, послідовність досить випадкова.

Проте оцінено було лише один шаблон. Насправді потрібно перевірити всі комбінації шаблонів, та ще й до того ж для різної довжини цих шаблонів. Скільки того та іншого потрібно, визначається виходячи з конкретних вимог, але зазвичай m беруть 9 або 10. Щоб отримати осмислені результати, слід брати N<100 і M>0.01*n.

2.3 Протидія прихованій передачі, шляхом перетирання полів службових заголовків

Існує не мала імовірність, що прочитати вміст виявленого стегоконтейнера не вдасться. Оскільки зловмисник може додатково підстрахуватись і зашифрувати дані за допомогою одного із методів криптографії. В цьому випадку необхідно забезпечити комп’ютерну мережу від потрапляння в її середину пакетів із шкідливим прихованим текстом.

Щоб точно не допустити проходження заповнених стегоконтейнерів без підозр можна перехоплювати пакети, що надходять в мережу. Не обов’язково перевіряти усі пакети, що надходять, оскільки у високонавантаженій мережі це може бути досить ресурсозатратно. Є сенс перевіряти лише ті пакети, в полях яких виникає певна надлишковість. Наприклад, в першому розділі було розглянуто структуру заголовку протоколу IPv4, поле «Ідентифікатор» якого мало надлишковість лише при умові, що фрагментація пакету не використовується. Тобто при виявленні прихованих даних можна не звертати уваги на пакети, де ці поля використовуються і не можуть мати ніяких інших даних, окрім потрібних для коректної роботи протоколу[16].

Може здатись, що таке втручання в роботу мережі обов’язково негативно відобразиться на звичайних користувачах. Проте можна з впевненістю сказати,

що боротьба з прихованими повідомленнями буде вестись так само непомітно для користувачів як і їх передача.

Перехоплення пакетів з прихованою інформацією та їх подальше знешкодження буде відбуватись за допомогою тих самих інструментів, що використовуються зловмисниками для атаки на мережу. Як вже згадувалось, стеганографічні інструменти є як атрибутом будь-якого успішного адміністратора мережі, так і його великою проблемою, з якою постійно потрібно боротись.

Використання програмного забезпечення для перехоплення пакетів дозволить на будь-якому проміжному пристрої проаналізувати дані, що надходять. Для запобігання потрапляння прихованих даних в мережу достатньо перехопити пакет, дублювати його поля та користувацькі дані, створити на його основі новий із заново згенерованими полями в яких може виникати надлишковість та відправити далі за вказаною адресою.

2.4 Розробка алгоритму для виявлення прихованих даних

Для того, щоб структурувати отримані знання та полегшити розробку програмного забезпечення, створюється загальний алгоритм перехоплення та перевірки мережевих пакетів на наявність вбудованих прихованих даних. При перехопленні пакетів необхідно тестувати лише ті з них, що мають поле «Ідентифікатор пакету». Це спростить завдання і дозволить перевірити більшу кількість даних з меншими затратами ресурсів.

Далі наведено повний словесний алгоритм роботи системи виявлення стеганографічного втручання:

1) початок;
2) початок захоплення мережевого трафіку;
3) отримання пакетів та даних про них(адреси та порти відправника і отримувача, протокол, корисне навантаження, час надходження тощо);

4) перевірка чи відповідає захоплений пакет умовам для його подальшого тестування, а саме відноситься до протоколу IPv4 та вдалося отримати його поле «Ідентифікатор», якщо так то перейти до пункту 4, якщо ні — повернутися до пункту 3;

5) формування двійкової послідовності у якості вхідних даних для проходження тестів на псевдовипадковість, використовуючи поле «Ідентифікатор»;

6) сформована послідовність проходить частотне побітове тестування, якщо за результатами тесту послідовність визнається близькою до справді випадкової, то перейти до пункту 7, якщо ні — перейти до пункту 11;

7) псевдовипадкова послідовність проходить тест на одинакові біти, що йдуть поряд, якщо за результатами тесту послідовність визнається близькою до істинно випадкової, то перейти до пункту 8, якщо ні — перейти до пункту 11;

8) протестована раніше послідовність проходить тест кумулятивних сум, якщо за його результатами послідовність визнається істинно випадковою, то перейти до пункту 11, якщо ні — перейти до пункту 9;

9) відмітка пакету, що не пройшов тестування як підозрілого;

10) перетворення двійкової послідовності у комфортну для перегляду форму за допомогою використання ASCII-таблиць;

11) виведення отриманих результатів для перегляду;

12) для перегляду захоплених та проаналізованих пакетів виконується зупинка захоплення мережевого трафіку, якщо так, то перейти до пункту 12, якщо ні — перейти до пункту 2;

13) закінчення захоплення мережевого трафіку;

14) фільтрація отриманих пакетів по протоколу, адресах, портах тощо.

15) перегляд та аналіз захоплених пакетів оператором;

16) кінець.

Для легшого та комфортнішого читання алгоритму розроблено його графічну схему, яку можна переглянути в додатку Д.

3 РЕАЛІЗАЦІЯ МЕТОДУ ВИЯВЛЕННЯ ПРИХОВАНИХ ДАНИХ

Для розробки програмного забезпечення призначеного для захоплення та тестування мережевого трафіку було обрано засоби платформи .NET Framework, бібліотека PcapDotNet для зручної роботи з пакетами, а також набір тестів NIST написаних мовою C++. Для створення користувацького інтерфейсу використовується графічна підсистема WPF.

3.1 Проектування та розробка графічного інтерфейсу

Для початку необхідно розбити робочу область вікна на чотири блоки:

— командний блок, з полями для вибору фільтрів та основними керуючими елементами;

— блок відображення перехоплених пакетів та їх даних у вигляді таблиці;

— блок відображення детальніших даних одного із обраних пакетів в таблиці;

— блок, що містить тестову інформацію про кількість захоплених пакетів та загальний їх розмір.

Створення користувацького інтерфейсу в середовищі WPF відбувається за допомогою декларативної мови розмітки XAML. Вона була створена спеціально для того, щоб спростити процес розробки користувацького інтерфейсу.

Тож, створення вікна програми починається з розмітки сторінки та розбиття на блоки (лістинг 3.1).

Лістинг 3.1 — Розмітка основного вікна програми

```
<Grid Background="#00001a">
    <Grid.RowDefinitions>
        <RowDefinition Height="3*"/>
        <RowDefinition Height="20*"/>
        <RowDefinition Height="12*"/>
        <RowDefinition Height="2*"/>
```

```
</Grid.RowDefinitions>
```

Далі буде йти розмітка командного блоку з елементами керування. В ньому створюються кнопки для керування початком та зупинкою захоплення мережевих пакетів, очищенням таблиці з отриманими даними, фільтрацією пакетів та показом усіх знайдених пакетів. Також там буде розміщено поле для вводу, яке можна використовувати для пошуку пакетів за адресами та портами (лістинг 3.2).

Лістинг 3.2 — Розмітка керуючого блоку програми

```
<Grid Grid.Row="0">
<CheckBox x:Name="filterCheckBox" Grid.Column="0"
VerticalAlignment="Center" Content="Filtering"/>
<TextBox x:Name="ipTextBox" Grid.Column="1" Height="30"/>
<ComboBox x:Name="typeComboBox" Grid.Column="2" Height="30"
Width="120" Text="Types"/>
<Button x:Name="filterButton" Grid.Column="3" Height="30" Width="100"
Content="Filter" Click="filterButton_Click"/>
<Button x:Name="startButton" Grid.Column="4" Height="30" Width="100"
Content="Start" Click="startButton_Click"/>
<Button x:Name="stopButton" Grid.Column="5" Height="30" Width="100"
Content="Stop" Click="stopButton_Click"/>
<Button x:Name="clearButton" Grid.Column="6" Height="30" Width="100"
Content="Clear" Click="clearButton_Click"
MouseDoubleClick="clearButton_DoubleClick"/>
 <Button x:Name="allButton" Grid.Column="7" Height="30" Width="100"
Content="Show all sniffed" Click="allButton_Click"/>
  </Grid>
```

Під блоком керування розміщується таблиця для відображення результатів перехоплення мережевих пакетів. Щоб організувати подібну структуру буде використано елемент інтерфейсу DataGrid, який дозволяє зручно оперувати та відображати колекції даних.

В таблиці буде створено 8 колонок:

— адреса відправника;

— адреса отримувача;

— порт відправника;

— порт отримувача;

— протокол;

— дата отримання пакету;

— загальна довжина;

— вміст поля «Ідентифікатор» (якщо воно містить дані, що не були згенеровані псевдовипадково).

Мова розмітки дозволяє прив’язатись до полів об’єктів колекції, що дасть змогу оновлювати дані в режимі реального часу (лістинг 3.3).

Лістинг 3.3 — Створення таблиці для відображення пакетів

```
<DataGrid x:Name="dataGrid" Margin="10" Grid.Row="1"
ColumnHeaderHeight="40" RowHeight="36" SelectionMode="Single"
FontSize="15" CanUserResizeRows="False" CanUserResizeColumns="False"
IsReadOnly="True" CanUserAddRows="False" CanUserDeleteRows="False"
AutoGenerateColumns="False" VerticalGridLinesBrush="Transparent"
SelectionChanged="dataGrid_SelectionChanged">
        <DataGrid.Columns>
            <DataGridTextColumn  Header="Source address" Binding="{Binding
SourceIP}" Width="5*" />
            <DataGridTextColumn  Header="Source port" Binding="{Binding
SourcePort}" Width="5*"/>
            <DataGridTextColumn Header="Destination address" Binding="{Binding
DestinationIP}" Width="5*"/>
            <DataGridTextColumn Header="Destination port" Binding="{Binding
DestinationPort}" Width="5*"/>
            <DataGridTextColumn Header="Type" Binding="{Binding Type}"
Width="5*"/>
```

```
            <DataGridTextColumn Header="Time" Binding="{Binding Timestamp}"
Width="5*"/>
            <DataGridTextColumn Header="Total length" Binding="{Binding
TotalLength}" Width="5*"/>
            <DataGridTextColumn Header="Identification" Binding="{Binding
Content}" Width="5*"/>
          </DataGrid.Columns>
        </DataGrid>
```

Далі іде блок з додатковою інформацією, в якій буде розміщуватись вміст пакету, якщо такий є, у вигляді символів та у шістнадцятковому варіанті (лістинг 3.4).

Лістинг 3.4 — Текстові поля для відображення вмісту пакетів

```
<TextBlock x:Name="charTextBox" Grid.Column="0" Background="White"
Margin="10"/>
  <TextBlock x:Name="hexTextBox" Grid.Column="1"
Background="WhiteSmoke" Margin="10"/>
```

Останнім іде поле з інформацією про кількість отриманих пакетів та їх загальний розмір (лістинг 3.5).

Лістинг 3.5 — Текстове поле з кількістю отриманих пакетів

```
<TextBlock x:Name="hintLabel" Grid.Row="3" Text="Packets received: 0 Total
length: [0 bytes]" VerticalAlignment="Center" FontSize="16" Foreground="White"/>
```

Макет згенерований в результаті розмітки можна переглянути в додатку Е.

3.2 Розробка моделі перехоплення пакетів та виявлення прихованих даних

Спочатку створюється клас Packet.cs, який представляє структуру мережевого пакета з його заголовками та вмістом. Він містить поля, які відповідають пакета. В конструкторі класу відбувається конвертування отриманого масиву байтів у відповідні поля заголовків.

Далі створюється клас Monitor.cs, що відповідає за створення сокету, та перехоплення пакетів, що курсують по локальній комп'ютерній мережі. За створення та ініціалізацію сокету відповідає метод Start() (лістинг 3.6).

Лістинг 3.6 — Створення та ініціалізація сокету

```
public void start(){
        if (monitor_Socket == null){
           try{
              if (ipAddress.AddressFamily == AddressFamily.InterNetwork) {
                 monitor_Socket = new Socket(AddressFamily.InterNetwork,
 SocketType.Raw, System.Net.Sockets.ProtocolType.IP); }
              else{
                 monitor_Socket = new Socket(AddressFamily.InterNetworkV6,
 SocketType.Raw, System.Net.Sockets.ProtocolType.IP);}
              monitor_Socket.Bind(new IPEndPoint(ipAddress, 0));
              monitor_Socket.IOControl(SIO_RCVALL,
 BitConverter.GetBytes((int)1), null);
              monitor_Socket.BeginReceive(buffer, 0, buffer.Length,
 SocketFlags.None, new AsyncCallback(this.OnReceive), null); }
           catch (Exception e) {
              monitor_Socket.Close();
              monitor_Socket = null;
```

За його закриття відповідає метод Stop().

За отримання пакетів та їх аналіз відповідає метод OnReceive(), який приймає масив байтів та конвертує в зручний для роботи об'єкт. Звідси отримується поле «Ідентифікатор» та перетворюється в двійкову послідовність, яка потім проходить тести на псевдовипадковість (лістинг 3.7).

Лістинг 3.7 — Отримання та аналіз пакету

```
int len = monitor_Socket.EndReceive(ar);
           if (monitor_Socket != null)
```

```
            {
                byte[] receivedBuffer = new byte[len];
                Array.Copy(buffer, 0, receivedBuffer, 0, len);
                    Packet packet = new Packet(receivedBuffer);
                    PcapDotNet.Packets.Packet packetNew = new
PcapDotNet.Packets.Packet(receivedBuffer, DateTime.Now,
PcapDotNet.Packets.DataLinkKind.IpV4);
                        If(CheckTests(packet)) OnNewPacket(packet);
```

Створення нового об’єкту для моніторінгу, що відповідає за прослуховування мережі, відбувається методом StartRaking(). (лістинг 3.8)

```
private void startRaking(){
        monitorList.Clear();
        IPAddress[] hosts =
Dns.GetHostEntry(Dns.GetHostName()).AddressList;
        if (hosts == null || hosts.Length == 0){
            MessageBox.Show("No hosts detected, check your network!");
        }
        for (int i = 0; i < hosts.Length; i++) {
            Monitor monitor = new Monitor(hosts[i]);
            monitor.newPacketEventHandler += new
Monitor.NewPacketEventHandler(onNewPacket);
            monitorList.Add(monitor);
        }
        foreach (Monitor monitor in monitorList)
        {
            monitor.start();
```

4 ІНСТРУКЦІЯ КОРИСТУВАЧА

При запускові програми перед користувачем відкривається головне вікно, яке зображено на рисунку 4.1.

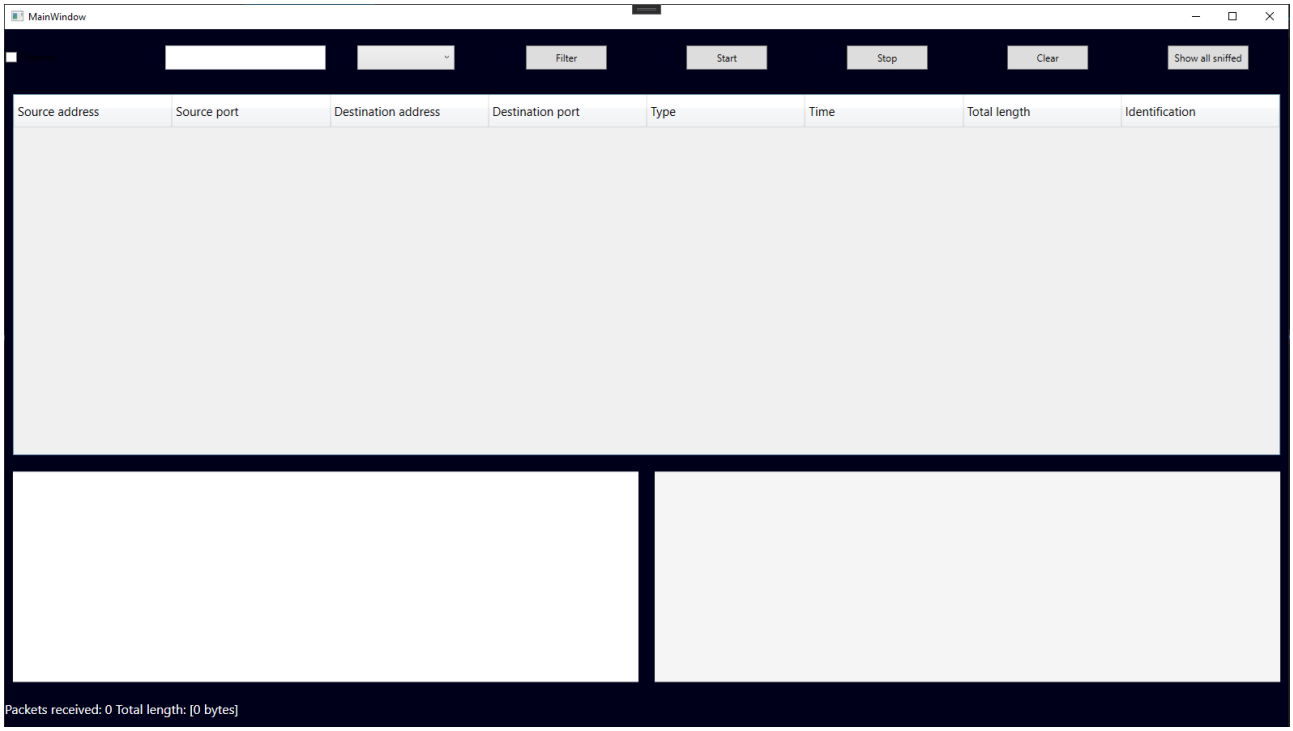

Рисунок 4.1 — Запуск програми

У верхній його частині знаходяться кнопки для керування процесом захоплення пакетів, їх фільтрації та очищення.

При натисненні кнопки «Start» починається захоплення пакетів, які в даний момент проходять по мережі. Переглянути їх кількість та загальний розмір можна в нижній лівій частині екрану. Детальніша інформація щодо захоплених пакетів знаходиться в таблиці та дає можливість переглянути адресу відправника, адресу отримувача, порт відправника, порт отримувача, протокол, розмір та поле ідентифікатора (якщо воно не є псевдовипадковим) (рис. 4.2).

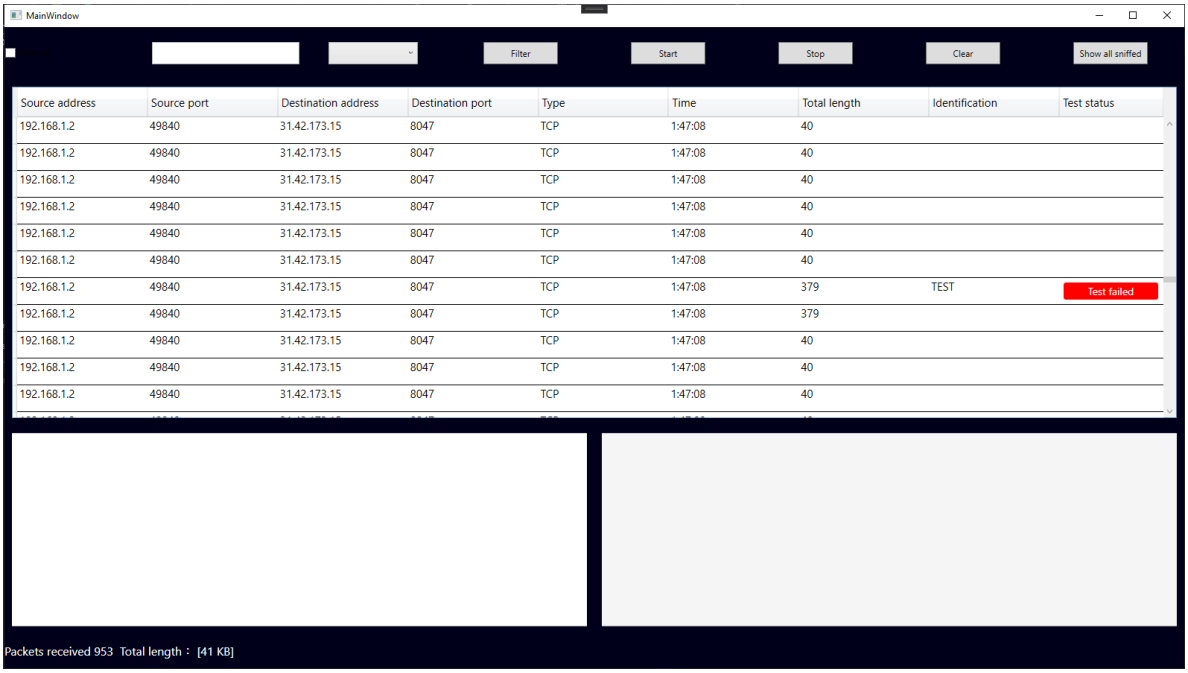

Рисунок 4.2 — Процес захоплення пакетів

При натисненні кнопки «Stop» можна переглянути дані пакетів, що були захоплені. Як можна помітити на рисунку 4.2, пакети що не пройшли тестування на псевдовипадковість полів будуть відмічені відповідною плашкою. На них потрібно звернути увагу. Також їм можна проаналізувати. Якщо обрати з таблиці якийсь елемент, то можна отримати певну інформацію про корисне навантаження пакету (рис 4.3).

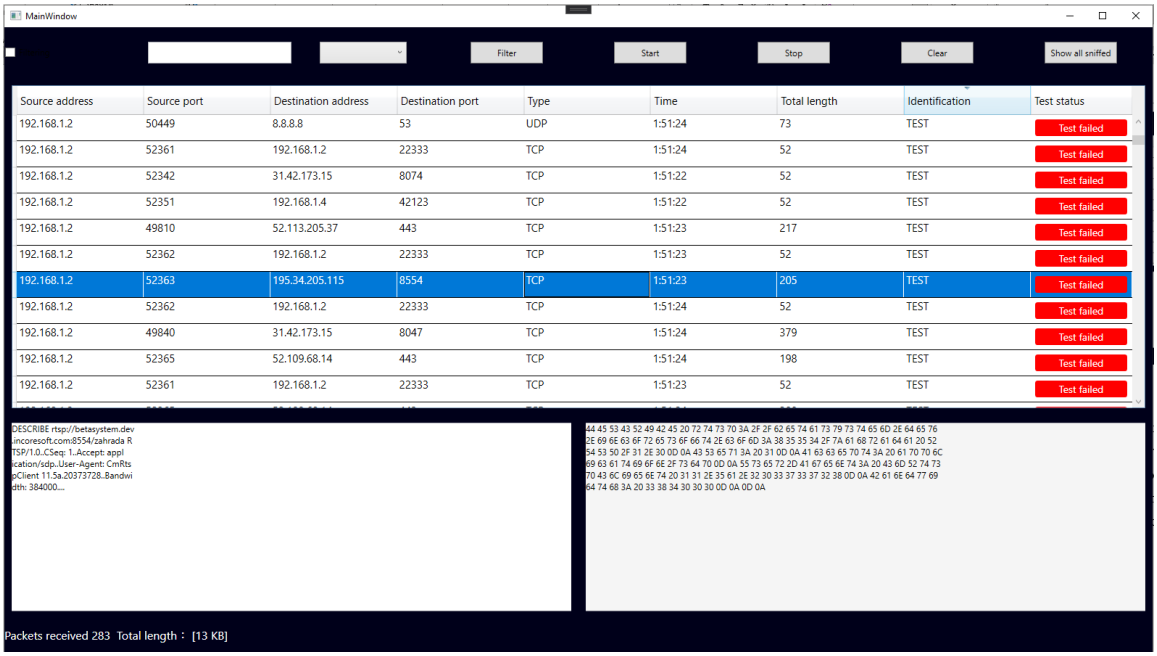

Рисунок 4.3 — Дослідження пакету

Текстове поле та випадаюче меню необхідні для фільтрації пакетів по отриманим результатам. При натисненні кнопки «Filter» буде відфільтровано всі пакети, наприклад, по адресі та протоколу (рис 4.4).

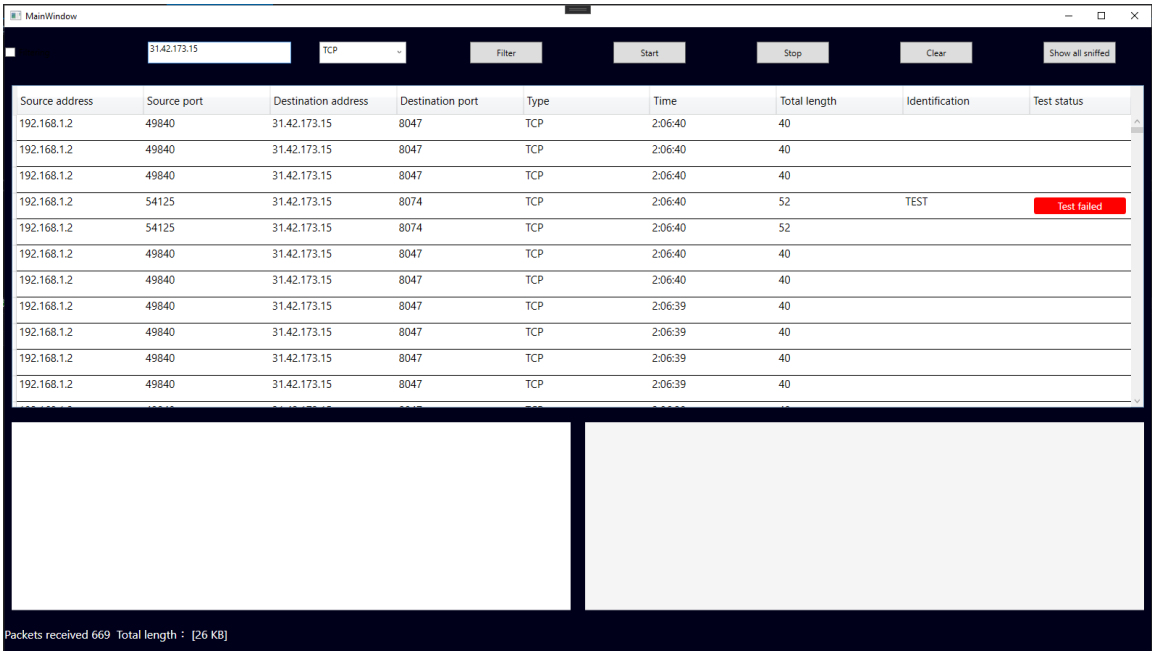

Рисунок 4.4 — Фільтрація

Для скидання фільтрів необхідно натиснути кнопку «Show all sniffed», а для очистки таблиці з результатами пошуку достатньо натиснути кнопку «Clear».

Для проведення тестування в локальній мережі на іншій машині було запущено програмне забезпечення, що вбудовує приховану інформацію в мережеві пакети протоколу IPv4. Оскільки для передачі в полі ідентифікатора пакету доступно лише 19 біт, то для тестування оберемо повідомлення, яке не буде перевищувати 4 байт. Повідомлення такого розміру буде складатись з 4 букв відповідно. З тестового ПК буде відправлятись повідомлення «TEST» закодоване за допомогою таблиць ASCII.

5 ЕКОНОМІЧНА ЧАСТИНА

Науково-технічна розробка має право на існування та впровадження, якщо вона відповідає вимогам часу, як в напрямку науково-технічного прогресу та і в плані економіки. Тому для науково-дослідної роботи необхідно оцінювати економічну ефективність результатів виконаної роботи.

Магістерська кваліфікаційна робота «Методи виявлення передавання прихованої інформації в службових заголовках протокольних блоків даних комп'ютерних мереж» відноситься до науково-технічних робіт, які орієнтовані на виведення на ринок (або рішення про виведення науково-технічної розробки на ринок може бути прийнято у процесі проведення самої роботи), тобто коли відбувається так звана комерціалізація науково-технічної розробки. Цей напрямок є пріоритетним, оскільки результатами розробки можуть користуватися інші споживачі, отримуючи при цьому певний економічний ефект. Але для цього потрібно знайти потенційного інвестора, який би взявся за реалізацію цього проекту і переконати його в економічній доцільності такого кроку.

Для наведеного випадку нами мають бути виконані такі етапи робіт:

— проведено комерційний аудит науково-технічної розробки, тобто встановлення її науково-технічного рівня та комерційного потенціалу;

— розраховано витрати на здійснення науково-технічної розробки;

— розрахована економічна ефективність науково-технічної розробки у випадку її впровадження і комерціалізації потенційним інвестором і проведено обґрунтування економічної доцільності комерціалізації потенційним інвестором.

5.1 Проведення комерційного та технологічного аудиту науково-технічної розробки

Метою проведення комерційного і технологічного аудиту дослідження за темою «Методи виявлення передавання прихованої інформації в службових заголовках протокольних блоків даних комп'ютерних мереж» є оцінювання

науково-технічного рівня та рівня комерційного потенціалу розробки, створеної в результаті науково-технічної діяльності.

Оцінювання науково-технічного рівня розробки та її комерційного потенціалу рекомендується здійснювати із застосуванням 5-ти бальної системи оцінювання за 12-ма критеріями, наведеними в табл. 5.1 [17]

Таблиця 5.1 — Рекомендовані критерії оцінювання науково-технічного рівня і комерційного потенціалу розробки та бальна оцінка

Бали (за 5-ти бальною шкалою)					
	0	1	2	3	4
Технічна здійсненність концепції					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено працездатність продукту в реальних умовах
Ринкові переваги (недоліки)					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів
4	Технічні та споживчі властивості продукту значно гірші, ніж в аналогів	Технічні та споживчі влас-тивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в аналогів	Технічні та споживчі властивості продукту значно кращі, ніж в аналогів
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
Ринкові перспективи					
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає

Закінчення таблиці 5.1

Практична здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуютьс я у виробництві
11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь- які регламентні обмеження на виробництво та реалізацію продукту

Результати оцінювання науково-технічного рівня та комерційного потенціалу науково-технічної розробки потрібно звести до таблиці.

Таблиця 5.2 — Результати оцінювання науково-технічного рівня і комерційного потенціалу розробки експертами

Критерії	Експерт (ПІБ, посада)		
	1	2	3
	Бали:		
1. Технічна здійсненність концепції	4	5	4
2. Ринкові переваги (наявність аналогів)	3	2	3
3. Ринкові переваги (ціна продукту)	3	4	3
4. Ринкові переваги (технічні властивості)	4	4	4
5. Ринкові переваги (експлуатаційні витрати)	0	0	0
6. Ринкові перспективи (розмір ринку)	3	4	4
7. Ринкові перспективи (конкуренція)	3	3	3
8. Практична здійсненність (наявність фахівців)	4	4	4
9. Практична здійсненність (наявність фінансів)	3	3	2
10. Практична здійсненність (необхідність нових матеріалів)	4	4	4
11. Практична здійсненність (термін реалізації)	4	3	4
12. Практична здійсненність (розробка документів)	3	3	3
Сума балів	38	39	38
Середньоарифметична сума балів $СБ_c$	38,3		

За результатами розрахунків, наведених в таблиці 4.2, зробимо висновок щодо науково-технічного рівня і рівня комерційного потенціалу розробки. При цьому використаємо рекомендації, наведені в табл. 4.3 [17]

Таблиця 5.3 — Науково-технічні рівні та комерційні потенціали розробки

Середньоарифметична сума балів СБ, розрахована на основі висновків експертів	Науково-технічний рівень та комерційний потенціал розробки
41…48	Високий
31…40	Вище середнього
21…30	Середній
11…20	Нижче середнього
0…10	Низький

Згідно проведених досліджень рівень комерційного потенціалу розробки за темою «Методи виявлення передавання прихованої інформації в службових заголовках протокольних блоків даних комп'ютерних мереж» становить 38,3 бала, що, відповідно до таблиці 4.3, свідчить про комерційну важливість проведення даних досліджень (рівень комерційного потенціалу розробки вище середнього).

5.2 Оцінювання рівня новизни розробки

Виводячи на ринок новинку виробник вважає, що тієї новизни, якою наділена нова розробка є достатньо для того, щоб вона була сприйнята споживачем як нова. Але це не завжди так, в силу того, що споживач і виробник неоднозначно визначать її рівень новизни. Тому доцільним є визначення рівня новизни розробки отриманої в результаті досліджень за темою «Методи виявлення передавання прихованої інформації в службових заголовках протокольних блоків даних комп'ютерних мереж».

Саме визначення рівня і ступеня інтегральної новизни є найбільш актуальним, оскільки її рівень визначає ступінь однакового позитивного сприйняття новизни розробки як виробником, так і споживачем, а отже і ринком в цілому, а це, у свою чергу, є гарантією того, що новинка знайде своє місце на ринку, користуватиметься попитом у споживачів і забезпечить відшкодування витрат, зазнаних товаровиробником під час розроблення та виробництва технічної розробки [18]

Рівень новизни нової продукції розраховуємо експертним методом шляхом протиставлення нової продукції та її аналогів, що існують в даний час на ринку, за чинниками що визначають її значення, в системі «краще-гірше». Рівень новизни встановлюємо відносно рівня аналога (або продукту, що досить близький до аналога).

Для визначення *i*-го виду новизни, застосуємо чинники, які впливають на її рівень. Кожен чинник *i*-го виду новизни розраховуємо в балах. Більша кількість

набраних балів свідчить про більший рівень новизни. Для оцінювання рівня новизни використаємо думки експертів, які встановлюють визначені бали відповідним чинникам. Бал відповідності проставляється в діапазоні від (-5 — значно гірше аналога до +5 — значно краще аналога). Результати попереднього оцінювання зведемо до відповідного листа оцінювання (таблиця 5.4).

Таблиця 5.4 — Лист оцінювання рівня новизни експертами

Види та чинники		Бали та експерти		
		Експерт 1	Експерт 2	Експерт 3
1		*2*	*3*	*4*
Споживча новизна	Питома вага 0,225	Максимальний бал $Б_{i\ max}$		25
1. Зміна поведінкових звичок споживача		3	3	3
2. Ступінь задоволення потреб і запитів		2	2	2
3. Спосіб задоволення потреби		2	2	2
4. Формування нової потреби		0	0	0
5. Формування нового споживача		0	0	0
Середній бал експертів $Б_{i\ отр}$		7		
Товарна новизна	Питома вага 0,217	Максимальний бал $Б_{i\ max}$		30
1. Параметричні зміни показників продукції				
1.1. Якісні		3	3	3
1.2. Технічні		3	4	3
1.3. Економічні		2	3	2
1.4. Сервісні		4	4	4
2. Якість продукції по відношенню до конкурентів		3	3	3
3. Функціональні зміни		3	3	3
Середній бал експертів $Б_{i\ отр}$		19		
Виробнича новизна	Питома вага 0,042	Максимальний бал $Б_{i\ max}$		25
1. Рівень унікальності товару для підприємства		5	5	5
2. Рівень унікальності для галузі		2	3	2
3. Рівень унікальності товару для країни		0	0	0
4. Зміна виробничої системи		4	4	4
5. Відносно існуючого асортименту		2	2	2
Середній бал експертів $Б_{i\ отр}$		13		
Прогресивна новизна	Питома вага 0,179	Максимальний бал $Б_{i\ max}$		25
1. Зміна технології виготовлення		4	4	4
2. Рівень застосування нових компонентів і матеріалів		1	2	1
3. Зміна технологічного принципу дії виробу		1	2	1
4. Зміна конструктивного виконання		3	2	3

Закінчення таблиці 5.4

5. Рівень застосування інновацій		2	2	2
Середній бал експертів $Б_{i\ отр}$		11		
Ринкова новизна	Питома вага 0,12	Максимальний бал $Б_{i\ MAX}$		20
1. Новий виріб на новому ринку		0	0	0
2. Новий виріб на відомому ринку		2	2	2
3. Модернізований виріб		2	2	2
4. Нова модель		1	2	2
Середній бал експертів $Б_{i\ отр}$		6		
Екологічна новизна	Питома вага 0,035	Максимальний бал $Б_{i\ MAX}$		20
1. Рівень екологічної чистоти технології виробництва		5	5	5
2. Рівень впровадження мало- та безвідходних технологій		5	5	5
3. Рівень екологічно небезпечних режимів експлуатації продукції		5	5	5
4. Рівень забруднення навколишнього середовища		5	5	5
Середній бал експертів $Б_{i\ отр}$		20		
Соціальна новизна	Питома вага 0,036	Максимальний бал $Б_{i\ MAX}$		20
1. Використання нового товару приводить до покращення стану здоров'я нації		0	0	0
2. Використання нового товару приводить до зростання доходів населення		0	0	0
3. Виробництво нового товару приводить до збільшення (зменшення) кількості робочих місць на підприємстві		4	5	4
4. Виробництво нового товару приводить до підвищення кваліфікації персоналу		3	3	3
Середній бал експертів $Б_{i\ отр}$		7		
Маркетингова новизна	Питома вага 0,146	Максимальний бал $Б_{i\ MAX}$		20
1. Нові методи маркетингових досліджень		0	0	0
2. Вживання нових стратегій сегментації ринку		1	1	1
3. Вибір нової маркетингової стратегії обхвату і розвитку цільового сегмента		2	3	2
4. Побудова нових каналів збуту		2	2	2
Середній бал експертів $Б_{i\ отр}$		5		

Значення *i*-го виду новизни розрахуємо за формулою [18]:

$$I_i = \frac{Б_{i\,отр}}{Б_{i\,MAX}}, \qquad (5.1)$$

де $Б_{i\ отр}$ — отримана кількість балів за шкалою оцінок чинників, що визначають i-й вид новизни;

$Б_{i\ max}$ — максимальна кількість балів, що може бути отримана за i-м видом новизни.

Загальний рівень інтегральної новизни розраховуємо шляхом перемноження отриманого значення i-го виду новизни на її вагомість, причому вагомість i-го виду новизни визначаємо експертним методом, за формулою [18]:

$$N_{інт} = \sum_{i}^{n} W_i \cdot I_i \ , \tag{5.2}$$

де $N_{інт}$ — рівень інтегральної (сукупної) новизни;

W_i — вагомість (питома вага) i-го виду новизни;

n — загальна кількість видів новизни.

$$N_{інт} = (0{,}225 \cdot 7/25)+(0{,}217 \cdot 19/30)+(0{,}042 \cdot 13/25)+(0{,}179 \cdot 11/25)+(0{,}12 \cdot 6/20)+$$
$$(0{,}035 \cdot 20/20)+(0{,}036 \cdot 7/20)+(0{,}146 \cdot 5/20) = 0{,}423.$$

Отримане значення інтегрального рівня новизни зіставляємо зі шкалою, що наведена в табл. 5.5 [17].

Таблиця 5.5 — Рівні новизни нового товару та їхня характеристика

Рівні новизни товару	Значення інтегральної новизни	Характеристика товару	Вид нового товару
Найвища	1,00	Абсолютно новий товар	Новий товар, що наділений ознаками інноваційності (інноваційний товар)
Висока	0,8…0,99	Товар, який не має аналогів	
Значуща	0,6…0,79	Принципова зміна споживчих властивостей товару	
Достатня	0,4…0,59	Принципова технологічна модифікація товару	
Незначна	0,2…0,39	Кардинальна зміна параметрів	Новий товар
Помилкова	0,00…0,19	Малоістотна модифікація	

Згідно таблиці 5.5 розробка відповідає рівню при значенні інтегральної новизни 0,423 — достатня новизна; за характеристикою: принципова програмна модифікація програмного забезпечення; вид розробки - новий товар, що наділений ознаками інноваційності (інноваційний товар).

5.3 Розрахунок витрат на проведення науково-дослідної роботи

Витрати, пов’язані з проведенням науково-дослідної роботи на тему «Методи виявлення передавання прихованої інформації в службових заголовках протокольних блоків даних комп'ютерних мереж», під час планування, обліку і калькулювання собівартості науково-дослідної роботи групуємо за відповідними статтями.

5.3.1 Витрати на оплату праці

До статті «Витрати на оплату праці» належать витрати на виплату основної та додаткової заробітної плати керівникам відділів, лабораторій, секторів і груп, науковим, інженерно-технічним працівникам, конструкторам, технологам, креслярам, копіювальникам, лаборантам, робітникам, студентам, аспірантам та іншим працівникам, безпосередньо зайнятим виконанням конкретної теми, обчисленої за посадовими окладами, відрядними розцінками, тарифними ставками згідно з чинними в організаціях системами оплати праці.

Основна заробітна плата дослідників

Витрати на основну заробітну плату дослідників ($З_о$) розраховуємо у відповідності до посадових окладів працівників, за формулою [17]:

$$З_о = \sum_{i=1}^{k} \frac{M_{ni} \cdot t_i}{T_p}, \qquad (5.3)$$

де k — кількість посад дослідників залучених до процесу досліджень;

M_{ni} — місячний посадовий оклад конкретного дослідника, грн;

t_i — число днів роботи конкретного дослідника, дн.;

T_p — середнє число робочих днів в місяці, T_p=24 дні.

$$З_о = 13750{,}00 \cdot 28 / 24 = 16041{,}67 \; грн.$$

Проведені розрахунки зведемо до таблиці.

Таблиця 5.6 — Витрати на заробітну плату дослідників

Найменування посади	Місячний посадовий оклад, грн	Оплата за робочий день, грн	Число днів роботи	Витрати на заробітну плату, грн
Керівник проекту	13750,00	572,92	28	16041,67
Інженер-розробник програмного забезпечення	12200,00	508,33	21	10675,00
Консультант (адміністратор комп'ютерної мережі)	11720,00	488,33	9	4395,00
Лаборант	6500,00	270,83	15	4062,50
Всього				35174,17

Основна заробітна плата робітників

Витрати на основну заробітну плату робітників ($З_р$) за відповідними найменуваннями робіт НДР на тему «Методи виявлення передавання прихованої інформації в службових заголовках протокольних блоків даних комп'ютерних мереж» розраховуємо за формулою:

$$З_р = \sum_{i=1}^{n} C_i \cdot t_i \,, \qquad (5.4)$$

де C_i — погодинна тарифна ставка робітника відповідного розряду, за виконану відповідну роботу, грн/год;

t_i — час роботи робітника при виконанні визначеної роботи, год.

Погодинну тарифну ставку робітника відповідного розряду C_i можна визначити за формулою:

$$C_i = \frac{M_M \cdot K_i \cdot K_c}{T_p \cdot t_{зм}}, \quad (5.5)$$

де M_M — розмір прожиткового мінімуму працездатної особи, або мінімальної місячної заробітної плати (в залежності від діючого законодавства), приймемо *Мм*=2379,00 грн;

K_i — коефіцієнт міжкваліфікаційного співвідношення для встановлення тарифної ставки робітнику відповідного розряду (табл. Б.2, додаток Б) [18];

K_c — мінімальний коефіцієнт співвідношень місячних тарифних ставок робітників першого розряду з нормальними умовами праці виробничих об'єднань і підприємств до законодавчо встановленого розміру мінімальної заробітної плати.

T_p — середнє число робочих днів в місяці, приблизно T_p = 24 дн;

$t_{зм}$ — тривалість зміни, год.

$C_1 = 2379{,}00 \cdot 1{,}10 \cdot 1{,}65 / (24 \cdot 8) = 22{,}49$ *грн.*
$З_{р1} = 22{,}49 \cdot 6{,}45 = 145{,}05$ *грн.*

Таблиця 5.7 — Величина витрат на основну заробітну плату робітників

Найменування робіт	Тривалість роботи, год	Розряд роботи	Тарифний коефіцієнт	Погодинна тарифна ставка, грн	Величина оплати на робітника грн
Підготовка обладнання досліджуваної комп'ютерної мережі	6,45	2	1,10	22,49	145,05
Підготовка комп'ютерного обладнання розробника програмного забезпечення	8,00	3	1,35	27,60	220,80
Монтаж мережі	6,20	4	1,50	30,67	190,13

Закінчення таблиці 5.7

Інсталяція програмного забезпечення для моделювання інформаційних потоків	4,32	4	1,50	30,67	132,48
Компіляція програмних блоків	7,50	5	1,70	34,76	260,67
Налагодження програмних блоків	4,50	4	1,50	30,67	138,00
Редагування протоколів	7,00	4	1,50	30,67	214,67
Тестування програмного забезпечення	6,00	2	1,10	22,49	134,93
Всього					1436,74

Додаткову заробітну плату робітників розраховуємо як 10 … 12% від суми основної заробітної плати дослідників та робітників за формулою:

$$З_{дод} = (З_о + З_р) \cdot \frac{Н_{дод}}{100\%}, \quad (5.6)$$

де $Н_{дод}$ — норма нарахування додаткової заробітної плати. Приймемо 11%.

$З_{дод}$ = *(35174,17 + 1436,74) · 11 / 100% = 4027,20 грн.*

5.3.2 Відрахування на соціальні заходи

Нарахування на заробітну плату дослідників та робітників розраховуємо як 22% від суми основної та додаткової заробітної плати дослідників і робітників за формулою:

$$З_н = (З_о + З_р + З_{дод}) \cdot \frac{Н_{зп}}{100\%} \quad (5.7)$$

де $Н_{зп}$ — норма нарахування на заробітну плату. Приймаємо 22%.

Зн = (35174,17 + 1436,74 + 4027,20) · 22 / 100% = 8940,38 грн.

5.3.3 Сировина та матеріали

До статті «Сировина та матеріали» належать витрати на сировину, основні та допоміжні матеріали, інструменти, пристрої та інші засоби і предмети праці, які придбані у сторонніх підприємств, установ і організацій та витрачені на проведення досліджень за темою «Методи виявлення передавання прихованої інформації в службових заголовках протокольних блоків даних комп'ютерних мереж».

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

$$М = \sum_{j=1}^{n} Н_j \cdot Ц_j \cdot К_j - \sum_{j=1}^{n} В_j \cdot Ц_{вj}, \quad (5.8)$$

де $Н_j$ — норма витрат матеріалу j-го найменування, кг;

n — кількість видів матеріалів;

$Ц_j$ — вартість матеріалу j-го найменування, грн/кг;

$К_j$ — коефіцієнт транспортних витрат, ($К_j$ = *1,1 ... 1,15*);

$В_j$ — маса відходів j-го найменування, кг;

$Ц_{вj}$ — вартість відходів j-го найменування, грн/кг.

$М_1$ = 4,00 · 108,00 · 1,1 - 0,000 · 0,00 = 475,20 грн.

Проведені розрахунки зведемо до таблиці.

Таблиця 5.8 — Витрати на матеріали

Найменування матеріалу, марка, тип, сорт	Ціна за 1 кг, грн	Норма витрат, кг	Величина відходів, кг	Ціна відходів, грн/кг	Вартість витраченого матеріалу, грн
Офісний папір Cristal A4 500	108,00	4,00	0,000	0,00	475,20

Закінчення таблиці 5.8

Папір для записів Cristal LightPapers 65 A5	55,00	3,00	0,000	0,00	181,50
Органайзер офісний Cristal	175,00	4,00	0,000	0,00	770,00
Канцелярське приладдя	78,00	4,00	0,000	0,00	343,20
Картридж для принтера Canon LBP5000	960,00	1,00	0,000	0,00	1056,00
Диск оптичний Vybir CD-R	13,25	3,00	0,000	0,00	43,73
Flesh-пам'ять Kingston 32 GB	234,00	1,00	0,000	0,00	257,40
Тека для паперів PapersBOX-2	100,00	3,00	0,000	0,00	330,00
Всього					3457,03

5.3.4 Розрахунок витрат на комплектуючі

Витрати на комплектуючі *($К_в$)*, які використовують при проведенні НДР на тему «Методи виявлення передавання прихованої інформації в службових заголовках протокольних блоків даних комп'ютерних мереж», розраховуємо, згідно з їхньою номенклатурою, за формулою:

$$К_в = \sum_{j=1}^{n} Н_j \cdot Ц_j \cdot К_j \qquad (5.9)$$

де $Н_j$ — кількість комплектуючих *j*-го виду, шт.;

$Ц_j$ — покупна ціна комплектуючих *j*-го виду, грн;

$К_j$ — коефіцієнт транспортних витрат, ($К_j$ *= 1,1 ... 1,15*).

$К_в = 1 \cdot 15830{,}00 \cdot 1{,}1 = 17413{,}00$ *грн.*

Проведені розрахунки зведемо до таблиці.

Таблиця 5.9 — Витрати на комплектуючі

Найменування комплектуючих	Кількість, шт.	Ціна за штуку, грн	Сума, грн
Серверна станція	1	15830,00	17413,00
Маршрутизатор	1	1240,00	1364,00
Мережевий адаптер	2	520,00	1144,00
Всього			19921,00

5.3.5 Спецустаткування для наукових (експериментальних) робіт

До статті «Спецустаткування для наукових (експериментальних) робіт» належать витрати на виготовлення та придбання спецустаткування необхідного для проведення досліджень, також витрати на їх проектування, виготовлення, транспортування, монтаж та встановлення. Спецустаткування для наукових (експериментальних) робіт відсутнє.

5.3.6 Програмне забезпечення для наукових (експериментальних) робіт

До статті «Програмне забезпечення для наукових (експериментальних) робіт» належать витрати на розробку та придбання спеціальних програмних засобів і програмного забезпечення, (програм, алгоритмів, баз даних) необхідних для проведення досліджень, також витрати на їх проектування, формування та встановлення.

Балансову вартість програмного забезпечення розраховуємо за формулою:

$$B_{прг} = \sum_{i=1}^{k} Ц_{iпрг} \cdot C_{прг.i} \cdot K_i \ , \qquad (5.10)$$

де $Ц_{iпрг}$ — ціна придбання одиниці програмного засобу даного виду, грн;

$C_{прг.i}$ — кількість одиниць програмного забезпечення відповідного найменування, які придбані для проведення досліджень, шт.;

K_i – коефіцієнт, що враховує інсталяцію, налагодження програмного засобу тощо, ($K_i = 1,10…1,12$);

k — кількість найменувань програмних засобів.

$B_{прг} = 14296{,}00 \cdot 1 \cdot 1{,}1 = 15725{,}60$ *грн.*

Отримані результати зведемо до таблиці:

Таблиця 5.10 — Витрати на придбання програмних засобів по кожному виду

Найменування програмного засобу	Кількість, шт	Ціна за одиницю, грн	Вартість, грн
Середовище розробки програмного забезпечення Visual Studio 2019 (ліцензія)	1	14296,00	15725,60
Всього			15725,60

5.3.7 Амортизація обладнання, програмних засобів та приміщень

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

$$A_{обл} = \frac{Ц_б}{T_в} \cdot \frac{t_{вик}}{12}, \quad (5.11)$$

де $Ц_б$ — балансова вартість обладнання, програмних засобів, приміщень тощо, які використовувались для проведення досліджень, грн;

$t_{вик}$ — термін використання обладнання, програмних засобів, приміщень під час досліджень, місяців;

$T_в$ — строк корисного використання обладнання, програмних засобів, приміщень тощо, років.

$A_{обл} = (28670{,}00 \cdot 2) / (2 \cdot 12) = 2389{,}17$ *грн.*

Проведені розрахунки зведемо до таблиці.

Таблиця 5.11 — Амортизаційні відрахування по кожному виду обладнання

Найменування обладнання	Балансова вартість, грн	Строк корисного використання, років	Термін використання обладнання, місяців	Амортизаційні відрахування, грн

Закінчення таблиці 5.11

Комп'ютер dell precision tower 5810	28670,00	2	2	2389,17
Робоче місце інженера-розробника програмного забезпечення	8650,00	5	2	288,33
Пристрої передачі даних	6900,00	3	2	383,33
Оргтехніка	8150,00	4	2	339,58
Приміщення дослідної лабораторії	307000,00	25	2	2046,67
ОС Windows 10	5700,00	2	2	475,00
Прикладний пакет Microsoft Office 2016	5100,00	2	2	425,00
Всього				6347,08

5.3.8 Паливо та енергія для науково-виробничих цілей

Витрати на силову електроенергію (B_e) розраховуємо за формулою:

$$B_e = \sum_{i=1}^{n} \frac{W_{yi} \cdot t_i \cdot Ц_e \cdot K_{вni}}{\eta_i}, \qquad (5.12)$$

де W_{yi} — потужність обладнання на визначеному етапі розробки, кВт;

t_i — тривалість роботи обладнання на етапі дослідження, год;

$Ц_e$ — вартість 1 кВт-години електроенергії, грн; (вартість електроенергії визначається за даними енергопостачальної компанії), приймемо $Ц_e$ = 4,50 грн;

$K_{вni}$ — коефіцієнт, що враховує використання потужності, $K_{вni} < 1$;

η_i — коефіцієнт корисної дії обладнання, $\eta_i < 1$.

$B_e = 0,45 \cdot 320,0 \cdot 4,50 \cdot 0,95 / 0,97 = 648,00$ *грн.*

Проведені розрахунки зведемо до таблиці.

Таблиця 5.12 — Витрати на електроенергію

Найменування обладнання	Встановлена потужність, кВт	Тривалість роботи, год	Сума, грн
Комп'ютер dell precision tower 5810	0,45	320,0	648,00
Робоче місце інженера-розробника програмного забезпечення	0,25	280,0	315,00
Пристрої передачі даних	0,12	150,0	81,00
Оргтехніка	0,12	24,0	12,96
Серверна станція	0,50	150,0	337,50
Маршрутизатор	0,02	150,0	13,50
Мережевий адаптер	0,01	150,0	6,75
Всього			1414,71

5.3.9 Службові відрядження

До статті «Службові відрядження» дослідної роботи на тему «Методи виявлення передавання прихованої інформації в службових заголовках протокольних блоків даних комп'ютерних мереж» належать витрати на відрядження штатних працівників, аспірантів тощо. Витрати за статтею «Службові відрядження» відсутні.

5.3.10 Витрати на роботи, які виконують сторонні підприємства, установи і організації

Витрати за статтею «Витрати на роботи, які виконують сторонні підприємства, установи і організації» розраховуємо як 30…45% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{сп} = (З_о + З_р) \cdot \frac{Н_{сп}}{100\%}, \quad (5.13)$$

де $Н_{сп}$ — норма нарахування за статтею «Витрати на роботи, які виконують сторонні підприємства, установи і організації», приймемо $Н_{сп}$= 30%.

$В_{сп}$ = (35174,17 + 1436,74) · 30 / 100% = 10983,27 грн.

5.3.11 Інші витрати

До статті «Інші витрати» належать витрати, які не знайшли відображення у зазначених статтях витрат і можуть бути віднесені безпосередньо на собівартість досліджень за прямими ознаками.

Витрати за статтею «Інші витрати» розраховуємо як 50…100% від суми основної заробітної плати дослідників та робітників за формулою:

$$I_в = (З_о + З_р) \cdot \frac{Н_{ів}}{100\%}, \qquad (5.14)$$

де $Н_{ів}$ — норма нарахування за статтею «Інші витрати», приймемо $Н_{ів}$ = 60%;

$І_в$ = (35174,17 + 1436,74) · 60 / 100% = 21966,54 грн.

5.3.12 Накладні (загальновиробничі) витрати

До статті «Накладні (загальновиробничі) витрати» належать: витрати, пов’язані з управлінням організацією; витрати на винахідництво та раціоналізацію; витрати на підготовку (перепідготовку) та навчання кадрів; витрати, пов’язані з набором робочої сили; витрати на оплату послуг банків; витрати, пов’язані з освоєнням виробництва продукції; витрати на науково-технічну інформацію та рекламу та ін.

Витрати за статтею «Накладні (загальновиробничі) витрати» розраховуємо як 100…150% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{нзв} = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%}, \quad (5.15)$$

де $Н_{нзв}$ — норма нарахування за статтею «Накладні (загальновиробничі) витрати», приймемо $Н_{нзв}$ = 100%.

$В_{нзв}$ = (35174,17 + 1436,74) · 100 / 100% = 36610,91 грн.

Витрати на проведення науково-дослідної роботи на тему «Методи виявлення передавання прихованої інформації в службових заголовках протокольних блоків даних комп'ютерних мереж» розраховуємо як суму всіх попередніх статей витрат за формулою:

$$B_{заг} = З_о + З_р + З_{дод} + З_н + М + К_в + В_{спец} + В_{прг} + А_{обл} + В_е + В_{св} + В_{сп} + І_в + В_{нзв}. \quad (4.18)$$

$В_{заг}$= 35174,17 +1436,74 +4027,20 +8940,38327 +3457,03 +19921,00 +0,00 + 15725,60 + 6347,08 +1414,71 +0,00 +10983,27 +21966,54 +36610,91 = 166004,63 грн.

Загальні витрати *ЗВ* на завершення науково-дослідної (науково-технічної) роботи та оформлення її результатів розраховується за формулою:

$$ЗВ = \frac{B_{заг}}{\eta}, \quad (5.16)$$

де η — коефіцієнт, який характеризує етап (стадію) виконання науково-дослідної роботи, приймемо η=0,9.

ЗВ = 166004,63 / 0,9 = 184449,59 грн.

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

В ринкових умовах узагальнюючим позитивним результатом, що його може отримати потенційний інвестор від можливого впровадження результатів

тієї чи іншої науково-технічної розробки, є збільшення у потенційного інвестора величини чистого прибутку.

Результати дослідження проведені за темою «Методи виявлення передавання прихованої інформації в службових заголовках протокольних блоків даних комп'ютерних мереж» передбачають комерціалізацію протягом 4-х років реалізації на ринку.

В цьому випадку майбутній економічний ефект буде формуватися на основі таких даних:

ΔN — збільшення кількості споживачів продукту, у періоди часу, що аналізуються, від покращення його певних характеристик;

Показник	1-й рік	2-й рік	3-й рік	4-й рік
Збільшення кількості споживачів, осіб	500	1000	800	500

N — кількість споживачів які використовували аналогічний продукт у році до впровадження результатів нової науково-технічної розробки, приймемо 13000 осіб;

$Ц_б$ — вартість програмного продукту у році до впровадження результатів розробки, приймемо 7000,00 грн;

$\pm\Delta Ц_o$ — зміна вартості програмного продукту від впровадження результатів науково-технічної розробки, приймемо 400,00 грн.

Можливе збільшення чистого прибутку у потенційного інвестора $\Delta П_i$ для кожного із 4-х років, протягом яких очікується отримання позитивних результатів від можливого впровадження та комерціалізації науково-технічної розробки, розраховуємо за формулою [17]**:**

$$\Delta П_i = (\pm\Delta Ц_o \cdot N + Ц_o \cdot \Delta N)_i \cdot \lambda \cdot \rho \cdot (1 - \frac{\vartheta}{100}), \qquad (5.17)$$

де λ — коефіцієнт, який враховує сплату потенційним інвестором податку на додану вартість. У 2021 році ставка податку на додану вартість складає 20%, а коефіцієнт $\lambda = 0,8333$;

ρ — коефіцієнт, який враховує рентабельність інноваційного продукту). Приймемо ρ =40%;

ϑ — ставка податку на прибуток, який має сплачувати потенційний інвестор, у 2021 році $\vartheta = 18\%$;

Збільшення чистого прибутку 1-го року:

$\Delta П_1$ =(400,00·13000,00+7400,00·500)·0,83·0,4·(1-0,18/100%)=2422936,00 грн.

Збільшення чистого прибутку 2-го року:

$\Delta П_2$ =(400,00·13000,00+7400,00·1500)·0,83·0,4·(1-0,18/100%)=4437512,00 грн.

Збільшення чистого прибутку 3-го року:

$\Delta П_3$ =(400,00·13000,00+7400,00·2300)·0,83·0,4·(1-0,18/100%)=6049172,80 грн.

Збільшення чистого прибутку 4-го року:

$\Delta П_4$ =(400,00·13000,00+7400,00·2800)·0,83·0,4·(1-0,18/100%)=7056460,80 грн.

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

$$ПП = \sum_{i=1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, \qquad (5.18)$$

де $\Delta П_i$ — збільшення чистого прибутку у кожному з років, протягом яких виявляються результати впровадження науково-технічної розробки, грн;

T — період часу, протягом якого очікується отримання позитивних результатів від впровадження та комерціалізації науково-технічної розробки, роки;

τ — ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні, τ =0,15;

t — період часу (в роках) від моменту початку впровадження науково-технічної розробки до моменту отримання потенційним інвестором додаткових чистих прибутків у цьому році.

$$ПП = 2422936,00/(1+0,15)^1+4437512,00/(1+0,15)^2+6049172,80/(1+0,15)^3+$$
$$+7056460,80/(1+0,15)^4=2106900,87+3355396,60+3977429,31+4034554,36=$$
$$=13474281,14 \text{ грн.}$$

Величина початкових інвестицій PV, які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної розробки:

$$PV = k_{інв} \cdot ЗВ, \tag{5.19}$$

де $k_{інв}$ — коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію, приймаємо $k_{інв}$ =2;

$ЗВ$ — загальні витрати на проведення науково-технічної розробки та оформлення її результатів, приймаємо 184449,59 грн.

$$PV = k_{інв} \cdot ЗВ = 2 \cdot 184449,59 = 368899,18 \text{ грн.}$$

Абсолютний економічний ефект $E_{абс}$ для потенційного інвестора від можливого впровадження та комерціалізації науково-технічної розробки становитиме:

$$E_{абс} = ПП - PV \tag{5.20}$$

де $ПП$ — приведена вартість зростання всіх чистих прибутків від можливого впровадження та комерціалізації науково-технічної розробки, 13474281,14 грн;

PV — теперішня вартість початкових інвестицій, 368899,18 грн.

$E_{абс} = ПП - PV$ = 13474281,14 - 368899,18 = 13105381,97 грн.

Внутрішня економічна дохідність інвестицій $E_в$, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$E_в = \sqrt[Тж]{1 + \frac{E_{абс}}{PV}} - 1, \quad (5.21)$$

де $E_{абс}$ — абсолютний економічний ефект вкладених інвестицій, 13105381,97 грн;

PV — теперішня вартість початкових інвестицій, 368899,18 грн;

$T_ж$ — життєвий цикл науково-технічної розробки, тобто час від початку її розробки до закінчення отримування позитивних результатів від її впровадження, 4 роки.

$$E_в = \sqrt[Тж]{1 + \frac{E_{абс}}{PV}} - 1 = (1+13105381,97/368899,18)^{1/4} = 1,46.$$

Мінімальна внутрішня економічна дохідність вкладених інвестицій $\tau_{мін}$:

$$\tau_{мін} = d + f, \quad (5.22)$$

де d — середньозважена ставка за депозитними операціями в комерційних банках; в 2021 році в Україні d =0,1;

f — показник, що характеризує ризикованість вкладення інвестицій, приймемо 0,25.

$\tau_{мін}$ = 0,1+0,25 = 0,35 < 1,46 свідчить про те, що внутрішня економічна дохідність інвестицій $E_в$, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки вища мінімальної внутрішньої дохідності. Тобто інвестувати в науково-дослідну роботу за темою

«Методи виявлення передавання прихованої інформації в службових заголовках протокольних блоків даних комп'ютерних мереж» доцільно.

Період окупності інвестицій $T_{ок}$ які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$T_{ок} = \frac{1}{E_{в}}, \tag{5.23}$$

де $E_{в}$ — внутрішня економічна дохідність вкладених інвестицій.

$T_{ок}$ = 1 / 1,46 = 0,69 р.

$T_{ок}$ < 3-х років, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження даної розробки та виведення її на ринок.

Згідно проведених досліджень рівень комерційного потенціалу розробки за темою «Методи виявлення передавання прихованої інформації в службових заголовках протокольних блоків даних комп'ютерних мереж» становить 38,3 бала, що, свідчить про комерційну важливість проведення даних досліджень (рівень комерційного потенціалу розробки вище середнього).

Також термін окупності становить 0,69 р., що менше 3-х років, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження даної розробки та виведення її на ринок.

Отже можна зробити висновок про доцільність проведення науково-дослідної роботи за темою «Методи виявлення передавання прихованої інформації в службових заголовках протокольних блоків даних комп'ютерних мереж».

ВИСНОВКИ

У магістерській кваліфікаційній роботі було розроблено та протестовано методи виявлення прихованих даних, вбудованих в поля службових заголовків протокольних блоків даних.

У першому розділі було проведено аналіз технологій та методів, що дозволяють передавати приховані повідомлення в комп'ютерній мережі. Виявлено їх переваги та недоліки, що дає змогу розробити підходящі методи для пошуку тих, що потрапили в локальну комп'ютерну мережу.

У другому розділі було розглянуто та проаналізовано методику, за якою створюються повідомлення-контейнери з вбудованою прихованою інформацією. Поетапно розглянута процедура генерування пакетів з вбудованими стеганограмами та процес атаки комп'ютерної корпоративної мережі з їх використанням. Були виведені основні положення щодо комплексу дій, які потрібно зробити для виявлення стеганогрфічних повідомлень всередині корпоративної мережі. Розглянуто набір тестів на псевдовипадковість полів NIST, що дозволяють перевірити яким чином було заповнено їх вміст. Обрано найоптимальніший набір тестів для перевірки полів мережевих пакетів. Розроблено структуру алгоритму для перехоплення та перевірки пакетів, що передаються в мережі.

У третьому розділі магістерської кваліфікаційної роботи було розроблено програму для перехоплення мережевого трафіку та аналізу пакетів на наявність вбудованих повідомлень в полі «Ідентифікатор пакету» шляхом його тестування на псевдовипадковість за допомогою набору тестів NIST. Було спроектовано мінімальний зручний користувацький інтерфейс та створено алгоритм перехоплення та перевірки пакетів.

У четвертому розділі було протестовано систему на її працездатність та функціональність. Покроково описано її функціональні можливості та процес взаємодії з користувачем.

У п’ятому розділі були проведені дослідження та розрахунки доцільності розробки та використання розробленої системи. У результаті проведення економічних розрахунків рівня комерційного потенціалу було отримано значення в 38,3 балів. Це вказує на те, що рівень комерційного потенціалу відповідає рівню вище середнього. Термін окупності таких досліджень становить 0.69 років, що є комерційно привабливим для потенційного інвестора, який може профінансувати впровадження даної розробки для виведення її на ринок.

На основі отриманих результатів можна зробити висновок, що використання подібних методів для виявлення стеганографічного втручання в комп’ютерній мережі є перспективним напрямком. Оскільки кількість кіберзлочинів такого виду щорічно зростає, поява на ринку системи, що може їм запобігти і захистити локальні мережі від зовнішнього втручання, буде значною подією. Тому потрібно продовжувати дослідження в цьому напрямку і пропонувати все новіші та комплексніші рішення цієї проблеми.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Сирцов С. Р. Методи виявлення передавання прихованої інформації в службових заголовках протокольних блоків даних комп'ютерних мереж. / Сирцов С. Р. //Тези доповіді. Молодь в науці: дослідження, проблеми, перспективи (МН-2022). — Режим доступу до ресурсу: https://conferences.vntu.edu.ua/index.php /mn/mn2022/paper/view/14233

2. Сирцов С. Р. Метод прихованої передачі інформації в службових підзаголовках протокольних блоків даних. / Сирцов С. Р. , Захарченко С. М. // Тези доповіді. XLIX регіональна науково-технічна конференція професорсько-викладацького складу, співробітників та студентів університету з участю працівників науково-дослідних організацій та інженерно-технічних працівників підприємств м.Вінниці та області. — Режим доступу до ресурсу: https://conferences.vntu.edu.ua/index.php/all-fitki/all-fitki-2020/paper/view/9759/8306

3. Фороузан Б. А Криптография и безопасность сетей: Учеб. пособие: с англ. под ред. А. Н. Берлина. — М.: Интернет-Университет Информационных Технологий: БИНОМ. Лаборатория знаний, 2010. — 784 с.

4. Панасенко С. А. Алгоритмы шифрования. Специальный справчник — М.: БХВ-Петербург, 209. —576 с.

5. Бабаш А. В., Шанькин Г. П. Криптография / Под ред. В. П. Шерстюка, Э. А. Применко. — М.: СОЛОН-ПРЕСС, 2007. — 512 с.

6. Шнайер Б. Практическая криптография. — М.: Вильямс, 2005. — 424 с.

7. Bender, W., Gruhl, D., Morimoto, N., Lu, A.: Techniques for Data Hiding. IBM. System Journal, 1996 — С. 313–336.

8. Белкина Т. А. Аналитический обзор применения сетевой стеганографии для решения задач информационной безопасности // Молодой ученый. — 2018. — №11. — С. 36-44.

9. Применение сетевой стеганографии для скрытия даних передаваемых по каналам связи [Електронний ресурс] — Режим доступу до ресурсу:

https://cyberleninka.ru/article/n/primenenie-setevoy-steganografii-dlya-skrytiya-dannyh-peredavaemyh-po-kanalam-svyazi

10. Szczypiorski, K., Steganography in TCP/IP Networks. State of the Art and a Proposal of a New System— HICCUPS, Institute of Telecommunications' seminar, Warsaw University of Technology, Poland,November 2003, URL:http://krzysiek.tele.pw.edu.pl/pdf/steg-seminar-2003.pdf

11. Kundur D., Ahsan K., Practical Internet Steganography: Data Hiding in IP, Proc. Texas Wksp. Security of Information Systems, Apr. 2003

12. Сетевая стеганографияя на основе ICMP-инкапсуляция [Електронний ресурс] — Режим доступу до ресурсу: https://cyberleninka.ru/article/n/primenenie-setevoy-steganografii-dlya-skrytiya-dannyh-peredavaemyh-po-kanalam-svyazi

13. Mazurczyk W., Smolarczyk M., Szczypiorski K. RSTEG: Retransmission Detection, In: Soft Computing in 2010, ISSN: 1432-7643 (print version) ISSN: 1433-7479 (electronic version), Journal no. 500 Springer. Steganography and Its

14. Mazurczyk, W., Szczypiorski K., Covert Channels in SIP for VoIP signaling, In Proc. of 4th International Conference on Global E-security 2008, London, United Kingdom, 23-25 June 2008, pp. 65-72

15. Статические тесты NIST [Електронний ресурс] — Режим доступу до ресурсу: https://ru.wikipedia.org/wiki/Статистические_тесты_NIST

16. Эндрю С. Таненбаум, Дэвид Дж. Ветерал. Комп’ютерные сети. Издательство Питер 2012г, серия классика Информатика. ISBN: 978-5-4461-0068-2, 960 с.

17. Методичні вказівки до виконання економічної частини магістерських кваліфікаційних робіт / Уклад. : В. О. Козловський, О. Й. Лесько, В. В. Кавецький. – Вінниця : ВНТУ, 2021. – 42 с.

18. Кавецький В. В. Економічне обґрунтування інноваційних рішень: практикум / В. В. Кавецький, В. О. Козловський, І. В. Причепа – Вінниця : ВНТУ, 2016. – 113 с.

ДОДАТОК А

Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет інформаційних технологій та комп’ютерної інженерії
Кафедра обчислювальної техніки

ЗАТВЕРДЖУЮ
Завідувач кафедри ОТ
д.т.н., проф. О. Д. Азаров
“__” _____ 2021 р.

ТЕХНІЧНЕ ЗАВДАННЯ
на виконання магістерської кваліфікаційної роботи на тему:
«Методи виявлення передавання прихованої інформації в службових заголовках протокольних блоків даних комп'ютерних мереж»
08-23.МКР.011.00.000 ПЗ

Науковий керівник к.т.н., проф. каф. ОТ
_____ Захарченко С. М.
Студент групи 1КІ-20м
_____ Сирцов С. Р.

Вінниця 2021

1 Підстава для використання МКР

а) актуальність розробки полягає у створенні та вдосконаленні технологій та методів захисту інформації в комп'ютерних мережах.

б) наказ про затвердження теми кваліфікаційної роботи.

2 Мета і призначення МКР

а) мета роботи — дослідження та вдосконалення методів виявлення передавання прихованої інформації в комп'ютерних мережах за допомогою протокольних блоків даних;

б) призначення розробки — програмній реалізації методу виявлення передавання прихованих даних завдяки перехопленню та перевірці пакетів.

3 Джерела розробки МКР

Інтегроване середовище розробки Visual Studio для платформи .NET Framework та мов програмування C# і C++ в ОС Windows.

4 Технічні вимоги до виконання МКР

— наявність мережі з підтримкою роботи протоколів TCP та IP;

— наявність програми, яка шифрує, приховує та відправляє користувацькі дані мережею.

5 Етапи МКР та очікувані результати

5.1 Робота виконується за п'ять етапів, таблиця А.1.

Таблиця А.1 — Етапи виконання роботи

№ етапу	Назва етапу	Термін виконання		Очікувані результати
		початок	кінець	
1	Аналіз сучасного стану досліджень	01.09.2021	01.10.2021	Аналітичний огляд літературних джерел, задачі досліджень

Закінчення таблиці А.1

2	Побудова математичних моделей	01.10.2021	10.10.2021	Математичні моделі, 2 розділ
3	Практичне застосування та оцінка ефективності розроблених моделей	11.10.2021	24.10.2021	3 і 4 розділи
4	Підготовка економічної частини	25.10.2021	7.11.2021	5 розділ
5	Апробація та/або впровадження результатів дослідження	8.11.2021	21.11.2021	тези доповідей /акт впровадження
6	Опублікування результатів досліджень	22.11.2021	3.12.2021	стаття
7	Оформлення пояснювальної записки, графічного матеріалу і/або презентації	4.12.2021	20.12.2021	пояснювальна записка, графічний матеріал і/або презентація

6 Матеріали, що подаються до захисту МКР

Пояснювальна записка МКР, графічні і ілюстративні матеріали, протокол попереднього захисту МКР на кафедрі, відзив наукового керівника, відзив опонента, протоколи проходження перевірки на плагіат, анотації до МКР українською та іноземною мовами, нормоконтроль про відповідність оформлення МКР діючим вимогам.

7 Порядок контролю виконання та захисту МКР

Виконання етапів графічної та розрахункової документації МКР контролюється науковим керівником згідно зі встановленими термінами. Захист МКР відбувається на засіданні Державної екзаменаційної комісії, затвердженою наказом ректора.

8 Вимоги до оформлення МКР

Оформлення магістерської кваліфікаційної роботи відбувається згідно «Положень про кваліфікаційні роботи» та ДСТУ 3008-2015.

Технічне завдання отримав _____ Сирцов С. Р.

ДОДАТОК Б

Узагальнене представлення стеганосистем

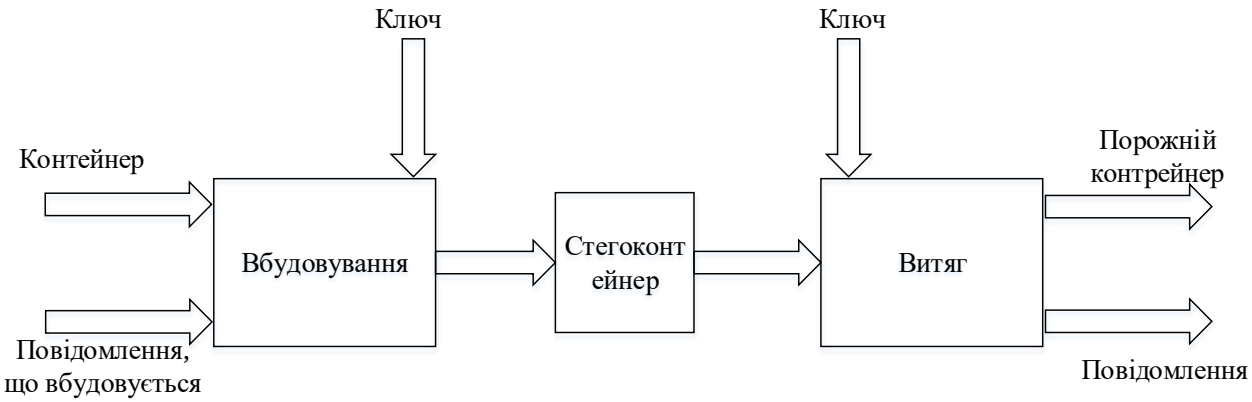

Рисунок В.1 — Загальний вигляд стеганосистем

ДОДАТОК В

Класифікація методів мережевої стеганографії

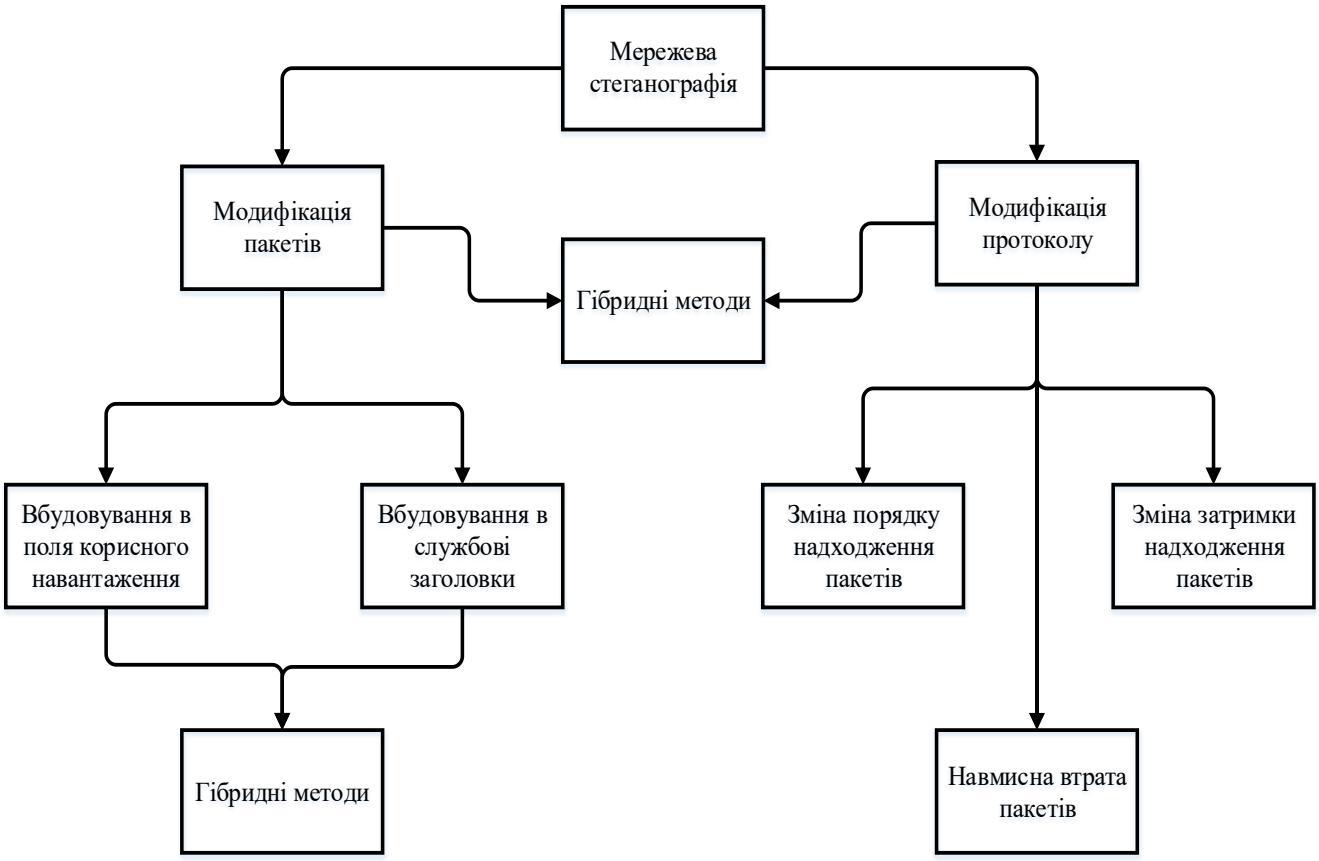

Рис. Г.1 — Класифікація методів мережевої стеганографії

ДОДАТОК Г

Спроектований макет програми перехоплення пакетів

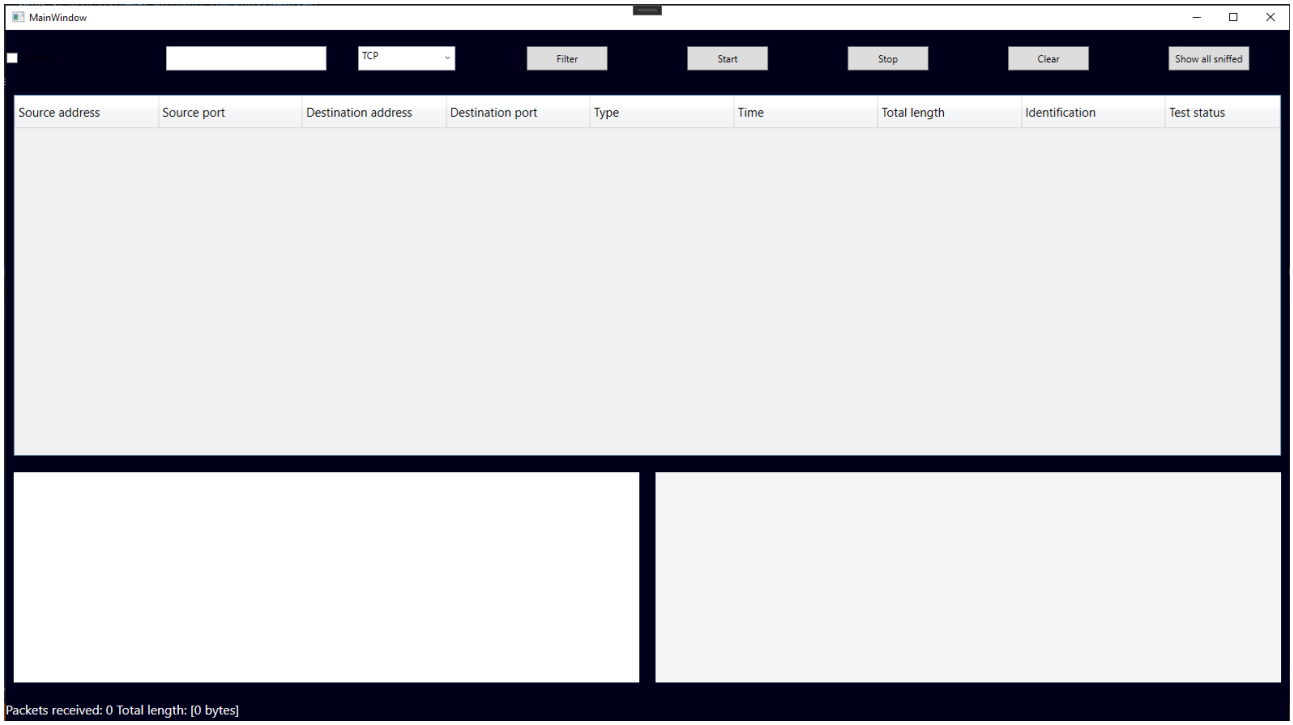

Рис. Д.1 — Спроектований макет програми перехоплення пакетів

ДОДАТОК Д

Графічна схема алгоритму

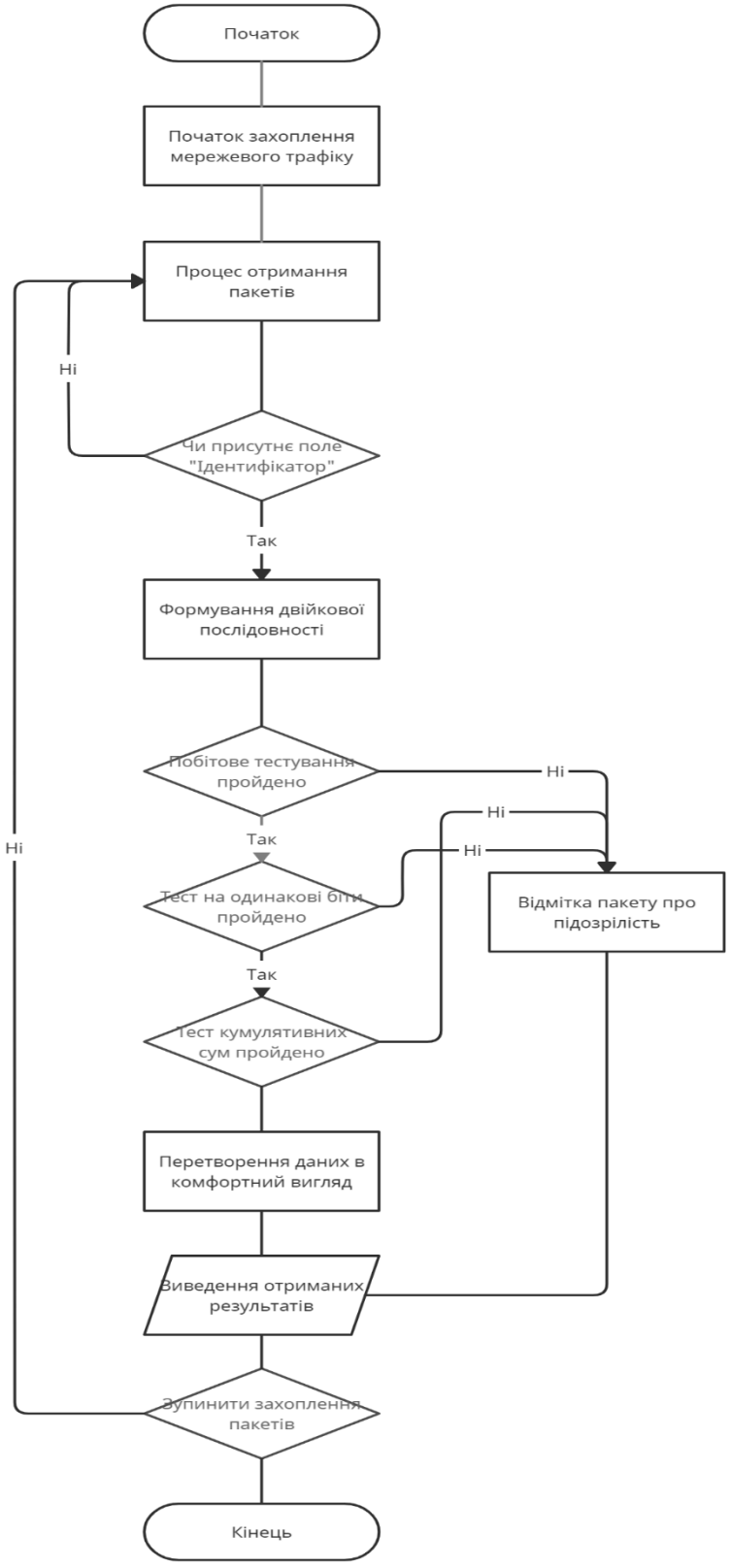

Рисунок Е.1 — Графічна схема алгоритму

ДОДАТОК Е
Лістинг коду програми

MainWindow.xaml

```
<Window x:Class="CovertChannelsDetector.MainWindow"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
      xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
      xmlns:local="clr-namespace:CovertChannelsDetector"
      mc:Ignorable="d"
      Title="MainWindow" Height="900" Width="1600" Loaded="Window_Loaded">
   <Grid Background="#00001a">
      <Grid.Resources>
         <local:WarningConverter x:Key="WarningConverter"/>

         <Style x:Key="ConvertColorCell1" TargetType="{x:Type Button}">
            <Setter Property="Background" Value="Red"/>
            <Setter Property="Visibility" Value="{Binding Identification,
Converter={StaticResource WarningConverter}}"/>
            <Setter Property="OverridesDefaultStyle" Value="True"/>
            <Setter Property="Margin" Value="5"/>
            <Setter Property="Template">
               <Setter.Value>
                  <ControlTemplate TargetType="Button">
                     <Border Name="border"
                           BorderThickness="0"
                           Padding="4,2"
                           BorderBrush="DarkGray"
                           CornerRadius="3"
                           Background="{TemplateBinding Background}">
                        <ContentPresenter HorizontalAlignment="Center"
VerticalAlignment="Center"/>
                     </Border>
                     <ControlTemplate.Triggers>
                        <Trigger Property="IsMouseOver" Value="False">
                           <Setter TargetName="border" Property="BorderBrush" Value="Black"/>
                        </Trigger>
                     </ControlTemplate.Triggers>
                  </ControlTemplate>
               </Setter.Value>
            </Setter>
         </Style>

         <DataTemplate x:Key="Show_status">
            <Button Style="{StaticResource ConvertColorCell1}">
               <TextBlock Text="Test failed" Foreground="White" FontSize="14"/>
            </Button>
         </DataTemplate>
      </Grid.Resources>
```

```
    <Grid.RowDefinitions>
      <RowDefinition Height="3*"/>
      <RowDefinition Height="20*"/>
      <RowDefinition Height="12*"/>
      <RowDefinition Height="2*"/>
    </Grid.RowDefinitions>

    <Grid Grid.Row="0">
      <Grid.ColumnDefinitions>
        <ColumnDefinition/>
        <ColumnDefinition/>
        <ColumnDefinition/>
        <ColumnDefinition/>
        <ColumnDefinition/>
        <ColumnDefinition/>
        <ColumnDefinition/>
        <ColumnDefinition/>
      </Grid.ColumnDefinitions>

      <CheckBox x:Name="filterCheckBox" Grid.Column="0" VerticalAlignment="Center"
Content="Filtering"/>
      <TextBox x:Name="ipTextBox" Grid.Column="1" Height="30"/>
      <ComboBox x:Name="typeComboBox" Grid.Column="2" Height="30" Width="120"
Text="TCP" IsEditable="True" IsReadOnly="True"/>
      <Button x:Name="filterButton" Grid.Column="3" Height="30" Width="100"
Content="Filter" Click="filterButton_Click"/>
      <Button x:Name="startButton" Grid.Column="4" Height="30" Width="100"
Content="Start" Click="startButton_Click"/>
      <Button x:Name="stopButton" Grid.Column="5" Height="30" Width="100"
Content="Stop" Click="stopButton_Click"/>
      <Button x:Name="clearButton" Grid.Column="6" Height="30" Width="100"
Content="Clear" Click="clearButton_Click" MouseDoubleClick="clearButton_DoubleClick"/>
      <Button x:Name="allButton" Grid.Column="7" Height="30" Width="100" Content="Show
all sniffed" Click="allButton_Click"/>
    </Grid>


    <DataGrid x:Name="dataGrid" Margin="10" Grid.Row="1" ColumnHeaderHeight="40"
RowHeight="36" SelectionMode="Single" FontSize="15"
                  CanUserResizeRows="False" CanUserResizeColumns="False"
IsReadOnly="True" CanUserAddRows="False" CanUserDeleteRows="False"
                  AutoGenerateColumns="False" VerticalGridLinesBrush="Transparent"
SelectionChanged="dataGrid_SelectionChanged">
      <DataGrid.Columns>
        <DataGridTextColumn  Header="Source address" Binding="{Binding SourceIP}"
Width="5*" />
        <DataGridTextColumn  Header="Source port" Binding="{Binding SourcePort}"
Width="5*"/>
        <DataGridTextColumn Header="Destination address" Binding="{Binding
DestinationIP}" Width="5*"/>
```

```
            <DataGridTextColumn Header="Destination port" Binding="{Binding DestinationPort}"
Width="5*"/>
            <DataGridTextColumn Header="Type" Binding="{Binding Type}" Width="5*"/>
            <DataGridTextColumn Header="Time" Binding="{Binding Timestamp}" Width="5*"/>
            <DataGridTextColumn Header="Total length" Binding="{Binding TotalLength}"
Width="5*"/>
            <DataGridTextColumn Header="Identification" Binding="{Binding Identification}"
Width="5*"/>
            <DataGridTemplateColumn  Header="Test status"   CellTemplate="{StaticResource
Show_status}"  Width="4*"/>
        </DataGrid.Columns>
    </DataGrid>

    <Grid Grid.Row="2">
        <Grid.ColumnDefinitions>
            <ColumnDefinition/>
            <ColumnDefinition/>
        </Grid.ColumnDefinitions>

        <TextBlock x:Name="charTextBox" Grid.Column="0" Background="White"
Margin="10"/>
        <TextBlock x:Name="hexTextBox" Grid.Column="1" Background="WhiteSmoke"
Margin="10"/>
    </Grid>

    <TextBlock x:Name="hintLabel" Grid.Row="3" Text="Packets received: 0 Total length: [0
bytes]" VerticalAlignment="Center" FontSize="16" Foreground="White"/>
  </Grid>
</Window>
```

MainWindow.xaml.cs

```
namespace CovertChannelsDetector
{
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {


        /// <summary>
        /// used to rake the underlying packets;
        /// </summary>
```

```
        List<Monitor> monitorList = new List<Monitor>();

        /// <summary>
        /// presenting packets;
        /// </summary>
        List<Packet> pList = new List<Packet>();

        /// <summary>
        /// the packets sniffed -- all;
        /// </summary>
        List<Packet> allList = new List<Packet>();

        /// <summary>
        /// used to refresh the packets sniffed and dataGrid and all the related info;
        /// </summary>
        /// <param name="p"></param>
        delegate void refresh(Packet p);

        /// <summary>
        /// total length sniffed so far - isolating the filtered;
        /// </summary>
        long totalLength = 0;

        /// <summary>
        /// the count of the packets sniffed;
        /// </summary>
        long totalCount = 0;

        ObservableCollection<PacketInfo> PacketsCollection = new
ObservableCollection<PacketInfo>();
```

```
public MainWindow()
{
    InitializeComponent();
    dataGrid.ItemsSource = PacketsCollection;

}

private void Window_Loaded(object sender, RoutedEventArgs e)
{

}

private void deactivateSearch()
{
    filterCheckBox.IsEnabled = false;
    ipTextBox.IsEnabled = false;
    typeComboBox.IsEnabled = false;
    startButton.IsEnabled = false;
    filterButton.IsEnabled = false;
    allButton.IsEnabled = false;
}

/// <summary>
/// activate the buttons deactivated before;
/// </summary>
private void activateSearch()
{
    filterCheckBox.IsEnabled = true;
    ipTextBox.IsEnabled = true;
    typeComboBox.IsEnabled = true;
    startButton.IsEnabled = true;
```

```
        filterButton.IsEnabled = true;
        allButton.IsEnabled = true;
    }

    private void startRaking()
    {
        monitorList.Clear();
        IPAddress[] hosts = Dns.GetHostEntry(Dns.GetHostName()).AddressList;
        if (hosts == null || hosts.Length == 0)
        {
            MessageBox.Show("No hosts detected, please check your network!");
        }
        for (int i = 0; i < hosts.Length; i++)
        {
            Monitor monitor = new Monitor(hosts[i]);
            monitor.newPacketEventHandler += new
Monitor.NewPacketEventHandler(onNewPacket);
            monitorList.Add(monitor);
        }
        foreach (Monitor monitor in monitorList)
        {
            monitor.start();
        }
    }

    private void onNewPacket(Monitor monitor, Packet p)
    {

        Dispatcher.Invoke(new refresh(onRefresh), p);
        //this.BeginInvoke(new refresh(onRefresh), p);
    }
```

```
        private void onRefresh(Packet p)
        {
            //MessageBox.Show(filterCheckBox.Checked.ToString() + filteringPattern);
            if ((bool)filterCheckBox.IsChecked)
            {
                string[] conditions = getFilterCondition();
                if (isIPOkay(p, conditions[0]) && isPORTOkay(p, conditions[1])
                    && (conditions[2] == "" || conditions[2] == p.Type))
                {
                    addAndUpdatePackets(p);
                }
            }
            else
            {
                addAndUpdatePackets(p);
            }
            //MessageBox.Show(p.Src_IP);
            if (totalLength < 10 * 1024)
            {
                hintLabel.Text = string.Format("Packets received {0}  Total length :  [{1} bytes]",
totalCount, totalLength);
            }
            else if (totalLength < 10 * 1024 * 1024)
            {
                hintLabel.Text = string.Format("Packets received {0}  Total length :  [{1} KB]",
totalCount, totalLength / 1024);
            }
            else if (totalLength < 1024 * 1024 * 1024)
            {
                hintLabel.Text = string.Format("Packets received {0}  Total length :  [{1} MB]",
totalCount, totalLength / (1024 * 1024));
```

```
        }
        else if (totalLength < (long)1024 * 1024 * 1024 * 2)
        {
            hintLabel.Text = string.Format("Packets received {0}  Total length :  [{1} GB]",
totalCount, totalLength / (1024 * 1024 * 1024));
        }
        else
        {
            totalCount = 0;
            totalLength = 0;
            dataGrid.Items.Clear();
            pList.Clear();
            allList.Clear();
            hintLabel.Text = string.Format("Packets received {0}  Total length :  [{1} bytes]", 0, 0);
        }
    }

    /// <summary>
    /// ip is "" or is equal to p.Src_IP or p.Des_IP;
    /// </summary>
    /// <param name="p"></param>
    /// <param name="ip"></param>
    /// <returns></returns>
    private bool isIPOkay(Packet p, string ip)
    {
        return ip == "" || p.Src_IP == ip || p.Des_IP == ip;
    }

    /// <summary>
    /// port is either "" or is equal to p.Src_PORT or p.Des_PORT;
    /// </summary>
```

```
/// <param name="p"></param>
/// <param name="port"></param>
/// <returns></returns>
private bool isPORTOkay(Packet p, string port)
{
    return port == "" || p.Src_PORT == port || p.Des_PORT == port;
}

/// <summary>
/// add one packet to pList and the dataGrid;
/// besides, refresh the totalCount, totalLength and allList globally;
/// </summary>
/// <param name="p"></param>
private void addAndUpdatePackets(Packet p)
{
    totalCount++;
    totalLength += p.TotalLength;
    allList.Add(p);
    pList.Add(p);

    PacketsCollection.Insert(0, new PacketInfo
    {
        SourceIP = p.Src_IP,
        DestinationIP = p.Des_IP,
        SourcePort = p.Src_PORT,
        DestinationPort = p.Des_PORT,
        Type = p.Type,
        Timestamp = p.Time,
        TotalLength = p.TotalLength.ToString(),
        Content = p.getCharString(),
        ContentHex = p.getHexString(),
```

```
            Identification = ""
        });

        if (p.getCharString().Contains("...."))
        {
            PacketsCollection.Insert(0, new PacketInfo
            {
                SourceIP = p.Src_IP,
                DestinationIP = p.Des_IP,
                SourcePort = p.Src_PORT,
                DestinationPort = p.Des_PORT,
                Type = p.Type,
                Timestamp = p.Time,
                TotalLength = p.TotalLength.ToString(),
                Content = p.getCharString(),
                ContentHex = p.getHexString(),
                Identification = "TEST"
            });
        }
        // dataGrid.EnsureVisible(dataGrid.Items.Count > 5 ? dataGrid.Items.Count - 10 :
dataGrid.Items.Count);
    }

    /// <summary>
    /// stop sniffing the network;
    /// </summary>
    private void stopReceiving()
    {
        foreach (Monitor monitor in monitorList)
        {
            monitor.stop();
```

```
        }
    }

    /// <summary>
    /// when not selecting the list item, clear the details of the previous item;
    /// </summary>
    private void clearDetail()
    {
        charTextBox.Text = "";
        hexTextBox.Text = "";
    }

    /// <summary>
    /// handle the start button event - deactivating filterCheckBox, ipTextBox, typeComboBox, startButton, filterButton and allButton
    /// and start sniffing the local network;
    /// </summary>
    /// <param name="sender"></param>
    /// <param name="e"></param>
    private void startButton_Click(object sender, RoutedEventArgs e)
    {
        clearDetail();
        deactivateSearch();
        startRaking();
    }

    /// <summary>
    /// activating buttons and stop sniffing around;
    /// </summary>
    /// <param name="sender"></param>
    /// <param name="e"></param>
```

```
private void stopButton_Click(object sender, RoutedEventArgs e)
{
    clearDetail();
    activateSearch();
    stopReceiving();
}

/// <summary>
/// clear the pList and dataGrid but not the allList;
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
private void clearButton_Click(object sender, RoutedEventArgs e)
{
    //dataGrid.Items.Clear();
    pList.Clear();
    clearDetail();
}


/// <summary>
/// when double click the clear button, clear all the status;
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
private void clearButton_DoubleClick(object sender, MouseButtonEventArgs e)
{
    totalCount = 0;
    totalLength = 0;
    clearDetail();
    //dataGrid.Items.Clear();
```

```
            PacketsCollection.Clear();
            this.pList.Clear();
            this.allList.Clear();
            this.hintLabel.Text = string.Format("Packets received {0}  Total length：  [{1} bytes]", 0,
0);
        }
        /// <summary>
        /// Display all the sniffed packets in dataGrid;
        /// </summary>
        /// <param name="sender"></param>
        /// <param name="e"></param>
        private void allButton_Click(object sender, RoutedEventArgs e)
        {
            //dataGrid.Items.Clear();
            PacketsCollection.Clear();
            pList.Clear();
            Packet p;
            for (int i = 0; i < allList.Count; i++)
            {
                p = allList[i];
                pList.Add(p);
                PacketsCollection.Insert(0, new PacketInfo
                {
                    SourceIP = p.Src_IP,
                    DestinationIP = p.Des_IP,
                    SourcePort = p.Src_PORT,
                    DestinationPort = p.Des_PORT,
                    Type = p.Type,
                    Timestamp = p.Time,
                    TotalLength = p.TotalLength.ToString(),
                    Content = p.getCharString(),
```

```
            ContentHex = p.getHexString()
        });
    }
    clearDetail();
}



//private void ipTextBox_GotFocus(object sender, System.EventArgs e)
//{
//    ipTextBox.ForeColor = Color.Black;
//    ipTextBox.Text = "";
//    ipTextBox.GotFocus -= ipTextBox_GotFocus;
//}

/// <summary>
/// when selecting the list item and present the details at the bottom of the panel;
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
private void dataGrid_SelectionChanged(object sender, SelectionChangedEventArgs e)
{
    if (dataGrid.SelectedItems != null && dataGrid.SelectedItems.Count != 0)
    {
        PacketInfo p = (PacketInfo)dataGrid.SelectedItem;
        //MessageBox.Show("charLength:" + p.getCharString().Length + "\n" + "hexLength:" +
p.getHexString().Length);
        this.hexTextBox.Text = p.ContentHex;
        this.charTextBox.Text = p.Content;
    }
}
```

```
//count a certain char in a string
private int getCharCount(string s, char c)
{
    int count = 0;
    for (int i = 0; i < s.Length; i++)
    {
        if (s[i] == c)
        {
            count++;
        }
    }
    return count;
}

/// <summary>
/// return the indexes of the substring - s0 in its parentString
/// </summary>
/// <param name="s">this value can be changed within the method</param>
/// <param name="s0"></param>
/// <returns></returns>
private List<int> getStringIndex(string s, string s0)
{
    List<int> countList = new List<int>();
    int index = 0;///indicate the current index
    while (s.Contains(s0))
    {
        index = s.IndexOf(s0);
        s = s.Substring(0, index + s0.Length);
        countList.Add(index);///record the relative indexes
    }
```

```
        for (int i = 1; i < countList.Count; i++)///get the absolute indexes
        {
            countList[i] += (countList[i - 1] + s0.Length);
        }
        return countList;
    }

    private void filterButton_Click(object sender, RoutedEventArgs e)
    {
        if (dataGrid.Items.Count < 1)
        {
            MessageBox.Show("Please sniff or show all the sniffed packets first ! ");
        }
        showIPPackets(getFilterCondition());
        clearDetail();
    }

    /// <summary>
    /// according to the ipTextBox and typeComboBox get the filter conditions including ip, port and type;
    /// default value "";
    /// </summary>
    /// <returns></returns>
    private string[] getFilterCondition()
    {
        string[] conditions = { "", "", "" };
        string tmpString = this.ipTextBox.Text;
        int port = 0;
        if (this.typeComboBox.SelectedIndex > -1)
            conditions[2] = this.typeComboBox.SelectedItem.ToString();
        if (tmpString.Contains('/') || tmpString.Contains(':'))//IP:PORT OR IP/PORT
```

```
    {
        string[] arr = { null, null };
        if (tmpString.Contains('/'))
            arr = tmpString.Split(new char[] { '/' });
        else
            arr = tmpString.Split(new char[] { ':' });
        conditions[0] = arr[0];
        conditions[1] = arr[1];
    }
    else if (int.TryParse(tmpString, out port))//just port;
        conditions[1] = port.ToString();
    else//just IP;
        conditions[0] = tmpString;
    //Console.WriteLine(conditions);
    return conditions;
}

private void showIPPackets(string[] conditions)
{
    string ipString = conditions[0];
    string port = conditions[1];
    string type = conditions[2];
    Packet p;
    //dataGrid.Items.Clear();
    PacketsCollection.Clear();
    pList.Clear();
    for (int i = 0; i < allList.Count; i++)
    {
        p = allList[i];
        if (isIPOkay(p, conditions[0]) && isPORTOkay(p, conditions[1])
            && (conditions[2] == "" || conditions[2] == p.Type))
```

```
            {
                pList.Add(p);
                PacketsCollection.Insert(0, new PacketInfo
                {
                    SourceIP = p.Src_IP,
                    DestinationIP = p.Des_IP,
                    SourcePort = p.Src_PORT,
                    DestinationPort = p.Des_PORT,
                    Type = p.Type,
                    Timestamp = p.Time,
                    TotalLength = p.TotalLength.ToString(),
                    Content = p.getCharString(),
                    ContentHex = p.getHexString()
```

Monitor.cs

```
class Monitor
   {
      private const int SECURITY_BUILTIN_DOMAIN_RID = 0x20;
      private const int DOMAIN_ALIAS_RID_ADMINS = 0x220;

      private const int IOC_VENDOR = 0x18000000;
      private const int IOC_IN = -2147483648; //0x80000000;
      private const int SIO_RCVALL = IOC_IN | IOC_VENDOR | 1;
      private const int BUF_SIZE = 1024 * 1024;


      private Socket monitor_Socket;
      private IPAddress ipAddress;
      private byte[] buffer;

      public Monitor(IPAddress ip)
      {
         this.ipAddress = ip;
         this.buffer = new byte[BUF_SIZE];
      }

      ~Monitor()
      {
         stop();
      }
```

```
    public void start()
    {
        if (monitor_Socket == null)
        {
            try
            {

                if (ipAddress.AddressFamily == AddressFamily.InterNetwork)
                {
                    monitor_Socket = new Socket(AddressFamily.InterNetwork, SocketType.Raw,
System.Net.Sockets.ProtocolType.IP);
                }
                else
                {
                    monitor_Socket = new Socket(AddressFamily.InterNetworkV6, SocketType.Raw,
System.Net.Sockets.ProtocolType.IP);
                }
                monitor_Socket.Bind(new IPEndPoint(ipAddress, 0));
                monitor_Socket.IOControl(SIO_RCVALL, BitConverter.GetBytes((int)1), null);
                monitor_Socket.BeginReceive(buffer, 0, buffer.Length, SocketFlags.None, new
AsyncCallback(this.OnReceive), null);
            }
            catch (Exception e)
            {
                monitor_Socket.Close();
                monitor_Socket = null;
                Console.WriteLine(e.ToString());
            }
        }
    }

    public void stop()
    {
        if (monitor_Socket != null)
        {
            monitor_Socket.Close();
            monitor_Socket = null;
        }
    }

    private void OnReceive(IAsyncResult ar)
    {
        try
        {
            int len = monitor_Socket.EndReceive(ar);
            if (monitor_Socket != null)
            {
                byte[] receivedBuffer = new byte[len];
                Array.Copy(buffer, 0, receivedBuffer, 0, len);
                try
                {
```

```
                Packet packet = new Packet(receivedBuffer);
                PcapDotNet.Packets.Packet packetNew = new
PcapDotNet.Packets.Packet(receivedBuffer, DateTime.Now,
PcapDotNet.Packets.DataLinkKind.IpV4);
                OnNewPacket(packet);

            }
            catch (ArgumentNullException ane)
            {
                Console.WriteLine(ane.ToString());
            }
            catch (ArgumentException ae)
            {
                Console.WriteLine(ae.ToString());
            }
        }
        monitor_Socket.BeginReceive(buffer, 0, buffer.Length, SocketFlags.None, new
AsyncCallback(this.OnReceive), null);
      }
      catch
      {
        stop();
      }
    }

    protected void OnNewPacket(Packet p)
    {
      if (newPacketEventHandler != null)
      {
        newPacketEventHandler(this, p);
      }
    }

    public event NewPacketEventHandler newPacketEventHandler;

    public delegate void NewPacketEventHandler(Monitor monitor, Packet p);

    /// <summary>
    /// check whether the current user is an administrator or not;
    /// </summary>
    /// <returns></returns>
    private bool IsUserAnAdmin()
    {
      WindowsIdentity identity = WindowsIdentity.GetCurrent();
      WindowsPrincipal principal = new WindowsPrincipal(identity);
      return principal.IsInRole(WindowsBuiltInRole.Administrator);
    }

    [DllImport("advapi32.dll")]
    private extern static int AllocateAndInitializeSid(byte[] pIdentifierAuthority, byte
nSubAuthorityCount, int dwSubAuthority0, int dwSubAuthority1, int dwSubAuthority2, int
```

```
dwSubAuthority3, int dwSubAuthority4, int dwSubAuthority5, int dwSubAuthority6, int
dwSubAuthority7, out IntPtr pSid);


        [DllImport("advapi32.dll")]
        private extern static int CheckTokenMembership(IntPtr TokenHandle, IntPtr SidToCheck, ref
int IsMember);


        [DllImport("advapi32.dll")]
        private extern static IntPtr FreeSid(IntPtr pSid);
    }
}
```

Packet.cs

```
class Packet
    {
        private const int LineCount = 30;

        /// <summary>
        /// use enumeration to identify the protocol type;
        /// </summary>
        enum ProtocolType
        {
            GGP = 3,
            ICMP = 1,
            IDP = 22,
            IGMP = 2,
            IP = 4,
            ND = 77,
            PUP = 12,
            TCP = 6,
            UDP = 17,
            OTHERS = -1
        }

        /// <summary>
        /// the original packet sniffed in the underlying layer;
        /// </summary>
        private byte[] raw_Packet;

        /// <summary>
        /// the sniffed time;
        /// </summary>
        private DateTime dateTime;

        /// <summary>
```

```
/// protocol type of the packet;
/// </summary>
private ProtocolType protocolType;

private IPAddress src_IPAddress;

private IPAddress des_IPAddress;

private int src_Port;

private int des_Port;

private int totalLength;

private int headLength;

//just for test
public int HeadLength
{
   get
   {
      return headLength;
   }
}


/// <summary>
/// using the raw and current system time to initialize the packet;
/// </summary>
/// <param name="raw"></param>
/// <param name="time"></param>
public Packet(byte[] raw)
{
   ///all the following exceptions should be caught when invoking this constructor;
   if (raw == null)
      throw new ArgumentNullException();

   ///when the orginal length is less than 20, it must be wrong;
   if (raw.Length < 20)
      throw new ArgumentException();
   raw_Packet = raw;
   dateTime = DateTime.Now;///the time sniffed the packet;

                        ///get the headlength in the packet;
   headLength = (raw[0] & 0x0F) * 4;
   if ((raw[0] & 0x0F) < 5)
      throw new ArgumentException(); // header is wrong for the length is incorrect;

   ///the actual length is the same with the header length indicator?
   if ((raw[2] * 256 + raw[3]) != raw.Length)
      throw new ArgumentException(); // length is incorrect;
```

```
    ///get the type of the packet;
    if (Enum.IsDefined(typeof(ProtocolType), (int)raw[9]))
        protocolType = (ProtocolType)raw[9];
    else
        protocolType = ProtocolType.OTHERS;

    src_IPAddress = new IPAddress(BitConverter.ToUInt32(raw, 12));
    des_IPAddress = new IPAddress(BitConverter.ToUInt32(raw, 16));
    totalLength = raw[2] * 256 + raw[3];

    ///handle the TCP OR UDP in particular method;
    if (protocolType == ProtocolType.TCP || protocolType == ProtocolType.UDP)
    {
        src_Port = raw[headLength] * 256 + raw[headLength + 1];
        des_Port = raw[headLength + 2] * 256 + raw[headLength + 3];
        if (protocolType == ProtocolType.TCP)
        {
            headLength += 20;
        }
        else if (protocolType == ProtocolType.UDP)
        {
            headLength += 8;
        }
    }
    else
    {
        src_Port = -1;
        des_Port = -1;
    }

}

/// <summary>
/// the accessible source IP address;
/// </summary>
public string Src_IP
{
    get
    {
        return src_IPAddress.ToString();
        //return "127.0.0.1";
    }
}

/// <summary>
/// the accessible source PORT;
/// </summary>
public string Src_PORT
{
    get
```

```
    {
        if (src_Port != -1)
            return src_Port.ToString();
        else
            return "";
    }
}

/// <summary>
/// the accessible destination IP address;
/// </summary>
public string Des_IP
{
    get
    {
        return des_IPAddress.ToString();
    }
}

/// <summary>
/// the accessible destination PORT;
/// </summary>
public string Des_PORT
{
    get
    {
        if (des_Port != -1)
            return des_Port.ToString();
        else
            return "";
    }
}

/// <summary>
/// the type of the packet;
/// </summary>
public string Type
{
    get
    {
        return protocolType.ToString();
    }
}

/// <summary>
/// the total length of the packet;
/// </summary>
public int TotalLength
{
    get
    {
```

```
        return totalLength;
    }
}

/// <summary>
/// the visibly formatted time;
/// </summary>
public string Time
{
    get
    {
        return dateTime.ToLongTimeString();
    }
}

/// <summary>
/// return the hexdecimal format of the packet ignoring the headers in the underlying layers;
/// </summary>
/// <returns></returns>
public string getHexString()
{
    StringBuilder sb = new StringBuilder(raw_Packet.Length);
    for (int i = headLength; i < TotalLength; i += LineCount)
    {
        for (int j = i; j < TotalLength && j < i + LineCount; j++)
        {
            sb.Append(raw_Packet[j].ToString("X2") + " ");
        }
        sb.Append("\n");
    }
    return sb.ToString();
}

/// <summary>
/// return the character format of the packet ignoring the headers in the underlying layers;
/// </summary>
/// <returns></returns>
public string getCharString()
{

    StringBuilder sb = new StringBuilder();

    for (int i = this.HeadLength; i < TotalLength; i += LineCount)
    {
        for (int j = i; j < TotalLength && j < i + LineCount; j++)
        {
            if (raw_Packet[j] > 31 && raw_Packet[j] < 128)
                sb.Append((char)raw_Packet[j]);
            else
```

ДОДАТОК Ж
ПРОТОКОЛ ПЕРЕВІРКИ НАВЧАЛЬНОЇ (КВАЛІФІКАЦІЙНОЇ) РОБОТИ

Назва роботи:___Методи виявлення передавання прихованої інформації в службових заголовках протокольних блоків даних комп'ютерних мереж.

Тип роботи:_____магістерська кваліфікаційна робота _

(кваліфікаційна роботи, курсовий проект (робота), реферат, аналітичний огляд, інше (зазначити))

Підрозділ _____кафедра обчислювальної техніки, 1КІ-20м

(кафедра, факультет (інститут), навчальна група)

Науковий керівник _Захарченко С.М., проф. кафедри ОТ_____

(прізвище, ініціали, посада)

Показники звіту подібності

Plagiat.pl (StrikePlagiarism)		Unicheck	
КП1		Оригінальність	85,4
КП2			
Тривога/Білі знаки	/	Схожість	14,6

Аналіз звіту подібності (відмінити подібне)

- ✓ Запозичення, виявлення у роботі, оформлені коректно і не містять ознак плагіату.
- □ Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності і відсутності самостійності її автора. Робот направити на доопрацювання.
- □ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Заявляю, що ознайомлений(-на) з повним звітом подібності, який був згенерований Системою щодо роботи (додається)

Автор _____ __Сирцов С. Р. _____

(підпис) (прізвище, ініціали)

Опис прийнятого рішення

Ступінь оригінальності роботи відповідає вимогам, що висуваються до МКР_____
_____
____

Особа, відповідальна за перевірку_____ _Захарченко С.М.__

(підпис) (прізвище, ініціали)

Експерт _____ _____

(за потреби) (підпис) (прізвище, ініціали)