Державний торговельно-економічний університет

Кафедра комп’ютерних наук та інформаційних систем

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Розробка програмного додатку для дослідження параметрів комп’ютерної мережі з використанням .NET Framework»

Студента 4 курсу, 4 групи, спеціальності 122 «Комп’ютерні науки» _____ *підпис студента* Омельчук Дмитро Анатолійович

Науковий керівник кандидат педагогічних наук, доцент _____ *підпис керівника* Базурін Віталій Миколайович

Гарант освітньої програми кандидат технічних наук, доцент _____ *підпис керівника* Демідов Павло Георгійович

Київ 2024

Державний торговельно-економічний університет

Факультет інформаційних технологій
Кафедра комп'ютерних наук та інформаційних систем
Спеціальність 122 «Комп'ютерні науки»
Освітня програма «Комп'ютерні науки»

Затверджую
Зав. кафедри _____Пурський О.І.
«18» грудня 2023р.

Завдання
на випускну кваліфікаційну роботу студенту

Омельчуку Дмитру Анатолійовичу
(прізвище, ім'я, по батькові)

1. Тема випускної кваліфікаційної роботи
«Розробка програмного додатку для дослідження параметрів комп'ютерної мережі з використанням .NET Framework»
Затверджена наказом ректора від *«27» листопада 2023 р. № 4175*

2. Строк здачі студентом закінченої роботи *31 травня 2024 року*

3. Цільова установка та вихідні дані до роботи
Мета роботи: розробка програмного забезпечення для дослідження параметрів комп'ютерної мережі з використанням .NET Framework.
Об'єкт дослідження: комп'ютерні мережі та їх параметри, які впливають на ефективність та стабільність роботи мережевої інфраструктури.
Предмет дослідження: моделі, методи та інформаційні технології для аналізу і моніторингу параметрів комп'ютерної мережі.
4. Перелік графічного матеріалу_____
_____
_____
_____

5. Консультанти по роботі із зазначенням розділів, за якими здійснюється консультування:

Розділ	Консультант (прізвище, ініціали)	Підпис, дата	
		Завдання видав	Завдання прийняв

1	Базурін В.М	22.12.2023 р.	22.12.2023 р.
2	Базурін В.М	22.12.2023 р.	22.12.2023 р.
3	Базурін В.М	22.12.2023 р.	22.12.2023 р.

6. Зміст випускного кваліфікаційної роботи (перелік питань за кожним розділом)

ВСТУП

РОЗДІЛ 1. ТЕОРЕТИЧНИЙ АНАЛІЗ ПРОБЛЕМИ ДОСЛІДЖЕННЯ

1.1Аналіз стану проблеми

1.2. Порівняльний аналіз програм для тестування параметрів комп'ютерної мережі

1.3.Засоби .NET Framework для роботи з налаштуваннями мережі

РОЗДІЛ 2. Модель обчислень

2.1 Характеристика параметрів мережі для діагностування

2.2. Концепція та структура додатка для тестування комп'ютерної мережі

2.3. Вибір засобів розроблення програми для тестування комп'ютерної мережі

РОЗДІЛ 3. Програмна реалізації додатку для тестування параметрів комп'ютерної мережі

3.1. Вимоги до додатка

3.2 Розробка програмного додатку. Опис класів і методів

3.3. Інтерфейс додатка

ВИСНОВКИ

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

7. Календарний план виконання роботи

№ Пор.	Назва етапів випускної кваліфікаційної роботи	Строк виконання етапів роботи	
		За планом	фактично
1	2	3	4
1	*Вибір теми випускної кваліфікаційної роботи*	*05.10.2023*	*05.10.2023*
2	*Розробка та затвердження завдання на випускну кваліфікаційну роботу*	*18.12.2023*	*18.12.2023*
3	*Вступ*	*02.02.2024*	02.02.2024
4	*РОЗДІЛ 1. ТЕОРЕТИЧНИЙ АНАЛІЗ ПРОБЛЕМИ ДОСЛІДЖЕННЯ*	*26.02.2024*	*26.02.2024*
5	*РОЗДІЛ 2. МОДЕЛЬ ОБЧИСЛЕНЬ*	*05.04.2024*	*05.04.2024*
6	*РОЗДІЛ 3. ПРОГРАМНА РЕАЛІЗАЦІЯ ДОДАТКУ ДЛЯ ТЕСТУВАННЯ ПАРАМЕТРІВ КОМП'ЮТЕРНОЇ МЕРЕЖІ*	*10.05.2024*	*10.05.2024*

7	*Висновки*	*15.05.2024*	*15.05.2024*
8	*Здача випускної кваліфікаційної роботи на кафедру науковому керівнику*	*20.05.2024*	*20.05.2024*
9	*Попередній захист випускної кваліфікаційної роботи*	*27.05.2024*	*27.05.2024*
10	*Виправлення зауважень, зовнішнє рецензування випускної кваліфікаційної роботи*	*28.05.2024*	*28.05.2024*
12	*Представлення готової зшитої випускної кваліфікаційної роботи на кафедру*	*31.05.2024*	*31.05.2024*
13	*Публічний захист випускної кваліфікаційної роботи*	*За розкладом роботи ЕК*	

Дата видачі завдання «22» грудня 2023 р.

9. Керівник випускної кваліфікаційної роботи Базурін В.М
(прізвище, ініціали, підпис)

10. Гарант освітньої програми Демідов П.Г.
(прізвище, ініціали, підпис)

11. Завдання прийняв до виконання студент Омельчук Д.А
(прізвище, ініціали, підпис)

12. Відгук керівника випускної кваліфікаційної роботи

_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____

Керівник випускної кваліфікаційної роботи

_____
(підпис, дата)

13. Висновок про випускну кваліфікаційну роботу

Випускна кваліфікаційна робота студента Омельчук Д.А
(прізвище, ініціали)
може бути допущена до захисту в екзаменаційній комісії.

Гарант освітньої програми Демідов П.Г.
(підпис, прізвище, ініціали)

Завідувач кафедри Пурський О.І.
(підпис, прізвище, ініціали)

«_____»_____2024 р.

Анотація

Розробка програмного додатку .NET Framework. Зростаюча залежність від ефективного функціонування електронних інформаційних мереж у всіх сферах нашого життя створює нагальну потребу в інструментах для їх діагностики. Метою цієї дипломної роботи є розробка програмного додатку, спеціально призначеного для дослідження критичних параметрів мережі.Використовуючи потужну платформу .NET Framework, додаток дозволяє IT-фахівцям оцінювати пропускну здатність мережі, затримку і втрату пакетів. Цей аналіз необхідний для забезпечення оптимальної продуктивності мережі, особливо у сферах системного управління, мережевої взаємодії та безпеки.Робота включатиме огляд існуючих рішень для моніторингу мережі, розробку та впровадження нового інструменту та тестування його функціональності. Результати дослідження параметрів мережі допоможуть фахівцям виявити потенційні вузькі місця та вжити необхідних заходів для оптимізації роботи мережі.Ключові слова: комп'ютерна мережа, параметри мережі, .NET Framework, програмний додаток, дослідження мережі, пропускна здатність, затримка пакетів, втрата пакетів. Дослідження параметрів комп'ютерної мережі з використанням .NET Framework.

Anotation

Development of a software application The growing dependence on the efficient functioning of electronic information networks in all spheres of our lives creates an urgent need for tools to diagnose them. The aim of this thesis is to develop a software application specifically designed to investigate critical network parameters.Using the powerful .NET Framework platform, the application allows IT professionals to evaluate network bandwidth, latency and packet loss. This analysis is necessary to ensure optimal network performance, especially in the areas of system management, network interaction and security.The work will include reviewing existing network monitoring solutions, developing and implementing a new tool, and testing its functionality. The results of the study of network parameters will help

specialists identify potential bottlenecks and take the necessary measures to optimise network performance.Keywords: computer network, network parameters, .NET Framework, software application, network research, bandwidth, packet delay, packet loss.Research of computer network parameters using the .NET Framework.

ЗМІСТ

ВСТУП

Актуальність дослідження. Комп'ютерні мережі стали невід’ємною частиною сучасного світу, забезпечуючи доступ до інформації та ресурсам більшості людей. Складність яка росте і масштаб мереж створює потребу в ефективних інструментах їх дослідження та аналізу.

Під час вивчення комп’ютерних мереж часто виникає потреба у визначенні їх характеристик. Додатку, який широко застосовується з цією метою, немає, оскільки розробка програмного забезпечення навчального призначення, на жаль, не є популярною серед програмістів. Тому існують суперечності між об’єктивною необхідністю використовувати таку програму і відсутністю додатка. Саме на розв’язання цієї суперечності і спрямоване наше дослідження.

Тема: Розробка програмного засобу для тестування характеристик комп'ютерної мережі.

Мета роботи: Проаналізувати характеристики програмного забезпечення для тестування комп'ютерної мережі та розробити власний програмний засіб на платформі .NET Framework.

Задачі дослідження:

1.Проаналізувати стан проблеми в галузі тестування комп'ютерних мереж.

2.Вивчити та проаналізувати існуючі програмні засоби для тестування характеристик комп'ютерної мережі.

3.Розробити програмне забезпечення, що дозволяє проводити комплексне тестування мереж з використанням платформи .NET Framework.

4.Провести експериментальну перевірку працездатності та ефективності розробленого програмного забезпечення.

Об'єкт: Процес розробки програмного забезпечення для тестування комп'ютерної мережі.

Предмет: Структура та алгоритм додатка для дослідження параметрів комп’ютерної мережі.

Практичне значення результатів дослідження: Розроблений додаток буде корисним для викладачів та студентів, що вивчають комп'ютерні мережі, для практичного тестування параметрів мережі та набуття практичних навичок у цій галузі.

Випускна кваліфікаційна робота складається з теоретичної та практичної частин, що включає огляд літератури, аналіз існуючих додатків, розробку та тестування програмного забезпечення. Робота містить використані джерела із 29 найменувань, додатків і містить 42 сторінки основного тексту, 16 рисунків і 4 таблиці.

РОЗДІЛ 1.

ТЕОРЕТИЧНИЙ АНАЛІЗ ПРОБЛЕМИ ДОСЛІДЖЕННЯ

1.1Аналіз стану проблеми

На сучасному етапі розвитку суспільства важко уявити його існування без комп’ютерних мереж. Комп’ютерні мережі використовуються практично у всіх сферах людського життя. Однак, незважаючи на стрімкий розвиток мережевих технологій, виникають різні проблеми, які перешкоджають ефективній роботі мереж.

У минулому в традиційних мережах мережеві адміністратори стикалися з обмеженнями в динамічному налаштуванні параметрів мережі. Це обмежувало гнучкість управління мережею, особливо для гетерогенних додатків, що вимагають зв'язку в режимі реального часу. Традиційні мережі підтримують специфічні політики, впроваджені під час створення пристрою, таким чином обмежуючи гнучкість динамічного налаштування параметрів мережі [1].

Однією з сучасних технологій, яка долає ці обмеження, є програмно-визначена мережа (SDN). Ця технологія дозволяє ефективно управляти гетерогенними мережами, відокремлюючи площину управління від площини даних, роблячи реалізацію мережевих функцій незалежною. Взаємодія між різними площинами реалізується за допомогою набору протоколів на основі стандарту OpenFlow. Архітектура SDN складається з трьох рівнів - площини даних, площини управління та площини додатків - і включає два основні функціональні компоненти: контролери та комутатори. Контролери генерують правила потоку (інструкції з обробки пакетів та управління мережею), а комутатори перенаправляють трафік відповідно до цих правил. Зв'язок між контролером і комутаторами регулюється набором протоколів OpenFlow [2].

Однією з головних переваг SDN є те, що параметрами мережі можна гнучко керувати, що дозволяє мережі швидко адаптуватися до мінливих вимог.

Це робить платформу SDN економічно вигідною і легкою для впровадження в реальних мережах. Крім того, SDN дозволяє розробникам розробляти функції управління і додатки на мовах програмування високого рівня, що підвищує продуктивність і надійність мережі [3].

Однак архітектура SDN має і свої обмеження, такі як висока залежність від контролера, що може призвести до різних проблем [4]. Наприклад, SDN може стати мішенню для мережевих атак, таких як спуфінг, хакерські атаки та атаки на відмову в обслуговуванні (DoS). Крім того, існує ризик перевантаження трафіку контролерів, а також високе енергоспоживання мережевих пристроїв. Обмежена кількість контролерів з нерівномірними зв'язками з комутаторами може призвести до проблем з продуктивністю у великих мережах.

Для вирішення цих завдань необхідно розробляти нові рішення. Наприклад, впровадження ефективних технологій управління трафіком і зниження енергоспоживання допоможе підвищити продуктивність мережі, знизити операційні витрати і підвищити безпеку. Удосконалені методи балансування навантаження допоможуть рівномірно розподілити трафік між контролерами, уникнути заторів і поліпшити якість обслуговування користувачів.

Дослідження, проведене в Політехнічному інституті Турина, показує, як пандемія COVID-19 змінила звички користувачів, змусивши їх використовувати онлайн-сервіси для навчання і роботи. Це спричинило значні зміни в мережевому трафіку: вхідний трафік значно зменшився, тоді як вихідний трафік збільшився більш ніж удвічі для підтримки онлайн-навчання та співпраці. Інфраструктура електронного навчання університету має підтримувати понад 600 занять на день для більш ніж 16 000 студентів, генеруючи трафік до 1,5 Гбіт/с.

Таким чином, теоретичний аналіз проблеми дослідження показує, що, незважаючи на значний прогрес у розвитку мережевих технологій, існує багато проблем, які потребують подальшого дослідження та вирішення. Розробка нових методів та підходів до управління трафіком, зменшення

енергоспоживання та забезпечення безпеки є ключовими напрямками підвищення продуктивності комп'ютерних мереж у сучасному суспільстві.

З кожним днем вплив комп’ютерних мереж поширюється і все більше впливає на абсолютно всі аспекти життя. Існують різні проблеми, з якими стикаються користувачі під час дослідження комп’ютерних мереж:

- Складність мереж. Сучасні мережі складаються з великої кількості компонентів, протоколів і технологій, що ускладнює їх розуміння й аналіз.
- Потреба в моніторингу та діагностиці: мережевим адміністраторам потрібні інструменти для моніторингу трафіку, виявлення проблем продуктивності та безпеки, а також діагностики та усунення проблем.
- Кіберзагрози: зростання кількості кіберзагроз, таких як хакерство та зловмисне програмне забезпечення, вимагає використання інструментів для аналізу трафіку та виявлення підозрілої активності.
- Оптимізація мереж. Збільшення обсягів даних і трафіку вимагає оптимізації мережевих ресурсів для забезпечення максимальної продуктивності та ефективності [5].

На сьогоднішній день існує ряд рішень для дослідження комп'ютерних мереж, але вони мають певні недоліки [6]:

- Комерційні інструменти: вони можуть бути багатофункціональними, але дорогими та складними у використанні.
- Безкоштовні інструменти: вони часто мають обмежену функціональність і незручний інтерфейс.
- Внутрішній розвиток: це трудомісткий і дорогий процес, доступний не всім.
- Розробка програмного додатку для дослідження параметрів комп’ютерної мережі за допомогою .NET Framework може бути ефективним і доступним рішенням, яке має ряд переваг:

- Простота використання: інтерфейс користувача буде зрозумілим та інтуїтивно зрозумілим завдяки .NET Framework.
- Доступність: .NET Framework є безкоштовною платформою, що робить інструмент доступним для широкого кола користувачів.
- Гнучкість: інструмент можна розширити для підтримки різних протоколів і технологій.
- Інтеграція: інструмент можна інтегрувати з іншими мережевими системами моніторингу та управління.
- Автоматизація: завдання з дослідження мережі можна автоматизувати, щоб заощадити час і зусилля.

Таким чином, розробка програмного забезпечення для дослідження комп'ютерних мереж за допомогою .NET Framework може допомогти мережевим адміністраторам, інженерам і дослідникам у моніторингу, діагностиці, оптимізації та захисті комп'ютерних мереж.

У сучасному світі, де комп'ютерні мережі відіграють життєво важливу роль, тестування параметрів мережі стає необхідною процедурою для забезпечення їх ефективної роботи, надійності та безпеки.

Основні причини, чому тестування продуктивності мережі є важливим [7]:

1) Моніторинг продуктивності: Тестування дозволяє вимірювати і аналізувати ключові параметри мережі, такі як пропускна здатність, затримка пакетів, втрата пакетів і час відгуку.Ці дані дають чітке розуміння продуктивності мережі, що допомагає виявити вузькі місця і потенційні проблеми до того, як вони негативно вплинуть на користувачів або додатки.Наприклад, тестування може виявити перевантажені ділянки мережі, які спричиняють затримки в обслуговуванні користувачів.

2) Усунення несправностей: Тестування є ефективним інструментом для діагностики та вирішення мережевих проблем.Аналізуючи результати тестування, можна визначити причини таких проблем, як несправні кабелі, перевантажені маршрутизатори, неправильно налаштовані протоколи або

вірусні інфекції. Швидке усунення несправностей мінімізує час простою і забезпечує безперервність роботи мережі.

3) Підвищена безпека: Тестування мережі допомагає виявити потенційні вразливості та ризики безпеки.Наприклад, тестування може виявити, що мережа не використовує шифрування, що робить її вразливою до перехоплення даних.Виявлення та усунення таких вразливостей запобігає кіберзлочинам і захищає конфіденційну інформацію.

4) Оптимізація мережі: Результати тестів можна використовувати для налаштування та оптимізації мережі для підвищення її продуктивності.Наприклад, тести можуть вказати на необхідність коригування маршрутизації трафіку для більш ефективного використання пропускної здатності.Оптимізація мережі мінімізує витрати та максимізує віддачу від інвестицій в інфраструктуру мережі.

5) Планування та масштабування: Тестування надає цінну інформацію про поточні та прогнозовані вимоги до мережі.

Ці дані можна використовувати для планування майбутніх розширень мережі та забезпечення її здатності задовольнити зростаючі потреби.

Тестування також може допомогти уникнути проблем масштабування, таких як перевантаження мережі або нестача пропускної здатності.

6) Використання мережевого аналізу за допомогою .NET Framework. Мережевий аналіз за допомогою .NET Framework використовується в багатьох організаціях і галузях, де комп'ютерні мережі відіграють важливу роль. Ось лише кілька прикладів:

– ІТ-компанії. Мережеві адміністратори використовують інструменти на основі .NET Framework для моніторингу, діагностики, оптимізації та захисту мереж своїх клієнтів.

Розробники програмного забезпечення використовують .NET Framework для створення власних інструментів мережевого дослідження або для інтеграції можливостей мережевого дослідження у вже існуючі продукти.

–Підприємства. Мережеві інженери малих і великих підприємств використовують інструменти .NET Framework для обслуговування та оптимізації корпоративних мереж.

– Відділи кібербезпеки використовують .NET Framework для виявлення та усунення кіберзагроз і захисту конфіденційних даних.

–Інтернет-провайдери (ISP). Інтернет-провайдери використовують інструменти .NET Framework для забезпечення та оптимізації своїх мереж, щоб забезпечити найкращу якість обслуговування клієнтів..NET Framework також використовується для моніторингу та усунення несправностей в мережах ISP, мінімізуючи час простою та забезпечуючи високий рівень обслуговування клієнтів.

– Дослідницькі інституції. Дослідники використовують .NET Framework для розробки нових алгоритмів та протоколів для мережевих досліджень..NET Framework також використовується для аналізу великих обсягів даних про мережевий трафік, виявляючи нові закономірності та покращуючи наше розуміння мереж.

– Державні установи. Урядові органи використовують .NET Framework для захисту своїх мереж від кібератак..NET Framework також використовується для моніторингу Інтернету та виявлення шкідливого контенту [8].

Це лише кілька прикладів того, як можна використовувати .NET Framework для дослідження параметрів комп'ютерних мереж. Ця платформа є потужною та гнучкою, що дозволяє вирішувати широкий спектр завдань мережевих досліджень.

Важливо відзначити, що .NET Framework - не єдина платформа, яка використовується для мережевих досліджень [9]. Такі платформи, як Java і Python, також широко використовуються і мають свої переваги і недоліки. Вибір платформи залежить від конкретних потреб користувача.

Хоча дослідження параметрів комп'ютерних мереж за допомогою .NET Framework не є самостійною науковою дисципліною, багато вчених і розробників використовують .NET Framework для створення інструментів і

додатків для мережевих досліджень. .NET Framework пропонує ряд переваг, таких як простота використання, доступність, гнучкість, інтеграція з іншими системами та можливість автоматизації задач. Ці переваги роблять платформу привабливою для різних користувачів, включаючи IT-компанії, підприємства, інтернет-провайдерів, дослідницькі інституції та державні установи[10].Використання .NET Framework дозволяє забезпечити ефективне моніторинг та діагностику мереж, виявлення та усунення кіберзагроз, оптимізацію мережевих ресурсів та інтеграцію з іншими системами. Це робить платформу потужним інструментом для дослідження параметрів комп'ютерних мереж, який допомагає вирішувати широкий спектр завдань. Хоча існують і інші платформи, такі як Java і Python, вибір конкретної платформи залежить від потреб користувача.

Таким чином, дослідження параметрів комп'ютерних мереж за допомогою .NET Framework сприяє підвищенню ефективності та безпеки мереж, що в свою чергу забезпечує надійне та продуктивне функціонування сучасних інформаційних систем.

1.2. Порівняльний аналіз програм для тестування параметрів комп’ютерної мережі

Існує багато програм для керування конфігурацією комп'ютерної мережі, кожна з яких має свої переваги та недоліки. Вибір правильної програми для певних потреб може бути непростим завданням. При виборі програмного забезпечення для тестування комп'ютерних мереж важливо враховувати наступні фактори:

- Функціональність: Які можливості тестування пропонує програмне забезпечення? Які функції має програмне забезпечення для тестування мережі?

•Простота використання: чи є програмне забезпечення простим у використанні, інтуїтивно зрозумілим і зручним для навігації?

•Вартість: Скільки коштує програмне забезпечення?

•Підтримка: Який рівень підтримки пропонує розробник, і чи доступні навчальні посібники, документація та форуми спільноти?

•Сумісність: чи сумісне програмне забезпечення з вашою операційною системою та обладнанням?

Деякі популярні програми для контролю конфігурації комп'ютерної мережі:

Wireshark: Wireshark - це безкоштовний і багатофункціональний аналізатор трафіку, який використовується для перехоплення та аналізу мережевого трафіку. Він підтримує широкий спектр протоколів і функцій, що робить його популярним вибором для професіоналів [11]

Основні функції програми Wireshark:

- Захоплення трафіку в реальному часі та аналіз збережених файлів.
- Фільтрація відображуваних та захоплюваних пакетів.
- Декодування сотень мережевих протоколів.
- Аналіз трафіку, включаючи стеження за TCP-потоками та виявлення аномалій.
- Показ статистики використання протоколів, активності IP-адрес та кінцевих вузлів.
- Налаштування кольорових правил для розпізнавання типів трафіку.
- Експорт трафіку у різні формати для подальшого аналізу іншими інструментами.

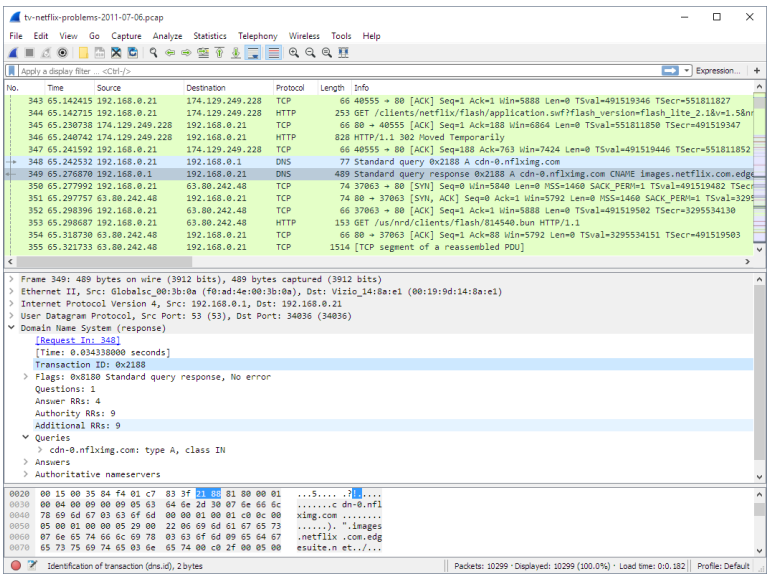

Рис. 1.1 Приклад роботи програми Wireshark

SolarWinds Network Performance Monitor - це платне програмне забезпечення для моніторингу та тестування мереж, яке пропонує широкий спектр функцій, включаючи тестування продуктивності, тестування QoS та тестування безпеки [12].

Основні функції програми SolarWinds Network Performance Monitor:

- Моніторинг стану мережевих пристроїв у реальному часі.
- Виявлення та відображення мережевої топології.
- Аналіз продуктивності мережі з використанням різних протоколів.
- Налаштування попереджень про мережеві проблеми та аномалії.
- Підтримка звітності про стан мережі та її продуктивність.
- Інтеграція з іншими інструментами для управління мережею.
- Підтримка різних типів мережевих пристроїв та інтерфейсів.

Рис. 1.2 Робота програми SolarWinds

PRTG Network Monitor - це платне програмне забезпечення для моніторингу та тестування мереж, яке пропонує широкий спектр функцій, включаючи тестування продуктивності, тестування доступності та тестування QoS.

Основні функції програми PRTG Network Monitor:

- Моніторинг стану мережевих пристроїв в реальному часі.
- Відстеження використання пропускної здатності.
- Моніторинг продуктивності серверів і додатків.
- Налаштування сповіщень про проблеми та аномалії.

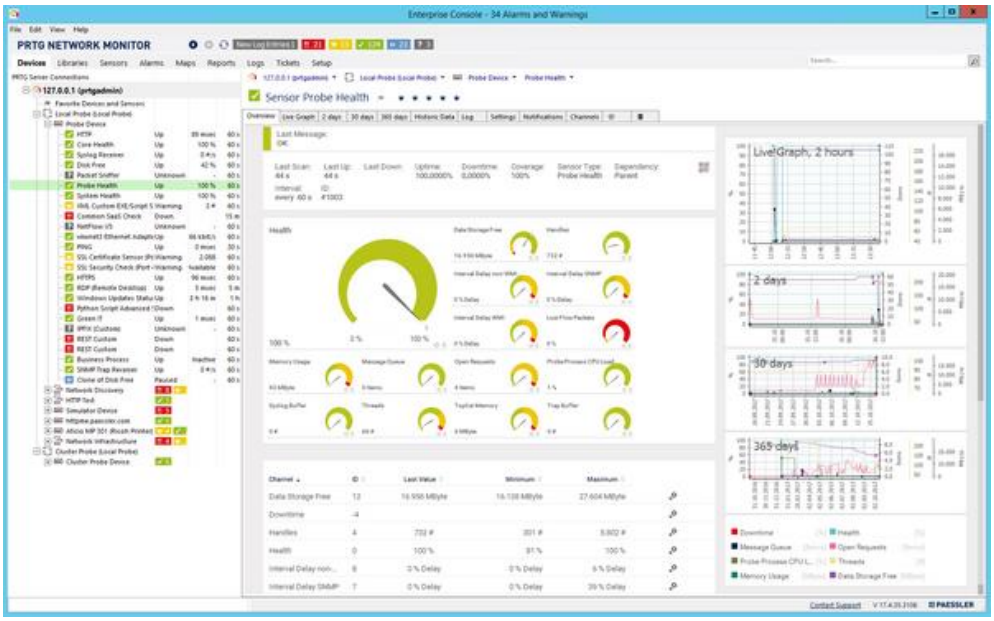

Рис.1.3 Робота програми PRTG Network Monitor

Colasoft Capsa Free - це безкоштовний аналізатор трафіку, який пропонує основні функції тестування мереж, такі як захоплення пакетів, аналіз трафіку та тестування продуктивності.

Основні функції програми Colasoft Capsa Free:

- Захоплення та аналіз мережевого трафіку в реальному часі.
- Підтримка аналізу понад 300 мережевих протоколів.
- Відображення мережевої топології та зв'язків.
- Моніторинг і аналіз мережевої продуктивності.
- Виявлення та аналіз мережевих аномалій і проблем.
- Налаштування фільтрів для точного аналізу трафіку.
- Генерація звітів про мережеву активність і продуктивність.

Рис. 1.4 Робота програми Colasoft Capsa Free

Nmap - це безкоштовний сканер мереж, який використовується для виявлення хостів і служб у мережі. Він є популярним інструментом для адміністраторів мереж та фахівців з безпеки.

Основні функції програми **Nmap:**

- Виявлення активних хостів у мережі.
- Визначення відкритих портів на хостах.
- Ідентифікація запущених сервісів і їхніх версій.
- Виявлення операційних систем хостів.
- Виявлення вразливостей і проблем безпеки.

•Виконання сценаріїв для розширеного аналізу (Nmap Scripting Engine).

•Генерація звітів про результати сканування.

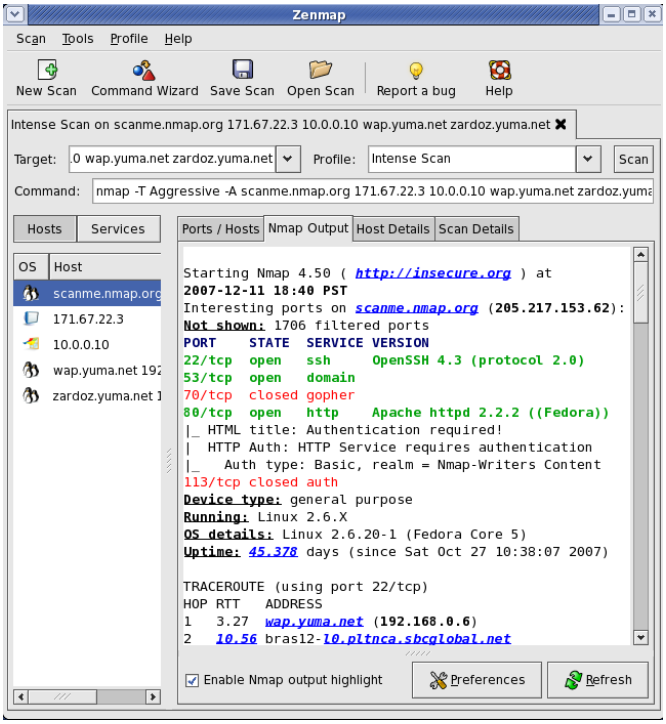

Рис. 1.5 Головне меню додатку Nmap

Порівняльний аналіз програм для тестування параметрів комп'ютерної мережі показує, що вибір програмного забезпечення залежить від конкретних потреб користувача. Кожна з розглянутих програм має свої переваги та недоліки, які можуть бути більш або менш важливими залежно від завдань.Враховуючи функціональність, простоту використання, вартість, підтримку та сумісність, можна зробити обґрунтований вибір, який найкраще відповідатиме вашим вимогам.

1.3. Засоби .NET Framework для роботи з налаштуваннями мережі

.NET Framework пропонує широкий спектр класів та бібліотек для роботи з налаштуваннями мережі [15]. Ось деякі з найпоширеніших:

1. System.Net: Цей простір імен містить класи для доступу до мережевих ресурсів, таких як веб-сайти, FTP-сервери та поштові сервери [16].

2. System.Net.Sockets: Цей простір імен містить класи для створення сокетних з'єднань, які використовуються для прямої комунікації з іншими мережевими пристроями.

3. System.Net.NetworkInformation: Цей простір імен містить класи для отримання інформації про мережеві інтерфейси та з'єднання.

4. System.Net.Web: Цей простір імен містить класи для створення веб-запитів та отримання веб-відповідей.

5. System.Security.Cryptography: Цей простір імен містить класи для шифрування та дешифрування даних, що передаються через мережу [17].

Приклади можливих задач:

- Пінгувати веб-сайт:
- Завантажити веб-сторінку:
- Надіслати HTTP-запит:

Це лише кілька прикладів того, як можна використовувати .NET Framework для роботи з налаштуваннями мережі. .NET Framework пропонує широкий спектр інших класів та бібліотек, які можна використовувати для виконання різних завдань, пов'язаних з мережею.

Окрім .NET Framework, існує багато інших бібліотек та інструментів, які можна використовувати для роботи з налаштуваннями мережі.

.NET Framework надає багатий набір класів і бібліотек для роботи з мережевими конфігураціями, що дозволяє ефективно виконувати різноманітні мережеві завдання. Основні простори імен, такі як System.Net, System.Net.Sockets, System.Net.NetworkInformation, System.Net.Web і System.Security.Cryptography, забезпечують доступ до мережевих ресурсів, створення сокетних з'єднань, отримання інформації про мережеві інтерфейси, виконання веб-запитів і шифрування даних.

Використання .NET Framework дозволяє реалізувати такі завдання, як пінг веб-сайтів, завантаження веб-сторінок і відправлення HTTP-запитів. Це робить

його потужним інструментом для роботи з мережевими конфігураціями. На додаток до .NET Framework існують інші бібліотеки та інструменти, які також можуть бути корисними для мережевих завдань, надаючи користувачам широкий спектр інструментів для ефективного вирішення їхніх потреб.

У ході дослідження представлено теоретичний аналіз проблеми дослідження. Аналіз стану проблеми показує, що зростаюча складність комп'ютерних мереж вимагає ефективних засобів моніторингу та перевірки параметрів. Сучасні дослідження підтверджують актуальність розробки нових методів оптимізації управління мережами.Порівняльний аналіз програмного забезпечення для тестування мереж виявив існування широкого спектру програмних засобів, які відрізняються функціональністю, точністю та простотою використання. Однак багато з них не пропонують комплексного підходу до оцінки всіх важливих параметрів.Дослідження інструментів .NET Framework показало, що ця платформа надає розробникам потужний набір інструментів для створення додатків моніторингу та управління мережею. Використання .NET Framework дозволяє створювати гнучкі та масштабовані рішення для дослідження комп'ютерних мереж.

РОЗДІЛ 2.
МОДЕЛЬ ОБЧИСЛЕНЬ

2.1 Характеристика параметрів мережі для діагностування

Щоб ефективно діагностувати проблеми комп'ютерної мережі, важливо розуміти та аналізувати різні мережеві метрики. Ці параметри надають цінну інформацію про стан мережі, продуктивність і потенційні проблеми [18].

Ось деякі з ключових характеристик мережевих параметрів, які слід враховувати при діагностиці:

1. Пропускна здатність: Це максимальна швидкість, з якою дані можуть передаватися через мережу. Пропускна здатність зазвичай вимірюється в бітах на секунду (біт/с) або мегабітах на секунду (Мбіт/с). Низька пропускна здатність може призвести до повільного завантаження сторінок, кешування відео та інших проблем з продуктивністю.

2. Затримка: Це час, необхідний для передачі пакету даних від одного хоста до іншого в мережі. Зазвичай затримка вимірюється в мілісекундах (мс). Висока затримка може призвести до затримок у чаті, обриву голосових дзвінків та інших проблем в онлайн-іграх.

3. Втрата пакетів: це відсоток пакетів даних, втрачених під час передачі через мережу. Втрата пакетів може бути спричинена низкою факторів, таких як перевантаження мережі, несправні кабелі або проблеми з обладнанням. Високий рівень втрати пакетів може призвести до пошкоджених завантажень, переривання дзвінків та інших проблем зі з'єднанням.

4. Jitter: Це варіант затримки пакетів даних. Jitter може бути викликаний тими ж факторами, що й втрата пакетів. Високий рівень jitter

може призвести до проблем з якістю голосу, переривання відео та інших проблем з онлайн-іграми [19].

5. Коефіцієнт використання мережі: Це відсоток часу, протягом якого мережа використовується на повну потужність. Високий коефіцієнт використання мережі може призвести до перевантаження та зниження продуктивності.

6. Маршрутизація: Це процес направлення пакетів даних до їхнього призначення в мережі. Таблиці маршрутизації використовуються для зберігання інформації про те, як маршрутизувати трафік до різних мереж. Неправильні або застарілі таблиці маршрутизації можуть призвести до того, що трафік буде направлений неправильним шляхом, що може призвести до повільного завантаження сторінок та інших проблем з підключенням.

7. IP-адреса: Це унікальна адреса, яка використовується для ідентифікації пристрою в мережі. IP-адреси можуть бути статичними або динамічними. Статичні IP-адреси призначаються вручну, тоді як динамічні IP-адреси призначаються сервером DHCP. Неправильна або недійсна IP-адреса може призвести до того, що пристрій не зможе підключитися до мережі.

8. MAC-адреса: Це унікальна фізична адреса, яка використовується для ідентифікації мережевого адаптера пристрою. MAC-адреси використовуються для керування доступом до мережі та для вирішення проблем з підключенням [20].

9. DNS: Система доменних імен (DNS) використовується для перетворення доменних імен (наприклад, www.google.com) на IP-адреси (наприклад, 172.217.163.14). Неправильна конфігурація DNS може призвести до того, що користувачі не зможуть отримати доступ до веб-сайтів та інших онлайн-ресурсів [21].

10. Протоколи: Мережеві протоколи - це правила та стандарти, які використовуються для передачі даних через мережу. Існує багато різних типів мережевих протоколів, таких як TCP/IP, HTTP та FTP. Неправильна

конфігурація протоколів може призвести до проблем з підключенням, повільного завантаження сторінок та інших проблем з продуктивністю.

Аналізуючи ці характеристики параметрів мережі, можна діагностувати широкий спектр мережевих проблем. В першу чергу потрібно сказати про низьку пропускну здатність: Якщо повільне завантаження сторінок, буферизацію відео або повільну передачу файлів, це може бути ознакою низької пропускної здатності мережі. В такому разі можна запустити тест швидкості інтернету, щоб перевірити швидкість завантаження і вивантаження. Також важлива висока затримка: у випадку затримки в чаті, переривання голосових дзвінків або проблеми з онлайн-іграми, це може бути ознакою високої затримки. Тоді потрібно скористатися інструментом ping, щоб перевірити час, необхідний для надсилання пакету даних на інший хост і отримання відповіді.

Втрата пакетів і тремтіння: перервані завантаження, обірвані дзвінки або проблеми з онлайн-іграми можуть бути ознаками втрати пакетів або тремтіння. Інструмент ping може допомогти перевірити відсоток втрати пакетів.

Висока перевантаженість мережі: повільне завантаження сторінок, низька загальна продуктивність мережі або часті переривання можуть свідчити про високу перевантаженість мережі. Інструменти моніторингу мережі можуть бути використані для перевірки завантаження мережі та виявлення вузьких місць.

Неправильна маршрутизація: повільне завантаження сторінок або проблеми з підключенням до певних веб-сайтів чи сервісів можуть свідчити про проблеми з маршрутизацією в мережі.

Неправильна IP-адреса: Якщо пристрій не може підключитися до мережі, це може бути ознакою неправильної або недійсної IP-адреси. Можна перевірити IP-адресу пристрою в налаштуваннях мережі та порівняти її з очікуваною адресою.

Неправильна конфігурація DNS: Якщо пристрій не може отримати доступ до деяких веб-сайтів або онлайн-ресурсів, це може бути ознакою того, що DNS налаштовано неправильно.

Неправильна конфігурація протоколу: Проблеми зі з'єднанням, повільне завантаження сторінок або інші проблеми з продуктивністю можуть вказувати на те, що протоколи налаштовані неправильно. У таких випадках адміністратору мережі може знадобитися перевірити конфігурацію протоколу.

На додаток до цих характеристик, також важливо враховувати фізичний стан мережі. Пошкоджені кабелі, несправне обладнання та інші фізичні проблеми також можуть призвести до проблем у мережі.

Отже аналізуючи всі ці характеристики мережі, можна ефективно діагностувати широкий спектр мережевих проблем і вжити відповідних заходів для їх усунення. Важливо регулярно контролювати пропускну здатність, затримку, втрату пакетів, перевантаження мережі, маршрутизацію, IP-адреси, DNS і конфігурацію протоколів, а також фізичний стан мережі. Своєчасне виявлення та усунення проблем може значно підвищити продуктивність і стабільність роботи мережі, забезпечити безперебійний доступ до інтернет-ресурсів та якісні онлайн-сервіси.

2.2 Концепція та структура додатка для тестування комп’ютерної мережі

Призначення програми. Додаток NetworkInfoApp призначений для аналізу та моніторингу параметрів комп'ютерної мережі. Він дозволяє користувачеві отримати детальну інформацію про мережеві інтерфейси, таблицю маршрутизації, ARP-таблицю, DNS-запити, активні мережеві з'єднання та стан TCP/IP-портів. Крім того, програма пропонує можливість вимірювати швидкість передачі даних в мережі, що важливо для аналізу продуктивності та ефективності мережі.

Користувачі. Системні адміністратори: можуть використовувати програму для швидкого отримання детальної інформації про стан мережевих інтерфейсів та інших параметрів мережі, що допоможе виявити та усунути проблеми.

Мережеві інженери: можуть використовувати програму для аналізу мережевих з'єднань і вивчення маршрутизації та ARP-таблиці, що є важливими для конфігурації та оптимізації мережі.

Студенти та викладачі: можуть використовувати програму як навчальний інструмент для вивчення основ комп'ютерних мереж, мережевих протоколів та інтерфейсів.

Користувачі, які цікавляться моніторингом мережі: Програма може бути корисною для користувачів, які хочуть отримувати інформацію про якість мережевих з'єднань і швидкість передачі даних.

Застосування у навчанні. Додаток може відігравати важливу роль у навчальному процесі, надаючи наступні можливості:

Практичне вивчення мережевих технологій: додаток дозволяє студентам і користувачам дізнатися, як на практиці працюють мережеві інтерфейси, таблиці маршрутизації і ARP, а також як працюють DNS-запити і TCP/IP-з'єднання.

Аналіз продуктивності мережі: маючи можливість вимірювати швидкість передачі даних, студенти можуть навчитися аналізувати продуктивність мережі і розуміти фактори, що впливають на швидкість і якість з'єднання.

Вивчення мережевих інструментів: додаток пропонує можливість ознайомитися з різними інструментами та методами моніторингу мережі, що є важливим для майбутніх системних адміністраторів та мережевих інженерів.

Поглиблення знань про мережеві протоколи: вивчення та аналіз інформації, отриманої за допомогою програми, допомагає користувачеві краще зрозуміти, як працюють і взаємодіють мережеві протоколи.

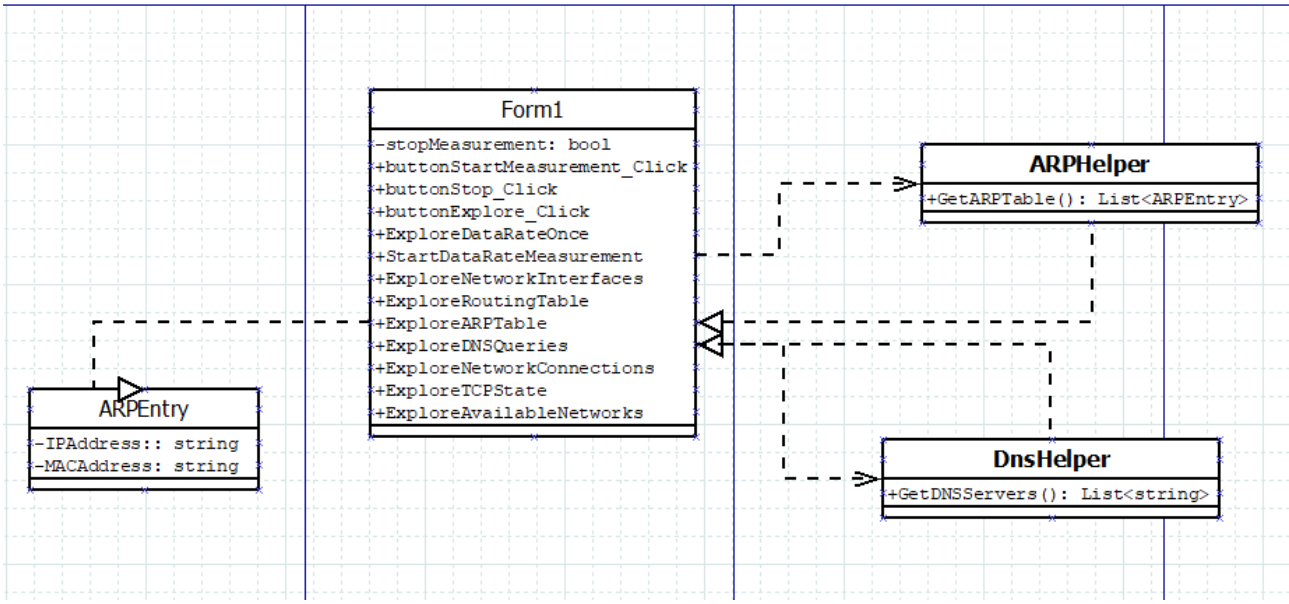

Рис.2.1 Form1

Таблиця 3.1 – Характеристика основних класів

Назва класу	Характеристика
Form1	Головний клас форми, що надає користувачам інтерфейс для взаємодії з програмою. Містить логіку для обробки подій, викликаних натисканням кнопок, та методи для дослідження різних аспектів комп'ютерної мережі.
ARPEntry	Клас, що представляє запис у ARP-таблиці. Містить властивості для збереження IP-адреси та MAC-адреси.
ARPHelper	Клас-помічник для отримання ARP-таблиці. Містить метод для збору та повернення списку записів ARP-таблиці у вигляді об'єктів ARPEntry.
DnsHelper	Клас-помічник для отримання DNS-серверів. Містить метод для збору та повернення списку DNS-серверів.

Таблиця 3.2 – Характеристика основних методів

Назва класу	Назва метода	Характеристика
Form1	buttonExplore_Click	Обробник події натискання кнопки, що викликає відповідний метод дослідження залежно від вибраного параметра в comboBox.
Form1	ExploreNetworkInterfaces	Досліджує і відображає інформацію про всі мережеві інтерфейси, включаючи ім'я, опис, тип, статус і швидкість.
Form1	ExploreRoutingTable	Відображає інформацію про таблицю маршрутизації, включаючи інформацію про шлюзи для всіх мережевих інтерфейсів.
Form1	ExploreARPTable	Досліджує і відображає ARP-таблицю, включаючи IP-адреси та MAC-адреси, використовуючи допоміжний клас ARPHelper.
Form1	ExploreDNSQueries	Відображає інформацію про DNS-запити, використовуючи допоміжний клас DnsHelper для збору DNS-серверів.
Form1	ExploreNetworkConnections	Відображає інформацію про активні мережеві підключення, включаючи локальні та віддалені адреси, порти та стан з'єднання.
Form1	ExploreTCPState	Відображає інформацію про стан TCP/IP портів, включаючи локальні та віддалені адреси, порти та стан з'єднання.

Form1	ExploreAvailableNetworks	Досліджує і відображає доступні мережі для кожного мережевого інтерфейсу, включаючи IP-адреси та підмережі.
Form1	buttonStartMeasurement_Click	Обробник події натискання кнопки, що запускає вимірювання швидкості передачі даних.
Form1	ExploreDataRateOnce	Запускає вимірювання швидкості передачі даних один раз і зупиняє його через 7 секунд.
Form1	StartDataRateMeasurement	Вимірює швидкість передачі даних для кожного мережевого інтерфейсу, відображаючи результати на форму кожну секунду до зупинки вимірювання.
ARPHelper	GetARPTable	Отримує і повертає список записів ARP-таблиці у вигляді об'єктів ARPEntry, використовуючи команду "arp -a" у процесі.
DnsHelper	GetDNSServers	Отримує і повертає список DNS-серверів для всіх мережевих інтерфейсів, використовуючи властивості IP-адрес інтерфейсів.

Основні класи, такі як Form1, ARPHelper, ARPEntry і DnsHelper, забезпечують взаємодію з користувачем через графічний інтерфейс і виконують різні мережеві завдання, такі як отримання інформації про мережеві інтерфейси, таблиці маршрутизації, ARP-таблиці, DNS-запити, мережеві з'єднання і стан портів TCP/IP. Програма також дозволяє вимірювати швидкість передачі даних.

NetworkInfoApp є корисним інструментом як для навчальних цілей, оскільки дозволяє студентам вивчати мережеві технології на практиці, так і для

практичного використання при моніторингу та діагностиці комп'ютерних мереж. Додаток сприяє поглибленню знань в області мережевих технологій і допомагає користувачам розуміти і аналізувати параметри і продуктивність мережі.

2.3 Вибір засобів розроблення програми для тестування комп’ютерної мережі

Під час вибору інструментів для розробки програми для тестування комп’ютерної мережі слід враховувати низку ключових факторів, таких як інтеграція з існуючими технологіями, потреби проекту, крос-платформна сумісність, продуктивність і доступність ресурсів.

Переваги C# : Інтеграція з .NET Framework, що дозволяє використовувати широкий спектр бібліотек та інструментів, висока продуктивність і ефективність у роботі з Windows, сильна підтримка від Microsoft і велика спільнота розробників, зручна робота з Windows Forms для створення графічного інтерфейсу.

Недоліки:В основному використовується для розробки додатків під Windows, менша популярність для кросплатформних рішень порівняно з Java або Python.

Переваги Java:Портативність та кросплатформеність завдяки JVM, високий рівень безпеки та стабільності, велика спільнота і велика кількість бібліотек та фреймворків [22-23].

Недоліки: відносно повільний запуск додатків через JVM, інколи складніше налаштування для роботи з системними функціями, порівняно з C# на Windows.

Переваги Python: простота і швидкість розробки завдяки читабельному синтаксису, велика кількість бібліотек для мережевого програмування (наприклад, scapy, socket), портативність і кросплатформеність [24-25].

Недоліки: менша продуктивність порівняно з C++ та Java, не підходить для додатків, що потребують дуже високої продуктивності.

Переваги C++:висока продуктивність і ефективність, можливість низькорівневого програмування та прямого доступу до системних ресурсів, широке використання у мережевому програмуванні та системному програмному забезпеченні.

Недоліки: складність і тривалість розробки через складніший синтаксис, відсутність автоматичного керування пам'яттю, що може призвести до помилок.

Вибір засобів розробки. Visual Studio переваги: підтримка широкого спектру мов програмування (C#, C++, Python), інтеграція з .NET Framework, потужні інструменти для дебагінгу та профілювання, підтримка розробки для різних платформ (Windows, веб, мобільні додатки) [27].

Недоліки: високі системні вимоги, платність для професійних версій.

SharpDevelop переваги: легковажний і безкоштовний IDE для C# та .NET, простота у використанні, підтримка основних функцій, необхідних для розробки додатків [28-29].

Недоліки:обмежена функціональність порівняно з Visual Studio, відсутність активної підтримки та оновлень.

Додаткові засоби

.NET Framework

Переваги:широкий набір бібліотек для різних типів завдань, висока продуктивність і оптимізація під Windows, підтримка з боку Microsoft та велика спільнота розробників.

Недоліки: основна спрямованість на Windows, можливі проблеми з кросплатформеністю, хоча .NET Core та .NET 5/6/7 вирішують ці питання.

Залежно від вимог і цілей проекту, можна зробити наступні висновки щодо розробки програми для тестування комп'ютерних мереж:

Отже,C# у поєднанні з Visual Studio та .NET Framework є найкращим вибором для розробки потужних та ефективних Windows-додатків.

Java буде корисною для крос-платформних рішень, але може мати нижчу продуктивність.

Python підходить для швидкої розробки та створення прототипів, але може бути недостатньо продуктивним для складних завдань.

C++ слід використовувати для додатків, де продуктивність і низькорівневий доступ до ресурсів є критично важливими, хоча це може підвищити складність розробки.

Вибір засобів розробки для програми тестування комп'ютерних мереж залежить від конкретних вимог проекту. .NET Framework (C#) пропонує інтеграцію з широким спектром бібліотек та інструментів, придатних для Windows-додатків. Java пропонує портативність і крос-платформну сумісність, але може працювати повільніше через використання JVM. Python перевершує за простотою і швидкістю розробки, але поступається в продуктивності. C++ пропонує високу продуктивність та ефективність, але є складнішим у розробці. Кожна з цих платформ має свої переваги та недоліки, які впливають на вибір розробників залежно від потреб проекту.

В цьому розділі було розроблено концепцію та структуру програми для тестування мережі, яка включає модулі для збору та аналізу даних про параметри мережі. Це дозволяє створити ефективний інструмент моніторингу та діагностики, що забезпечує успішне виявлення та вирішення проблеми.Охоплено різні засоби розробки програмного забезпечення, включаючи мови програмування, платформи та інструменти. Підібрано оптимальні засоби, що забезпечують ефективну реалізацію поставлених завдань, що дозволяє створити функціональний і зручний додаток для тестування комп’ютерних мереж, що відповідає сучасним вимогам і стандартам.

РОЗДІЛ 3.

ПРОГРАМНА РЕАЛІЗАЦІЯ ДОДАТКУ ДЛЯ ТЕСТУВАННЯ ПАРАМЕТРІВ КОМП'ЮТЕРНОЇ МЕРЕЖІ

3.1 Вимоги до додатка

Основні вимоги такі: функціональні, системні, вимоги до інтерфейсу.

Таблиця 3.3 – Функціональні вимоги додатка

Вимога	Опис	Характеристика
Перегляд мережевих інтерфейсів	Отримання та відображення інформації про всі доступні мережеві інтерфейси.	Ім'я інтерфейсу.Опис інтерфейсу.Тип інтерфейсу (Ethernet, Wireless тощо). Статус (працює, не працює).Швидкість (в кбіт/с)
Перегляд таблиці маршрутизації	Отримання та відображення поточної таблиці маршрутизації.	Шлюз.Інтерфейс маршруту.Мережевий адреса
Перегляд ARP-таблиці	Отримання та відображення ARP-таблиці.	IP-адреса.MAC-адреса
Перегляд DNS-запитів	Отримання та відображення списку DNS-серверів, які використовуються комп'ютером.	DNS-сервер (IP-адреса)
Перегляд мережевих підключень	Отримання та відображення інформації про активні	Локальна адреса.Локальний порт. Віддалена адреса. Віддалений порт.Стан

	мережеві підключення.	підключення (Established, Listen, TimeWait тощо)
Перегляд стану TCP/IP портів	Отримання та відображення стану TCP/IP портів.	Локальна адреса. Локальний порт.Віддалена адреса.Віддалений порт.Стан підключення
Перегляд доступних мереж	Отримання та відображення списку доступних бездротових мереж (Wi-Fi).	Ім'я інтерфейсу.Список доступних мереж (SSID)
Вимірювання швидкості передачі даних	Можливість вимірювати швидкість передачі даних через обраний мережевий інтерфейс і відображати результати вимірювання.	Ім'я інтерфейсу. Швидкість відправлення (в кбіт/с).Швидкість отримання (в кбіт/с). Якість з'єднання (Погана, Середня, Добра).Загальний час вимірювання (в секундах)

Таблиця 3.4 – Нефункціональні вимоги додатка

Нефункціональні вимоги	Опис	Характеристика
Кросплатформенність	Розробка для операційної системи Windows 10.	Windows 10
Продуктивність	Швидка обробка запитів користувача і відображення результатів.	Максимальний час відповіді для кожної операції - 3 секунди

	Максимальний час відповіді - 3 секунди.	
Інтерфейс користувача	Інтуїтивно зрозумілий графічний інтерфейс, реалізований з використанням Windows Forms.	Кнопки для запуску тестів та перегляду інформації. Комбіновані списки для вибору опцій. Текстові поля для відображення результатів. Повідомлення про помилки та успішне виконання операцій
Надійність	Механізми обробки виключень для стабільної роботи без неочікуваних збоїв. Обробка виключень для всіх критичних операцій.	Обробка виключень для всіх критичних операцій, логування помилок
Масштабованість	Архітектура додатка повинна дозволяти легко додавати нові функціональні можливості та розширення без значних змін в існуючому коді.	Використання модульної архітектури, відокремлення бізнес-логіки від інтерфейсу користувача
Безпека	Безпечна обробка даних, захист конфіденційної інформації від	Використання безпечних методів обробки даних, захист конфіденційної

	несанкціонованого доступу та маніпуляцій.	інформації
Документація	Супроводження детальною документацією для розробників та користувачів, включаючи інструкції по встановленню, використанню та налагодженню.	Приклади використання.Інструкції по встановленню та налаштуванню.Опис функціональних можливостей та їх використання

Програмний додаток для тестування параметрів комп'ютерної мережі повинен мати можливість перегляду інформації про мережеві інтерфейси, таблицю маршрутизації, ARP-таблицю, DNS-запити, мережеві підключення, стан TCP/IP портів та доступні бездротові мережі. Також, він повинен вимірювати швидкість передачі даних і мати кросплатформенний, продуктивний інтерфейс з надійними механізмами обробки виключень, масштабованою архітектурою та забезпеченням безпеки даних, а також супроводжуватись документацією для розробників та користувачів.

3.2 Розробка програмного додатку. Опис класів і методів

Form1 - це основний клас форми для програми Windows Forms. Він містить різні методи для вивчення інформації про мережу та обробки взаємодії користувачів через інтерфейс користувача. До методів Form1 належить:

- buttonExplore_Click(відправник об’єкта, EventArgs e):Обробник подій для натискання кнопки «Огляд». Він виконує різні методи на основі вибраного параметра у списку.

•ExploreNetworkInterfaces(): Збирає та відображає інформацію про мережеві інтерфейси, таку як назва, опис, тип, робочий стан і швидкість.

•ExploreRoutingTable(): Отримує та відображає таблицю маршрутизації, включаючи адреси шлюзів для кожного мережевого інтерфейсу.

•ExploreARPTable(): Отримує та відображає записи таблиці ARP, включаючи IP- та MAC-адреси за допомогою класу ARPHelper.

•ExploreDNSQueries(): Отримує та відображає адреси DNS-серверів за допомогою класу DnsHelper.

•ExploreNetworkConnections(): Отримує та відображає активні мережеві підключення, включаючи локальні та віддалені адреси, порти та стани підключення.

•ExploreTCPState(): Подібно до ExploreNetworkConnections(), він отримує та відображає стан портів TCP/IP.

•ExploreAvailableNetworks(): Асинхронно сканує доступні мережі, перевіряючи можливі IP-адреси в підмережі кожного мережевого інтерфейсу.

•buttonStartMeasurement_Click(відправник об’єкта, EventArgs e): Обробник події для запуску вимірювання швидкості передачі даних.

•ExploreDataRateOnce(): Починає вимірювання швидкості передачі даних і автоматично зупиняє його через 7 секунд.

•StartDataRateMeasurement():Вимірює швидкість передачі даних, обчислюючи різницю в надісланих і отриманих байтах з інтервалом в 1 секунду. Він оновлює форму з якістю та швидкістю з'єднання.

•buttonStop_Click(відправник об’єкта, EventArgs e): Обробник події для зупинки вимірювання швидкості передачі даних.

ARPentry – клас котрий представляє запис у таблиці ARP. До властивостей класу входять:

•IP-адреса запису.

•MAC-адреса запису.

ARPHelper – клас, який надає методи для отримання таблиці ARP. До методів цього класу входить:

•GetARPTable(): Отримує ARP-таблицю, виконавши команду arp -a та аналізуючи вихідні дані. Повертає список об’єктів ARPentry.

DnsHelper - надає методи для отримання адрес DNS-сервера.

•GetDNSServers(): Отримує адреси DNS-серверів для всіх мережевих інтерфейсів. Повертає список адрес DNS-серверів у вигляді рядків.

NetworkInfoApp — це програма Windows Forms, призначена для дослідження різних аспектів мережевої інформації. Програма містить функції для перегляду мережевих інтерфейсів, таблиць маршрутизації, ARP-таблиць, запитів DNS, мережевих підключень, станів портів TCP/IP, доступних мереж і швидкості передачі даних. Програма використовує обробники подій для реагування на дії користувача та асинхронні методи для ефективної обробки мережевих операцій.

3.3 Інтерфейс додатка

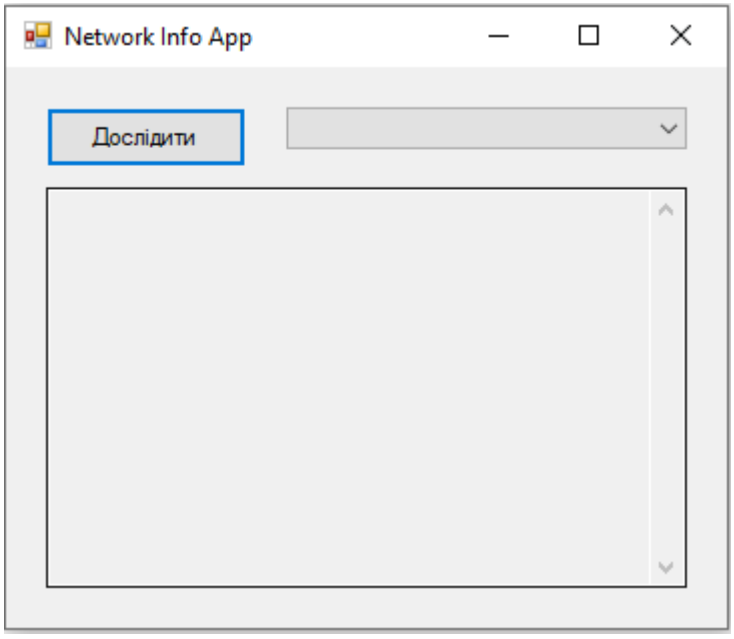

Рис.3.1 Головне меню програми

Згідно з зображенням, інтерфейс вашої програми "Network Info App" складається з наступних елементів:

Заголовок: Заголовок програми "Network Info App" розташований у верхній частині екрана. Він дає користувачам чітке уявлення про те, для чого призначена програма.

Навігаційна панель: Навігаційна панель розташована в нижній частині екрана. Вона містить кнопки, які дозволяють користувачам переходити до різних розділів програми. Кнопки мають чіткі та лаконічні підписи, які роблять їх легко зрозумілими для користувачів.

Область вмісту: Область вмісту розташована в центрі екрана. Вона використовується для відображення інформації, яка відповідає поточній вибраній кнопці навігаційної панелі. Наприклад, якщо вибрано кнопку "Мережеві інтерфейси", область вмісту відображатиме список мережевих інтерфейсів на пристрої.

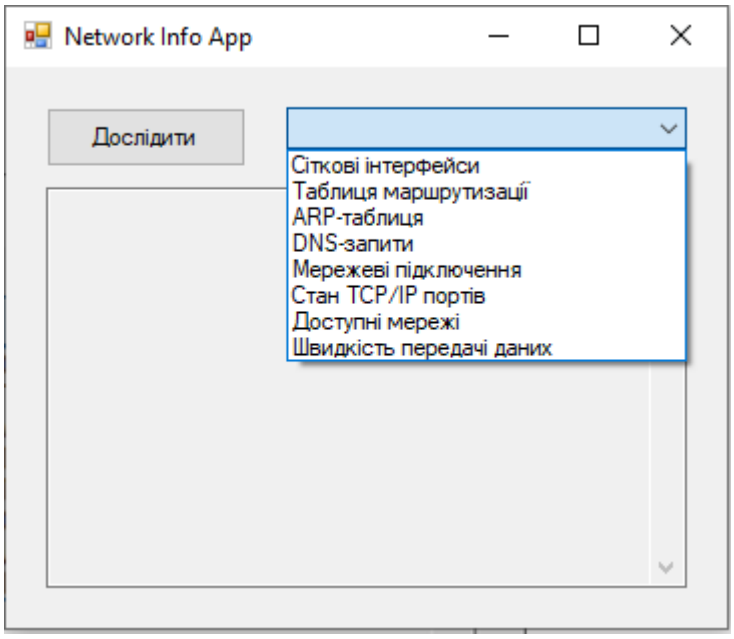

Рис 3.2 Вибір параметрів для дослідження

Програма має такі розділи:

Мережеві інтерфейси: Цей розділ відображає список мережевих інтерфейсів на пристрої. Для кожного інтерфейсу відображається така інформація, як ім'я, IP-адреса, MAC-адреса та стан.

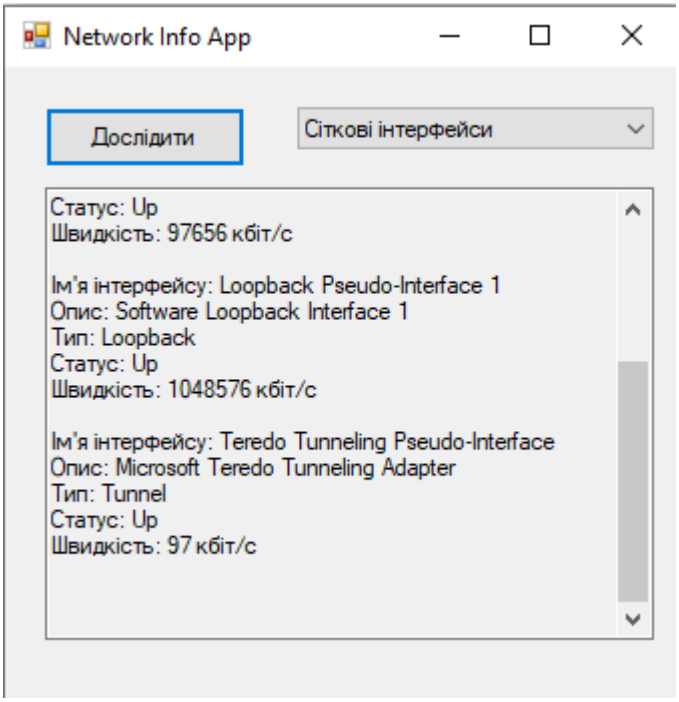

Рис.3.3 Вивід інформації про сіткові інтервейси

Таблиця маршрутизації: Цей розділ відображає таблицю маршрутизації пристрою. Таблиця маршрутизації використовується для визначення того, як маршрутизувати трафік до різних мереж.

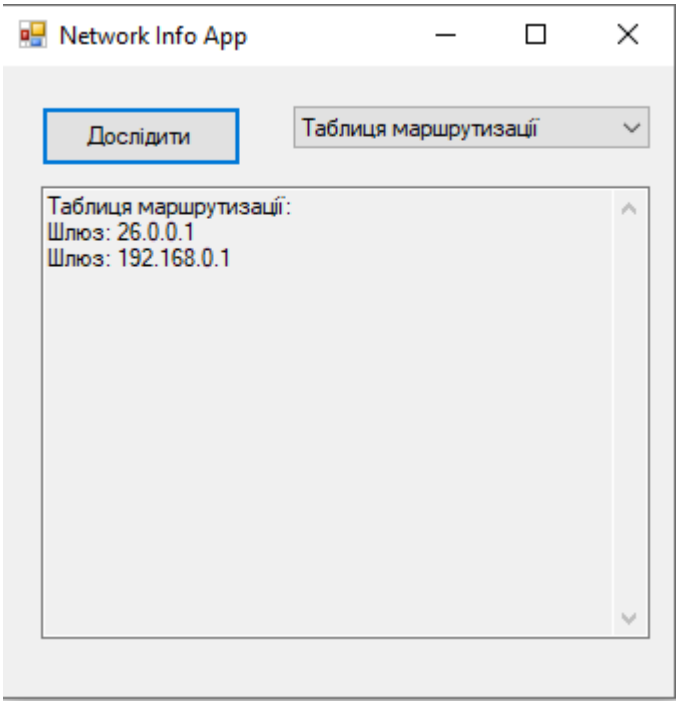

Рис.3.4 Вивід таблиці маршрутизації

ARP-таблиця: Цей розділ відображає ARP-таблицю пристрою. ARP-таблиця використовується для перетворення MAC-адрес на IP-адреси.

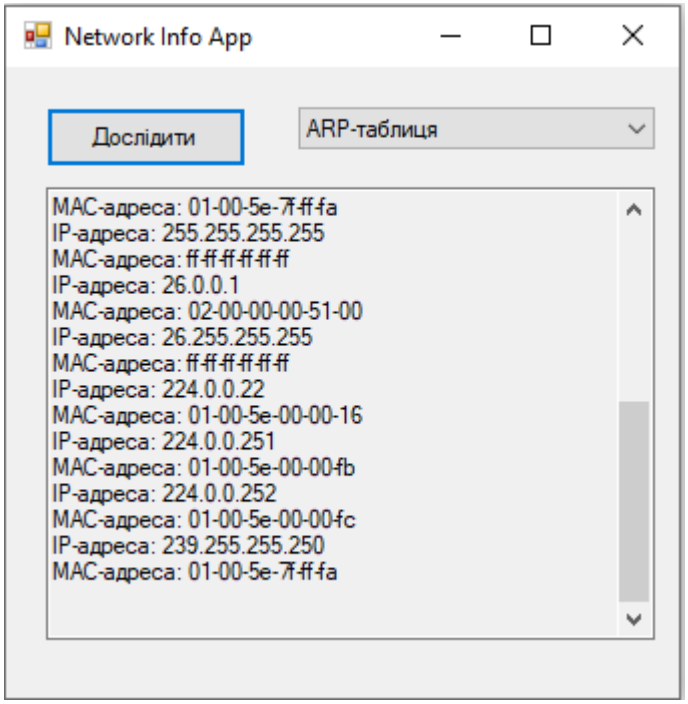

Рис.3.5 Вивід інформації про ARP-таблицю

DNS-запити: Цей розділ відображає список DNS-запитів, виконаних пристроєм. DNS-запити використовуються для перетворення доменних імен на IP-адреси.

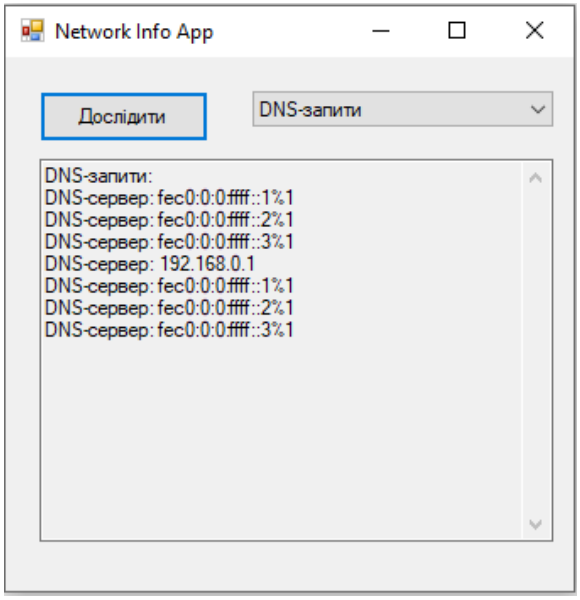

Рис.3.6 Вивід інформації про DNS-запити

Мережеві підключення: Цей розділ відображає список мережевих підключень до пристрою. Для кожного підключення відображається така інформація, як тип підключення, ім'я мережі та стан.

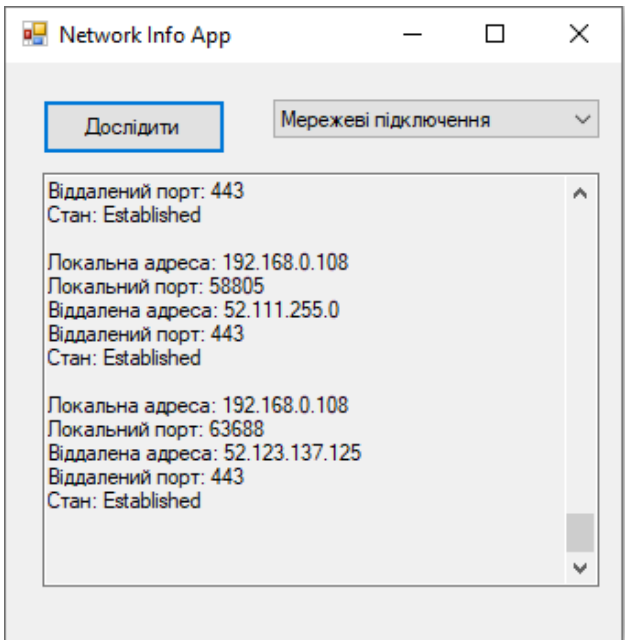

Рис.3.7 Вивід інформації про мережеві підключення

Стан TCP/IP портів: Цей розділ відображає список TCP/IP-портів, які використовуються на пристрої. Для кожного порту відображається така інформація, як номер порту, протокол, стан та процес, який використовує порт.

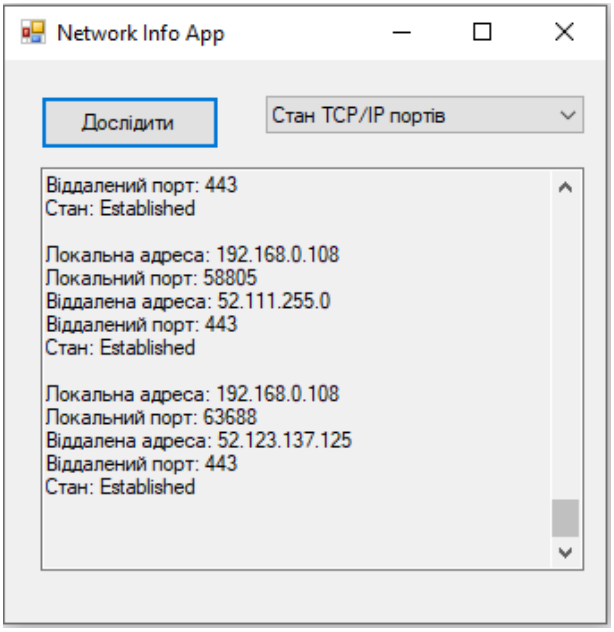

Рис.3.8 Стан TCP/IP портів

Доступні мережі: Цей розділ відображає список доступних мереж Wi-Fi.

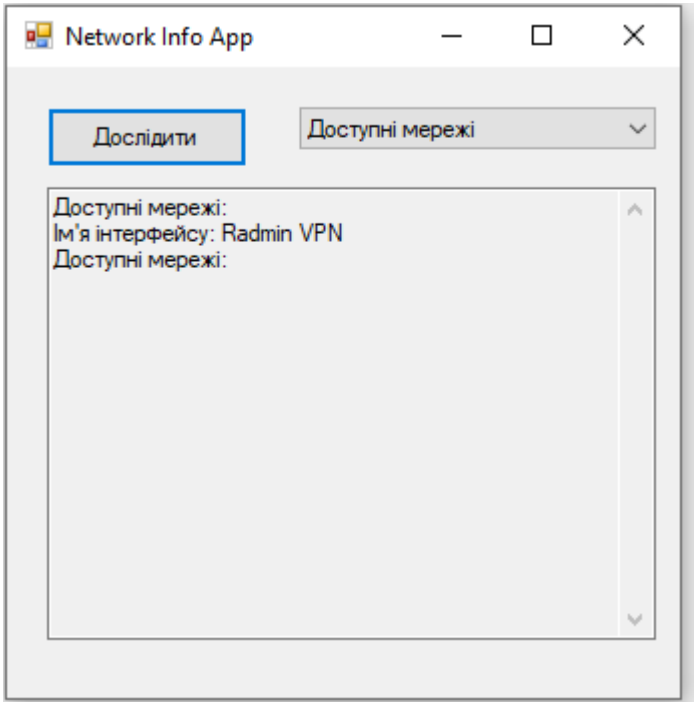

Рис.3.9 Доступні мережі

Швидкість передачі даних: Цей розділ відображає швидкість передачі даних для поточної мережі.

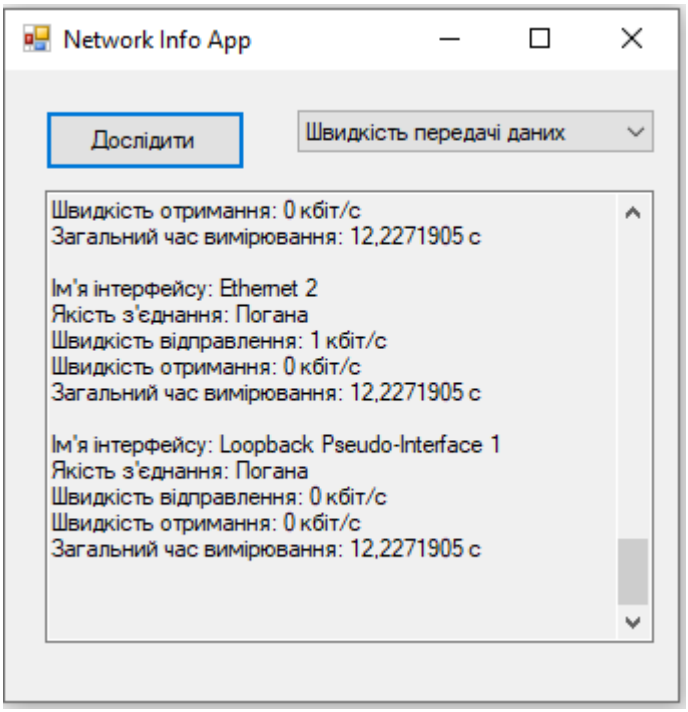

Рис.3.9 Швидкість передачі даних

Інтерфейс програми "Network Info App" є простим і зручним для користувачів. Навігаційна панель чітко позначена, а область вмісту відображає відповідну інформацію. Програма має широкий спектр функцій, які роблять її

цінним інструментом для будь-кого, хто хоче дізнатися більше про свою мережу.

У цьому розділі викладено вимоги до програм тестування комп’ютерної мережі, включаючи функціональні та нефункціональні аспекти зручності використання, ефективності та надійності.Опис класів і методів також надає інформацію про його структуру та функціональність, включаючи збір, аналіз і відображення інформації про мережеві інтерфейси, маршрутизацію, таблиці ARP, запити DNS, мережеві підключення, статус порту TCP/IP, доступні мережі та швидкість передачі даних. . Це забезпечує гнучкість і масштабованість надбудови.Інтерфейс розроблено з урахуванням зручності та інтуїтивності, що дозволяє легко отримувати доступ до необхідної інформації та функцій, підвищуючи ефективність використання надбудови. Можливе розширення функцій відповідно до нових вимог.

ВИСНОВКИ

У дипломній роботі представлені результати теоретичних і практичних досліджень, що полягають у розробці інформаційних технологій дослідження параметрів комп’ютерних мереж з метою підвищення ефективності їх управління та моніторингу. Результати прикладних досліджень стали основою для створення автоматизованої системи оцінки параметрів комп’ютерної мережі, включаючи мережеві інтерфейси, таблиці маршрутизації, ARP-таблиці, DNS-запити, мережеві з’єднання, стан портів TCP/IP, доступні мережі та дані.

1. На основі аналізу наукових джерел і поточного стану розвитку мережевих технологій встановлено, що додаток може стати корисним інструментом для мережевих адміністраторів та інженерів, які займаються моніторингом і підтримкою мережевої інфраструктури.Вона допомагає швидко ідентифікувати та вирішувати мережеві проблеми, також може бути використана як навчальний інструмент для демонстрації роботи мережевих протоколів і інтерфейсів., потрібно мати у своєму розпорядженні порівняно простий, але функціональний додаток, який буде визначати параметри комп'ютерної мережі: мережеві інтерфейси, таблиці маршрутизації, ARP-таблиці, DNS-запити, мережеві підключення , стан TCP/IP портів, доступні мережі, швидкість передачі даних.

2. Також було детально розглянуто основні параметри, що використовуються для діагностики комп'ютерних мереж, а також обґрунтовано необхідність їх моніторингу. Було визначено, що діагностика мережевих інтерфейсів, таблиць маршрутизації, ARP-таблиць, DNS-запитів, мережевих підключень, стану TCP/IP портів, доступних мереж та швидкості передачі даних є критично важливою для забезпечення надійного функціонування мережі.Було розглянуто різні засоби для розроблення програми, включаючи мови програмування, платформи та інструменти розробки. На основі аналізу було обрано оптимальні засоби, що забезпечують ефективну реалізацію поставлених завдань. Вибрані технології та інструменти дозволяють створити

функціональний та зручний у використанні додаток для тестування комп'ютерної мережі, що відповідає сучасним вимогам та стандартам.

3. Визначено та сформульовано вимоги до програмної надбудови для тестування комп’ютерної мережі. Ці вимоги включають функціональні та нефункціональні аспекти, які забезпечують зручність використання, продуктивність і надійність програмного забезпечення.Опис класів і методів програмної надбудови надав детальне відкриття про її внутрішню структуру та функціональність. Було відзначено основні класи та методи, прийняті для збору, аналізу та відображення інформації про мережеві інтерфейси, таблиці маршрутизації, таблиці ARP, DNS-запити, підключення до мережі, статус порту TCP/IP, доступні мережі та швидкість передачі даних. Такий підхід забезпечує гнучкість і масштабованість надбудови, що є місцем для її подальшого розвитку та вдосконалення.Інтерфейс програмного доповнення розроблено з урахуванням принципів зручності та інтуїтивності для користувачів. Основні елементи інтерфейсу дозволяють легко отримувати необхідну інформацію і додавати функції, що забезпечує ефективність його використання в різних сценаріях роботи з мережею. Також була врахована можливість розширення функціонального доповнення в майбутньому для відповідності новим вимогам і викликам.В умовах швидкого розвитку мережевих технологій та зростання вимог до якості мережевих підключень, такий додаток стає незамінним інструментом для забезпечення стабільної та ефективної роботи комп'ютерних мереж. Використання додатку сприяє підвищенню продуктивності та ефективності роботи в різних галузях, пов'язаних з комп'ютерними мережами. Таким чином, додаток є актуальним і необхідним рішенням для сучасних користувачів, які прагнуть мати надійний інструмент для моніторингу та діагностики мережевих підключень, що відповідає їхнім потребам і вимогам.

Список використаних джерел

1. Кириченко О.В. Проблеми динамічного налаштування параметрів мережі в сучасному суспільстві / О.В. Кириченко. - Київ: Інформаційні технології в освіті та науці, 2020. – С. 45-56.
2. Open Networking Foundation. (2012). Software-Defined Networking: The New Norm for Networks. - Режим доступу : https://www.opennetworking.org/sdn-resources/sdn-definition/
3. Основні принципи роботи мережі SDN та її переваги. - Режим доступу : https://prezi.com/pmmqvi9yumth/sdn/
4. Коваленко В. М. Обмеження архітектури програмно-визначених мереж та їх наслідки / В. М. Коваленко. - Вісник Національного технічного університету України "Київський політехнічний інститут". Серія: Радіоелектроніка, 2019. – № 58. – С. 112-121.
5. Гордієнко В. С. Оптимізація ресурсів комп'ютерних мереж для забезпечення продуктивності / В. С. Гордієнко. - Комп'ютерні мережі та телекомунікації. – 2020. – № 4. – С. 35-48.
6. Недоліки ряду рішень для дослідження комп’ютерних мереж. - Режим доступу : https://ela.kpi.ua/server/api/core/bitstreams/e0a0c843-a57d-4d82-8f42-0eba294bef1f/content
7. Чому тестування продуктивності мережі є важливим. - Режим доступу : https://training.qatestlab.com/blog/technical-articles/performance-testing/
8. Smith, J., & Brown, A. (2021). "Network Analysis using .NET Framework in Various Industries". *Journal of Network and Computer Applications*, 178, 102983.
9. .NET Framework. - Режим доступу: https://dotnet.microsoft.com/en-us/learn/dotnet/what-is-dotnet-framework
10. Переваги .NET Framework. - Режим доступу: https://www.tatvasoft.com/outsourcing/2021/07/advantages-of-net-framework.html

11. Що таке Wireshark. - Режим доступу: https://hackyourmom.com/kibervijna/wireshark-dokladnyj-posibnyk-z-pochatku-vykorystannya/

12. SolarWinds Network Performance Monitor. - Режим доступу: https://softprom.com/ru/vendor/solarwinds/product/solarwinds-network-performance-monitor-npm

13. Портативний мережевий аналізатор Capsa. - Режим доступу: https://www.colasoft.com/capsa/

14. Мережевий сканер Nmap. - Режим доступу: https://uaintech.blogspot.com/2013/02/nmap-1.html

15. Troelsen, A., & Japikse, P. (2017). Pro C# 7: With .NET and .NET Core (pp. 1025-1043). Apress. ISBN: 978-1484230176.

16. System.Net. - Режим доступу: https://learn.microsoft.com/ru-ru/dotnet/api/system.net?view=net-8.0

17. System.Security.Cryptography. - Режим доступу: https://learn.microsoft.com/ru-ru/dotnet/api/system.security.cryptography?view=net-8.0

18. Kurose, J. F., & Ross, K. W. (2017). Computer Networking: A Top-Down Approach (7th ed.). Pearson. С. 220-225.

19. Що таке Jitter. - Режим доступу: https://uk.wikipedia.org/wiki/%D0%94%D0%B6%D0%B8%D1%82%D0%B5%D1%80

20. Що таке MAC-адреса. - Режим доступу : https://2ip.ua/ua/blog/mac-address

21. Як працює DNS. - Режим доступу : https://hostiq.ua/blog/ukr/how-does-dns-work/

22. Переваги мови програмування Java. - Режим доступу: https://qagroup.com.ua/publications/benefits-of-java/

23. Віртуальна машина JWM. - Режим доступу: https://foxminded.ua/jvm-tse/

24. Lutz, M. (2019). Learning Python: Powerful Object-Oriented Programming (5th ed.). O'Reilly Media. С. 1-160.

25. Переваги мови Python. - Режим доступу : https://buki.com.ua/blogs/populiarnist-movi-programuvannia-python-pricini-ta-perevagi/

26. Advantages and Disadvantages of C++. - Режим доступу: https://www.geeksforgeeks.org/advantages-and-disadvantages-of-cpp/

27. Visual Studio description. - Режим доступу : https://uk.education-wiki.com/3958588-what-is-visual-studio-code

28. Hawkes, M. Beginning SharpDevelop for .NET Developers. Apress, 2009.

29. SharpDevelop. - Режим доступу: https://en.wikipedia.org/wiki/SharpDevelop#Features

Додатки

Додаток А) Програмний код

```
{
   partial class Form1
   {
      private System.ComponentModel.IContainer components = null;

      protected override void Dispose(bool disposing)
      {
         if (disposing && (components != null))
         {
            components.Dispose();
         }
         base.Dispose(disposing);
      }

      private void InitializeComponent()
      {
         this.buttonExplore = new System.Windows.Forms.Button();
         this.comboBoxOptions = new System.Windows.Forms.ComboBox();
         this.textBoxOutput = new System.Windows.Forms.TextBox();
         this.SuspendLayout();
         //
         // buttonExplore
         //
         this.buttonExplore.Location = new System.Drawing.Point(20, 20);
         this.buttonExplore.Name = "buttonExplore";
         this.buttonExplore.Size = new System.Drawing.Size(100, 30);
         this.buttonExplore.TabIndex = 0;
```

```
            this.buttonExplore.Text = "Дослідити";
            this.buttonExplore.Click += new
System.EventHandler(this.buttonExplore_Click);
            //
            // comboBoxOptions
            //
            this.comboBoxOptions.DropDownStyle =
System.Windows.Forms.ComboBoxStyle.DropDownList;
            this.comboBoxOptions.FormattingEnabled = true;
            this.comboBoxOptions.Items.AddRange(new object[] {
            "Сіткові інтерфейси",
            "Таблиця маршрутизації",
            "ARP-таблиця",
            "DNS-запити",
            "Мережеві підключення",
            "Стан TCP/IP портів",
            "Доступні мережі",
            "Швидкість передачі даних"});
            this.comboBoxOptions.Location = new System.Drawing.Point(146, 20);
            this.comboBoxOptions.Name = "comboBoxOptions";
            this.comboBoxOptions.Size = new System.Drawing.Size(178, 21);
            this.comboBoxOptions.TabIndex = 1;
            //
            // textBoxOutput
            //
            this.textBoxOutput.Location = new System.Drawing.Point(20, 60);
            this.textBoxOutput.Multiline = true;
            this.textBoxOutput.Name = "textBoxOutput";
            this.textBoxOutput.ReadOnly = true;
```

```
            this.textBoxOutput.ScrollBars =
System.Windows.Forms.ScrollBars.Vertical;
            this.textBoxOutput.Size = new System.Drawing.Size(304, 226);
            this.textBoxOutput.TabIndex = 2;
            //
            // Form1
            //
            this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
            this.ClientSize = new System.Drawing.Size(336, 315);
            this.Controls.Add(this.buttonExplore);
            this.Controls.Add(this.comboBoxOptions);
            this.Controls.Add(this.textBoxOutput);
            this.Name = "Form1";
            this.Text = "Network Info App";
            this.ResumeLayout(false);
            this.PerformLayout();

        }

        private System.Windows.Forms.Button buttonExplore;
        private System.Windows.Forms.ComboBox comboBoxOptions;
        private System.Windows.Forms.TextBox textBoxOutput;
    }
}
```

Додаток Б) Програмний код реалізації додатку

```
using System;
using System.Collections.Generic;
```

```
using System.Diagnostics;
using System.IO;
using System.Net.NetworkInformation;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace NetworkInfoApp
{
    public partial class Form1 : Form
    {
        private bool stopMeasurement = false;

        public Form1()
        {
            InitializeComponent();
        }

        private void buttonExplore_Click(object sender, EventArgs e)
        {
            string selectedOption = comboBoxOptions.SelectedItem.ToString();
            switch (selectedOption)
```

```
{
    case "Сіткові інтерфейси":
        ExploreNetworkInterfaces();
        break;
    case "Таблиця маршрутизації":
        ExploreRoutingTable();
        break;
    case "ARP-таблиця":
        ExploreARPTable();
        break;
    case "DNS-запити":
        ExploreDNSQueries();
        break;
    case "Мережеві підключення":
        ExploreNetworkConnections();
        break;
    case "Стан TCP/IP портів":
        ExploreTCPState();
        break;
    case "Доступні мережі":
        ExploreAvailableNetworks();
```

```
                break;
            case "Швидкість передачі даних":
                buttonStartMeasurement_Click(sender, e);
                break;
            default:
                break;
        }
    }

    private void ExploreNetworkInterfaces()
    {
        textBoxOutput.Clear();
        NetworkInterface[] networkInterfaces = NetworkInterface.GetAllNetworkInterfaces();
        foreach (NetworkInterface networkInterface in networkInterfaces)
        {
            textBoxOutput.AppendText($"Ім'я інтерфейсу: {networkInterface.Name}\r\n");
            textBoxOutput.AppendText($"Опис: {networkInterface.Description}\r\n");
            textBoxOutput.AppendText($"Тип: {networkInterface.NetworkInterfaceType}\r\n");
```

```
            textBoxOutput.AppendText($"Статус: {networkInterface.OperationalStatus}\r\n");
            textBoxOutput.AppendText($"Швидкість: {networkInterface.Speed / 1024} кбіт/с\r\n\r\n");
        }
    }

    private void ExploreRoutingTable()
    {
        textBoxOutput.Clear();
        textBoxOutput.AppendText("Таблиця маршрутизації:\r\n");

        foreach (NetworkInterface networkInterface in NetworkInterface.GetAllNetworkInterfaces())
        {
            IPInterfaceProperties ipProps = networkInterface.GetIPProperties();
            foreach (GatewayIPAddressInformation gateway in ipProps.GatewayAddresses)
            {
                textBoxOutput.AppendText($"Шлюз: {gateway.Address}\r\n");
            }
        }
```

```
        }

        private void ExploreARPTable()
        {
            textBoxOutput.Clear();
            textBoxOutput.AppendText("ARP-таблиця:\r\n");

            ARPHelper arpHelper = new ARPHelper();
            foreach (ARPEntry arpEntry in arpHelper.GetARPTable())
            {
                textBoxOutput.AppendText($"IP-адреса: {arpEntry.IPAddress}\r\n");
                textBoxOutput.AppendText($"MAC-адреса:
{arpEntry.MACAddress}\r\n");
            }
        }

        private void ExploreDNSQueries()
        {
            textBoxOutput.Clear();
            textBoxOutput.AppendText("DNS-запити:\r\n");
```

```
            DnsHelper dnsHelper = new DnsHelper();
            foreach (var dnsServer in dnsHelper.GetDNSServers())
            {
                textBoxOutput.AppendText($"DNS-сервер: {dnsServer}\r\n");
            }
        }

        private void ExploreNetworkConnections()
        {
            textBoxOutput.Clear();
            textBoxOutput.AppendText("Мережеві підключення:\r\n");

            IPGlobalProperties properties = IPGlobalProperties.GetIPGlobalProperties();
            TcpConnectionInformation[] tcpConnections = properties.GetActiveTcpConnections();
            foreach (TcpConnectionInformation tcpConnection in tcpConnections)
            {
                textBoxOutput.AppendText($"Локальна адреса: {tcpConnection.LocalEndPoint.Address}\r\n");
                textBoxOutput.AppendText($"Локальний порт: {tcpConnection.LocalEndPoint.Port}\r\n");
```

```
            textBoxOutput.AppendText($"Віддалена адреса: {tcpConnection.RemoteEndPoint.Address}\r\n");
            textBoxOutput.AppendText($"Віддалений порт: {tcpConnection.RemoteEndPoint.Port}\r\n");
            textBoxOutput.AppendText($"Стан: {tcpConnection.State}\r\n");
            textBoxOutput.AppendText("\r\n");
        }
    }

    private void ExploreTCPState()
    {
        textBoxOutput.Clear();
        textBoxOutput.AppendText("Стан TCP/IP портів:\r\n");

        IPGlobalProperties properties = IPGlobalProperties.GetIPGlobalProperties();
        TcpConnectionInformation[] tcpConnections = properties.GetActiveTcpConnections();
        foreach (TcpConnectionInformation tcpConnection in tcpConnections)
        {
            textBoxOutput.AppendText($"Локальна адреса: {tcpConnection.LocalEndPoint.Address}\r\n");
```

```
                textBoxOutput.AppendText($"Локальний порт: {tcpConnection.LocalEndPoint.Port}\r\n");
                textBoxOutput.AppendText($"Віддалена адреса: {tcpConnection.RemoteEndPoint.Address}\r\n");
                textBoxOutput.AppendText($"Віддалений порт: {tcpConnection.RemoteEndPoint.Port}\r\n");
                textBoxOutput.AppendText($"Стан: {tcpConnection.State}\r\n");
                textBoxOutput.AppendText("\r\n");
            }
        }

        private async void ExploreAvailableNetworks()
        {
            textBoxOutput.Clear();
            textBoxOutput.AppendText("Доступні мережі:\r\n");

            NetworkInterface[] networkInterfaces = NetworkInterface.GetAllNetworkInterfaces();
            foreach (NetworkInterface networkInterface in networkInterfaces)
            {
                textBoxOutput.AppendText($"Ім'я інтерфейсу: {networkInterface.Name}\r\n");
                textBoxOutput.AppendText("Доступні мережі:\r\n");
```

```
                IPInterfaceProperties ipProps = networkInterface.GetIPProperties();
                foreach (UnicastIPAddressInformation ip in ipProps.UnicastAddresses)
                {
                    if (ip.Address.AddressFamily == System.Net.Sockets.AddressFamily.InterNetwork)
                    {
                        string subnetMask = ip.IPv4Mask.ToString();
                        string[] ipParts = ip.Address.ToString().Split('.');
                        string[] maskParts = subnetMask.Split('.');
                        string baseIP = $"{ipParts[0]}.{ipParts[1]}.{ipParts[2]}";
                        string baseSubnet = $"{maskParts[0]}.{maskParts[1]}.{maskParts[2]}";

                        await Task.Run(async () =>
                        {
                            for (int i = 1; i < 255; i++)
                            {
                                string targetIP = $"{baseIP}.{i}";
                                Ping ping = new Ping();
                                PingReply reply = await ping.SendPingAsync(targetIP);
                                if (reply.Status == IPStatus.Success)
```

```
                    {
                        textBoxOutput.AppendText($"  {targetIP}\r\n");
                    }
                }
            });
        }
    }

    textBoxOutput.AppendText("\r\n");
  }
}

private async void buttonStartMeasurement_Click(object sender, EventArgs e)
{
  await ExploreDataRateOnce();
}

private async Task ExploreDataRateOnce()
{
  textBoxOutput.Clear();
```

```
            textBoxOutput.AppendText($"Вимірювання швидкості передачі даних...\r\n");

            await StartDataRateMeasurement();

            // Зупинка через 7 секунд
            await Task.Delay(7000);
            stopMeasurement = true;
        }

        private async Task StartDataRateMeasurement()
        {
            DateTime startTime = DateTime.Now;
            TimeSpan duration;

            while (!stopMeasurement)
            {
                duration = DateTime.Now - startTime;

                NetworkInterface[] networkInterfaces = NetworkInterface.GetAllNetworkInterfaces();
                foreach (NetworkInterface networkInterface in networkInterfaces)
```

```
{
    long bytesSent1 = networkInterface.GetIPv4Statistics().BytesSent;
    long bytesReceived1 = networkInterface.GetIPv4Statistics().BytesReceived;
    await Task.Delay(1000); // Затримка на 1 секунду
    long bytesSent2 = networkInterface.GetIPv4Statistics().BytesSent;
    long bytesReceived2 = networkInterface.GetIPv4Statistics().BytesReceived;

    long sentSpeed = ((bytesSent2 - bytesSent1) * 8) / 1024; // кбіт
    long receiveSpeed = ((bytesReceived2 - bytesReceived1) * 8) / 1024; // кбіт

    string quality = "Погана";
    if (sentSpeed > 5 && receiveSpeed > 5)
    {
       quality = "Добра";
    }
    else if (sentSpeed > 2 && receiveSpeed > 2)
    {
       quality = "Середня";
    }
```

```
                textBoxOutput.AppendText($"Ім'я інтерфейсу: {networkInterface.Name}\r\n");
                textBoxOutput.AppendText($"Якість з'єднання: {quality}\r\n");
                textBoxOutput.AppendText($"Швидкість відправлення: {sentSpeed} кбіт/с\r\n");
                textBoxOutput.AppendText($"Швидкість отримання: {receiveSpeed} кбіт/с\r\n");
                textBoxOutput.AppendText($"Загальний час вимірювання: {duration.TotalSeconds} с\r\n");
                textBoxOutput.AppendText("\r\n");
            }

            Application.DoEvents(); // Оновити вікно
        }
    }
    private void buttonStop_Click(object sender, EventArgs e)
    {
        stopMeasurement = true;
    }

    class ARPEntry
```

```
{
    public string IPAddress { get; set; }
    public string MACAddress { get; set; }
}

class ARPHelper
{
    public List<ARPEntry> GetARPTable()
    {
        List<ARPEntry> arpEntries = new List<ARPEntry>();
        try
        {
            ProcessStartInfo psi = new ProcessStartInfo("arp", "-a");
            psi.RedirectStandardOutput = true;
            psi.UseShellExecute = false;
            psi.CreateNoWindow = true;

            Process process = Process.Start(psi);
            StreamReader reader = process.StandardOutput;
            string line;
            while ((line = reader.ReadLine()) != null)
```

```
                {
                    if (!string.IsNullOrWhiteSpace(line) &&
!line.Contains("Інтерфейс"))
                    {
                        string[] parts = line.Split(new char[] { ' ' },
StringSplitOptions.RemoveEmptyEntries);
                        if (parts.Length == 3)
                        {
                            arpEntries.Add(new ARPEntry { IPAddress = parts[0],
MACAddress = parts[1] });
                        }
                    }
                }

                process.WaitForExit();
            }
            catch (Exception ex)
            {
                Console.WriteLine($"Помилка отримання ARP-таблиці:
{ex.Message}");
            }
```

```
            return arpEntries;
        }
    }

    class DnsHelper
    {
        public List<string> GetDNSServers()
        {
            List<string> dnsServers = new List<string>();
            try
            {
                NetworkInterface[] adapters = NetworkInterface.GetAllNetworkInterfaces();
                foreach (NetworkInterface adapter in adapters)
                {
                    IPInterfaceProperties ipProps = adapter.GetIPProperties();
                    foreach (var dns in ipProps.DnsAddresses)
                    {
                        dnsServers.Add(dns.ToString());
                    }
                }
```

```
                }
                catch (Exception ex)
                {
                    Console.WriteLine($"Помилка отримання DNS-серверів: {ex.Message}");
                }

                return dnsServers;
            }
        }
    }
}
```