Вінницький національний технічний університет

Факультет інформаційних технологій та комп'ютерної інженерії

Кафедра обчислювальної техніки

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему «**ПРОГРАМНИЙ ЗАСІБ ДЛЯ МОНІТОРИНГУ ТА КОНТРОЛЮ МЕРЕЖЕВОГО ТРАФІКУ МОБІЛЬНИХ ПРИСТРОЇВ**»

Виконав: студент 2 курсу, групи 2КІ-21м
спеціальності 123 Комп'ютерна інженерія
Любецький С.І. _____
Керівник к.т.н., доц. каф. ОТ
Муращенко О.Г. _____
"11" 12 2023р
Опонент д.ф. (PhD), доц. каф. МБІС
Салієва О.В. _____
"14" 12 2023р.

Допущено до захисту
Завідувач кафедри ОТ
д.т.н., проф. Азаров О.Д.
«15» 12 2023 р.

Вінниця 2023

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
обчислювальної техніки
проф., д.т.н. О. Д. Азаров
_____
«_26__» _вересня_ 2023 р.

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
Любецькому Сергію Ігоровичу

1 Тема роботи «Програмний засіб для моніторингу та контролю мережевого трафіку мобільних пристроїв», керівник роботи Муращенко Олександр Геннадійович, к.т.н., доцент,затверджені наказом вищого навчального закладу від 18.09.223 року № 247.

2 Строк подання студентом роботи 9.12.2023 р.

3 Вихідні дані до роботи: бібліотека PCap, мови програмування C/C++, Java, .NET. Бібліотека libpcap для Unix-подібних систем і WinPcap для Microsoft Windows.

4 Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити): вступ, дослідження існуючих систем моніторингу та контролю мережевого трафіку, завдання проектування системи, розробка прототипу системи моніторингу та контролю мережного трафіку, дослідження програмного засобу моніторингу та контролю мережного трафіку, висновки.

5 Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень): існуючі технології СВВ, структурна схема системи аналізу мережного трафіку, стек технологій MEAN, діаграма класів.

6 Консультанти розділів роботи представлено в табл. 1.

Таблиця 1 — Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1, 2, 3,4	Муращенко О.Г., к.т.н., доц. каф. ОТ		
5	Небава М.І., к.е.н., проф. каф. ЕП ВМ		

7 Дата видачі завдання __.__._____ р.

8 Календарний план наведено в табл. 2.

Таблиця 2 — Календарний план

з/п	Назва етапів виконання магістерської роботи	Строк виконання етапів роботи	.
	Постановка мети та задач роботи	21.10.23	
	Класифікація засобів моніторингу та контролю	25.10-30.10.23	
	Технології виявлення аномальної діяльності	31.10-08.11.23	
	Проектування системи	09.11-15.11.23	
	Розробка прототипу системи моніторингу мережного трафіку	16.11-.20.1123	
	Розробка прототипу системи контролю мережного трафіку	21.11-25.11.23	
	Обґрунтування вибору програмних засобів	26.11-31.11.23	
	Дослідження програмного засобу моніторингу та контролю мережного трафіку	01.12-04.12.23	
	Розрахунок економічної частини роботи	01.12-04.12.23	
	Оформлення пояснювальної записки та ілюстративного матеріалу	05.12.23	
	Аналіз виконання роботи, висновки, додатки		
	Перевірка якості виконання магістерської роботи та усунення недоліків		

Студент _____ Любецький С.І.

Керівник роботи _____ Муращенко О.Г.

АНОТАЦІЯ

УДК 004.42

Любецький С.І.

Програмний засіб для моніторингу та контролю мережевого трафіку мобільних пристроїв. Магістерська кваліфікаційна робота зі спеціальності 123 — комп’ютерна інженерія, освітня програма — комп’ютерна інженерія. Вінниця: ВНТУ, 2023, 82 с.

На укр.мові. Бібліогр.: 32 назви, рис. 25, табл. 11.

Дана магістерська кваліфікаційна робота присвячена створенню програмного засобу для моніторингу та контролю мережевого трафіку мобільних пристроїв

Підвищення рівня автоматизації процесів обробки, зберігання та передачі даних також відображається на появі проблем безпеки, а витрати на відшкодування збитків, завданих злочинцями, постійно зростають, при цьому спостерігається постійна тенденція до збільшення кількості атак на комп’ютерні системи та мережі.

Робота передбачає створення програмного забезпечення для аналізу внутрішнього трафіку, на основі якого можна вчасно вжити заходів і тим самим захистити пристрій і програми від впливу мережі.

Ключові слова: трафік, комп’ютерна мережа, програмне забезпечення, мобільні пристрої, моніторинг, контроль.

ANNOTATION

Lubetsky S.I.

A software tool for monitoring and controlling network traffic of mobile devices. Master's qualification route in the specialty 123 — computer engineering, educational program computer engineering. Vinnitsa, VTNU, 2023, 113 p.

In the Ukr. leng. Libr. name 28, figure 25, table 11.

This master's thesis is devoted to the creation of a software tool for monitoring and controlling the network traffic of mobile devices

The increasing level of automation of data processing, storage and transmission processes is also reflected in the emergence of security problems, and the costs of compensation for damages caused by criminals are constantly increasing, while there is a constant trend of increasing the number of attacks on computer systems and networks

The work involves the creation of software for internal traffic analysis, based on which measures can be taken in time and thereby protect the device and applications from the influence of the network.

Keywords: traffic, computer network, software, mobile devices, monitoring, control.

ЗМІСТ

ВСТУП

Швидка популярність і відповідний розвиток мереж зробили комп’ютерні системи дуже складними та взаємопов’язаними, роблячи їх менш захищеними від зловмисної діяльності. Підвищення рівня автоматизації процесів обробки, зберігання та передачі даних також відображається на появі проблем безпеки, а витрати на відшкодування збитків, завданих злочинцями, постійно зростають [1].

Слід зазначити, що спостерігається постійна тенденція до збільшення кількості атак на комп’ютерні системи та мережі [2-23]: методи та способи дистанційного впливу постійно вдосконалюються, а існуючі системи захисту не дозволяють своєчасно реагувати на зміни; це область, тому що ви повинні спочатку виявити, а потім дізнатися про мережеву атаку. Тому повністю видалити зловмисний трафік неможливо. У цих ситуаціях велика увага приділяється розробці ефективних методів виявлення трафіку та розробленню відповідних заходів захисту даних.

Актуальність обраної теми пов'язана з тим, що в даний час активно розробляються і використовуються різні методи контролю і моніторингу дорожнього руху, але на практиці вони не завжди ефективні. В результаті всі технології захисту постійно вивчаються і вдосконалюються.

Робота передбачає створення програмного забезпечення для аналізу внутрішнього трафіку, на основі якого можна вчасно вжити заходів і тим самим захистити пристрій і програми від впливу мережі.

Якщо ви розгорнете таку схему на кількох пристроях і мережах, безпека вашої мережевої інфраструктури буде виведена на новий рівень.

Метою є вдосконалення системи збору мережного трафіку мобільних пристроїв та розширення її функціональних можливостей для аналізу та виявлення несанкціонованої активності.

Для досягнення мети пропонується вирішити такі завдання:

— дослідити існуючі засоби моніторингу та контролю мережевого трафіку;

— обґрунтувати вибір інструментів розробки;

— розробити оптимальну архітектури системи;

— створити алгоритм обробки даних;

— реалізувати прототип програмної системи виявлення шкідливого трафіку з урахуванням розробленої архітектури та алгоритмів;

— провести тестування додатку.

Об’єктом дослідження є процес моніторингу та контролю мережевого трафіку мобільних пристроїв.

Предметом дослідження є методи та засоби моніторингу та контролю мережевого трафіку мобільних пристроїв.

Практична цінність полягає у можливості зменшення використання даних, збереженні заряду акумулятора та збільшені конфіденційності..

Наукова новизна полягає у вдосконаленні програмного засобу моніторингу та контролю мережевого трафіку мобільних пристроїв, що дозволяє забезпечити прості та розширені способи блокування доступу до Інтернет, а також додаткам і адресам можна окремо дозволити або заборонити доступ до Wi-Fi та/або мобільного з’єднання.

Апробацію результатів наукової роботи було проведено на науковій конференції «Молодь в науці: дослідження, проблеми, перспективи (МН–2024)», доповідь на тему “Моніторинг та контроль мережевого трафіку мобільних пристроїв з операційною системою Android”.

1 ДОСЛІДЖЕННЯ ІСНУЮЧИХ СИСТЕМ МОНІТОРИНГУ ТА КОНТРОЛЮ МЕРЕЖЕВОГО ТРАФІКУ

У процесі обробки та зберігання даних неминуче відбувається обмін інформацією між учасниками цього процесу. З кінця 1970-х років почався бурхливий розвиток комп'ютерних мереж і відповідного мережевого обладнання. Продовжують розвиватися локальні і глобальні мережі, з'являються нові протоколи передачі даних, розширюються апаратні можливості мережевих пристроїв, збільшується кількість підключених абонентів і загальний обсяг трафіку.

Такий інтенсивний розвиток регіону створює ряд проблем. Одна з них полягає в тому, що зі збільшенням кількості споживачів послуг передачі даних зростають вимоги до мережевого та серверного обладнання, яке використовується для підтримки належного рівня якості обслуговування. Останнє базується на необхідності захисту даних, що циркулюють у мережах і пристроях [24].

Для вирішення цих проблем використовуються контроль і моніторинг трафіку, які допомагають ефективно виявляти та вирішувати їх, коли вони виникають.

1.1 Класифікація засобів моніторингу та контролю

Засоби, що пропонуються для моніторингу та контролю, можна розділити на декілька груп [25]. Групи, про які йде мова, перераховані нижче.

Системи керування мережею — це централізовані програмні системи, які збирають інформацію про стан мережевих пристроїв і мережевий трафік. Функціональність цих програм не обмежується моніторингом та аналізом мережі. Крім того, в напівавтоматичному або автоматичному (залежно від реалізації) режимі виконуються керуючі дії в системі: установка і зміна адресних таблиць комутаторів та інших пристроїв, відкриття і закриття портів пристроїв. Системи цієї категорії включають HPOpenView, SunNetManager, IBMNetView.

Внутрішні системи діагностики та контролю (Internal Systems). Ці типи систем розроблені як програмні та апаратні модулі, вбудовані в комунікаційні пристрої, або операційні системи як програмні модулі. Він дозволяє контролювати та діагностувати лише ті пристрої, які розташовані. Прикладом такої системи є розподілений модуль керування концентратором 5000, який автоматично сегментує порти після виявлення несправностей, призначає порти внутрішнім сегментам концентратора тощо. Як правило, вбудовані модулі керування діють як агенти SNMP, передаючи інформацію про стан пристрою в систему керування.

Засоби системного адміністрування (Керування системою). Засоби цієї групи виконують функції, аналогічні функціям систем управління, але щодо інших об'єктів. У першому випадку об’єктом управління є програмно-апаратні засоби комп’ютерів мережі, а в другому – пристрої зв’язку. При цьому деякі функції цих типів систем можуть дублюватися (наприклад, засоби управління мережею можуть виконувати найпростіший аналіз трафіку).

Аналізатори протоколів (Protocol Analyzers) — це програмні або апаратні системи, які використовуються лише для моніторингу та аналізу мережевого трафіку. Хорошим аналізатором вважається той, який може підібрати та декодувати багато пакетів протоколів, що використовуються в мережах - можливо, десятки. Ця системна група може створювати деякі логічні умови для захоплення окремих пакетів і виконання повного кодування пакетів, тобто шляхом декодування вмісту кожного поля пакета вона може відображати гніздо пакетів різних протоколів у зручний спосіб.

Приступаючи до проектування або модернізації мережі, часто виникає необхідність визначити характеристики мережі: наприклад, затримки на різних етапах, частоту подій вибірки, інтенсивність потоку даних по лініях зв'язку та час відповіді на запити.

Обладнання для діагностики та сертифікації кабельних систем Призначення цієї групи зрозуміло з назви. Загалом можна виділити чотири типи таких

пристроїв: кабельні сканери, мережеві монітори, мультиметри та пристрої для гарантування кабельних систем.

Експертні системи поєднують людські знання для виявлення причин аномальної активності мережі та можливих способів відновлення робочого стану мережі. Часто представлені як окремі підсистеми інших інструментів моніторингу та аналізу, про які йшлося раніше.

Простий варіант експертної системи містить контекстно-залежну довідкову систему, а більш складний — бази знань з елементами штучного інтелекту. Прикладом цієї групи є вбудована система керування спектром Cabletron.

Багатофункціональні прилади для аналізу та діагностики. Через поширення локальних мереж виникла потреба в розробці недорогих портативних пристроїв із функціональністю кількох пристроїв: сканерів кабелю, програм керування мережею та аналізаторів протоколів. Приклади включають Comp від MicrotestInc або 675 LANMeter від FlukeCorp.

Варто відзначити ще два методи моніторингу мережі. Перший – з роутером. Моніторинг та інші засоби безпеки, вбудовані безпосередньо в маршрутизатор, не вимагають додаткового встановлення. Другий спосіб - це не роутер сам по собі, тобто професійний підбір необхідного обладнання та програмного забезпечення для поточних потреб.

1.2 Системи виявлення та запобігання вторгненням

Впровадження подібних систем захисту інформації є необхідністю всім серйозних мережевих інфраструктур, оскільки існують програми, які постійно вишукують уразливості у будь-якому устаткуванні, підключеному до глобальної мережі. Наприклад, пошуковий двигун Shodan в автоматичному режимі збирає інформацію про підключені пристрої, які не мають будь-якої частини системи безпеки. Користувачі Shodan знаходять системи керування , які не мають реквізитів доступу або вони налаштовані за умовчанням. Отже, до них можна легко проникнути та зменшити працездатність.

Проти такого впливу і спрямовані системи виявлення та запобігання вторгненням, тому вони є інструментом, що часто використовується в політиці безпеки [2-14].

Система виявлення вторгнень (СВВ) (англ. Intrusion Detection System (IDS)) – програмний чи апаратний засіб, призначений виявлення фактів неавторизованого доступу (вторгнення чи мережевої атаки) в комп'ютерну систему чи мережу.

Система запобігання вторгненням (СЗВ) (англ. Intrusion Prevention System (IPS)) — програмний або апаратний засіб, що здійснюємоніторинг мережі або системи в реальному часі з метою виявлення, запобігання або блокування шкідливої активності.

Системи запобігання вторгнень можна вважати розширенням систем виявлення вторгнень, оскільки завдання відстеження атак залишається однаковим. Але СЗВ має відслідковувати вторгнення в реальному часі і одразу здійснювати дії щодо запобігання атакам. Для цього вони використовують: скидання з'єднань, блокування потоків трафіку в мережі, видачу сигналів оператору. Крім цього, такі системи можуть дефрагментувати пакети, змінювати порядок TCP пакетів для захисту від пакетів зі зміненими SEQ і ACK номерами тощо [3].

Дані системи використовуються для автоматизації процесу контролю над подіями, що протікають у комп'ютерній системі чи мережі, та аналізу цих подій з метою пошуку ознак проблем безпеки. Оскільки кількість різних способів та видів організації несанкціонованих вторгнень у мережі останнім часом значно збільшилася, то системи виявлення вторгнень стали обов'язковою частиною безпекової інфраструктури для більшості організацій. Цьому сприяють як велика кількість літератури з цього питання, яку потенційні зловмисники уважно вивчають, так і більш витончені підходи до виявлення спроб проникнення в інформаційні системи.

Сучасні системи виявлення вторгнень мають різну архітектуру, основними з яких є: мережна та локальна. Мережеві системи встановлюють на виділених для

цього комп'ютерах так, щоб вони могли аналізувати трафік, що протікає по локальній обчислювальної мережі. Локальні системи розміщуються на тих пристроях, які потребують захисту, і вивчають певні події (програмні виклики або дії користувача).

Крім архітектури СВВ також можуть розрізняти за методикою виявлення: частина систем шукає аномальну поведінку, інша — зловмисну [7].

1.2.1 Методики виявлення аномальної та зловмисної поведінки

Системи виявлення аномальної поведінки (від англ. anomaly detection) засновані на тому, що СВВ знає ознаки, що характеризують правильну або допустиму поведінку об'єкта управління. «Нормальна» або «правильна» поведінка відноситься до дій об’єкта, які суперечать політиці безпеки.

Системи виявлення шкідливої поведінки (зловживання) засновані на передбаченні симптомів, що характеризують поведінку кривдника. Найпоширенішою реалізацією технології виявлення зловмисної поведінки є професійні системи (наприклад, Snort, RealSecure IDS, Enterasys Advanced Dragon IDS).

Розглянемо технології в цих системах більш детально (рис. 1.1) [24, 25]. Датчики аномалій виявляють незвичайну поведінку, яка називається аномаліями, у роботі певного об’єкта. Тому основні труднощі їх практичного використання пов'язані з нестабільністю об'єктів, що охороняються, а також їх взаємодією із зовнішніми об'єктами. Загалом, конкретний комп’ютер, мережева служба (наприклад, файловий сервер FTP), користувач тощо. може виступати об'єктом спостереження. Якщо атаки відрізняються від «нормальної» (легітимної) діяльності, спрацьовують датчики. Тут слід зазначити, що різні операції мають власне визначення прийнятного відхилення поведінки від дозволеного, а датчик моніторингу має власне визначення «порогу спрацьовування».

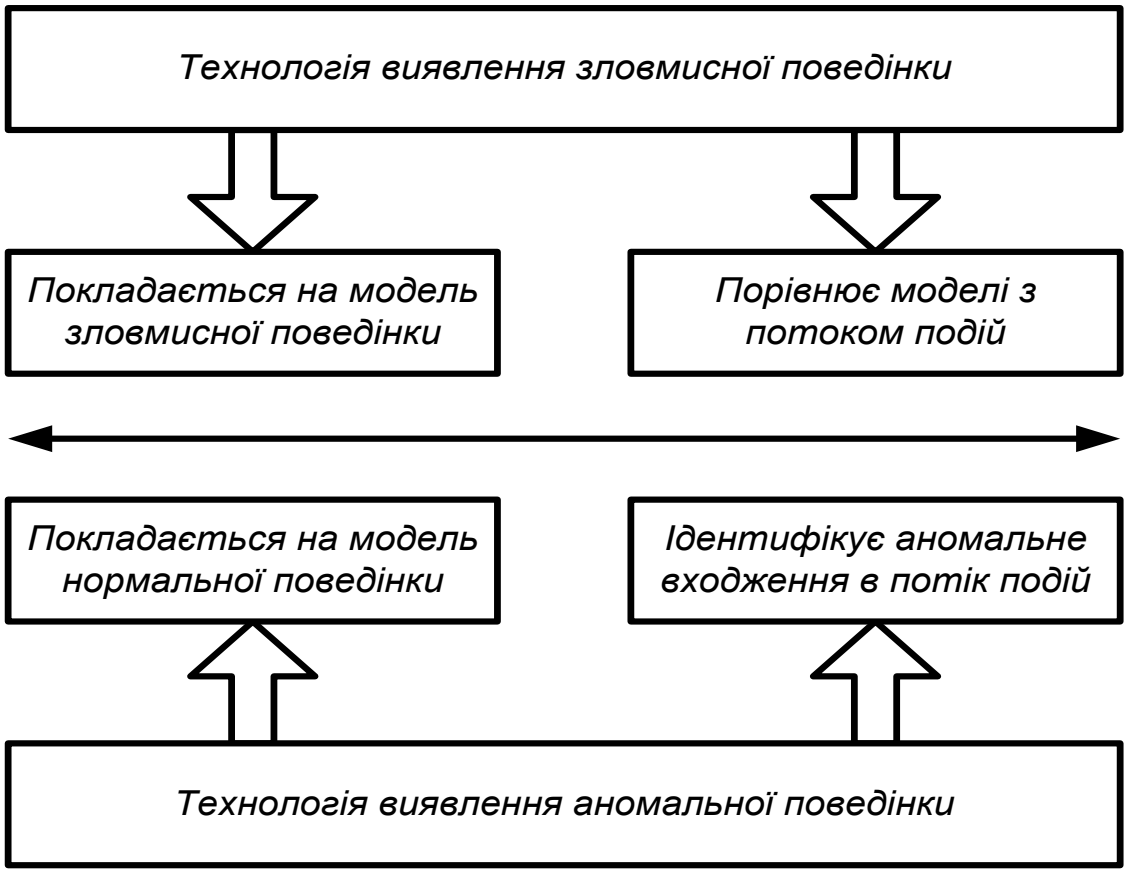

Рисунок 1.1 — Існуючі технології СВВ

1.2.2 Технології виявлення аномальної діяльності

Системи виявлення аномалій (від англ. anomaly detection) засновані на знанні симптомів, що характеризують правильну або допустиму поведінку об'єкта управління SBB. «Нормальна» або «правильна» поведінка відноситься до поведінки об'єкта, яка не відповідає політиці безпеки [3].

Системи виявлення зловмисної поведінки (зловживання) засновані на передбаченні характеристик поведінки правопорушника. Найпоширенішим типом технології виявлення зловмисної поведінки є професійні системи (наприклад, Snort, RealSecure IDS, Enterasys Advanced Dragon IDS).

Розглянемо технології в цих системах більш детально (рис. 1.1) [11].

Датчики аномалій виявляють незвичайну поведінку, яка називається аномаліями, у роботі певного об’єкта. Тому основні труднощі його практичного використання пов'язані з нестабільністю об'єктів, що захищаються, а також їх взаємодією із зовнішніми тілами. Як правило, конкретний комп’ютер, мережева

служба (наприклад, файловий сервер FTP), користувач тощо. може виступати об'єктом спостереження.

Якщо атаки відрізняються від «нормальної» (легітимної) діяльності, спрацьовують датчики. Тут слід зазначити, що є визначення допустимого відхилення поведінки від допустимого діапазону дій, а також визначення «порогу спрацьовування» датчика моніторингу.

Не потрібно оновлювати сигнатури та правила виявлення атак;

Можливість виявлення нових видів атак, сигнатур, які ще не розроблені;

Створення даних, які можна використовувати в системах виявлення шкідливих програм.

Недоліками цих систем є:

Створення багатьох помилок другого типу;

Вимагає тривалої та якісної підготовки;

Зазвичай він дуже повільний у роботі та вимагає великої кількості обчислювальних ресурсів.

1.2.3 Статистичний аналіз комп'ютерних атак

Використання методів статистичного аналізу є найпоширенішим застосуванням технології для виявлення аномальної поведінки. Статистичні датчики збирають різну інформацію про типову поведінку об'єкта і формують її у вигляді профілю. Профіль умов — це набір параметрів, які описують типову поведінку об’єкта. Він генерується на основі статистики спостережуваного об'єкта за допомогою методів математичної статистики (наприклад, методу ковзного вікна та методу зважених сум).

Спочатку відбувається початковий період генерації профілю, після чого рухи об’єкта порівнюються з відповідними параметрами, і при виявленні значних відхилень дається сигнал для початку атаки. Налаштування профілю можна організувати за загальними групами:

Або категоріальні параметри (імена файлів, команди користувача, відкриті порти тощо);

Поряд з попередніми параметрами, числові параметри, які не відповідають класифікації (кількість даних, що передаються за різними протоколами, навантаження на центральний процесор, кількість вхідних файлів і т.д.).

Існують також механізми для динамічної зміни профілів для більш повного опису поведінки об’єкта, що змінюється. Системи, що використовують статистичні методи, мають кілька переваг:

— не вимагають постійного оновлення бази даних сигнатур атак (це спрощує роботу з обслуговування систем);

— здатні адаптуватися до змін у поведінці користувачів і тому більш сприйнятливий до спроб вторгнення, ніж люди;

— невідомі сигнатури можуть виявляти невідомі атаки, які ще не були задокументовані, і таким чином слугувати своєрідним буфером, поки не буде розроблено відповідний шаблон для професійних систем;

— дозволяють виявляти більш складні атаки, ніж інші методи, наприклад об’єкти з розподілом у часі або атаки.

До недоліків систем виявлення вторгнень можна віднести:

— статистичні методи мають більшу ймовірність отримання помилкових звітів про атаку, ніж інші методи;

— складність визначення порогового значення (вибір цих значень не є тривіальним завданням, що вимагає глибоких знань контрольованої системи);

Статистичні методи не фіксують точно зміни в активності користувачів (наприклад, коли керівник виконує важливу підлеглу функцію), і ці недосконалі зміни відбуваються часто і можуть бути серйозною проблемою в організаціях. Помилкові звіти про загрози та негативні повідомлення відображаються як помилкові спрацьовування (пропущені атаки). Крім того:

— система може прийняти поведінку, що відповідає атаці, як нормальну, оскільки вона адаптується до нової поведінки, якщо зміни в рутині є поступовими;

— статистичні методи не здатні виявити напади суб’єктів, типову поведінку яких неможливо описати;

— статистичні методи повинні бути попередньо налаштовані (визначати порогові значення для кожного параметра, для кожного користувача).

Тільки системи, засновані на статистичних методах, не можуть з самого початку виявити атаки від суб'єктів, які виконують несанкціоновані дії, оскільки типова поведінка для них включає атаки, статистичні методи на основі профілю нечутливі до порядку маршрутизації подій.

Однак існують способи вирішення цих проблем, і це лише питання часу, коли вони будуть втілені в життя. Чиста реалізація технології аномальної поведінки статистичного методу очевидна. Статистичний метод успадковує всі практичні переваги технології виявлення аномалій [9].

1.3 Аналіз недоліків сучасних систем виявлення вторгнень

Враховуючи вищесказане, всі системи виявлення вторгнень можна розділити на системи виявлення:

Сигнати відомих атак;

Аномалії взаємодії контрольованих об'єктів;

Спотворення довідкової профільної інформації.

Наразі майже не існує гібридних систем, систем, які використовують розподілені в часі та просторі дані. У роботі переважної більшості сучасних систем техніка підпису використовується лише для розпізнавання ефектів зловмисника або лише для пошуку аномалій у поведінці контрольованої системи.

Крім того, більшість відомих систем не мають симулятора атаки чи інших засобів перевірки дійсності SBA, які б забезпечили простий і надійний засіб перевірки параметрів конфігурації, що використовуються в будь-якій комп’ютерній мережі.

З логічних причин цей інструмент має дозволити моделювати діяльність вірусного програмного забезпечення (наприклад, CodeRed, NetSky, Bagle, MSBlast), атак типу «відмова в обслуговуванні» (наприклад, SYN-storm або хак-атаки). , атаки, спрямовані на фіксацію привілеїв облікових записів (наприклад, можуть бути виявлені уразливості в мережевих службах MS SQL Server 2000, MS

Internet Information Service 5.0), атаки, спрямовані на перенаправлення трафіку та введення неправдивої інформації (заміна ARP і служба введення DNS). Бажано, щоб програмне забезпечення могло генерувати атаки розподіленого характеру.

Наприклад, архітектура деяких типів симуляторів SBA складається з різних типів агентів, що спеціалізуються на вирішенні підзадач виявлення вторгнень. Агенти розгортаються на окремих комп'ютерах системи. У цій архітектурі явно немає «центру контролю» для сімейства агентів, оскільки будь-який агент може стати лідером, ініціюючи співпрацю та функції управління залежно від ситуації. При необхідності їх можна перемістити або відключити в мережі та локальному середовищі. Залежно від ситуації (тип і кількість атак на комп'ютерні мережі, наявність обчислювальних ресурсів для виконання функцій захисту) може знадобитися створення кількох екземплярів представників кожного класу. Очікується, що архітектура системи адаптується до конфігурації мережі, змін трафіку та нових типів атак, використовуючи накопичений досвід.

Багатоагентні системи є цікавою розробкою, але внутрішні операції не розкривають алгоритми, які використовуються або розроблені для виявлення атак. Крім того, поточні версії певних емуляторів не працюють у режимі реального часу (оскільки вони не дозволяють створювати вибрані основні набори інструментів).

Загалом відсутність симуляторів атак для оцінки ефективності SBA не є основною проблемою в цій галузі. Реальними недоліками існуючих систем виявлення є переважання простого виявлення сигнатур, низька ефективність виявлення складних атак, розподілених у часі та просторі, а також недостатня інформація на рівні хоста та мережі для виявлення скоординованих атак та ізольованих атак.

Як практичний недолік можна відзначити те, що на звичайних персональних комп'ютерах неможливо обробити всю інформацію, що надходить від швидкості обробки та великої кількості обчислювальних операцій, щоб просто розділити релевантність однієї події на «іншу». Трафік мережі або інших подій часто в 1,5-2 рази повільніший за реальний час. Тому в деяких системах аналіз

виконується із затримкою. Це означає, що атака на захищені дані та обчислювальні ресурси не буде здійснена вчасно або навіть відображена за допомогою існуючих засобів захисту. Таким чином інструменти виявлення атак можна використовувати для фіксації всіх етапів атаки для кращого аналізу.

Більшість сучасних SBA спочатку не розроблені для роботи на різноманітних операційних системах і автономних апаратних і обчислювальних платформах. Тому більшість продуктів (як західних, так і вітчизняних) не можуть працювати на кількох операційних системах. Ці системи не отримують переваг від розробки коду та оптимізації для вибраних операційних систем і апаратних платформ, що є одним із їхніх головних недоліків.

Крім того, жодна програмна чи апаратна система не забезпечує режим «гарячої заміни», який дозволяє швидко розгорнути резервний набір у разі збою основного набору та відновити зруйновану лінію захисту периметра мережі. .

Однак у розвитку систем виявлення аномалій є позитивний момент - бажання розробників інтегрувати свої системи з наявними засобами захисту (інтернет-екрани, блокувальники каналів, QoS-менеджери).

На жаль, сучасний підхід до побудови систем виявлення вторгнень і виявлення ознак комп’ютерних атак на інформаційні системи повний недоліків і вразливостей, які дозволяють зловмисникам успішно долати межі захисту інформації.

На ринку існують десятки систем запобігання та виявлення вторгнень [13], але всі вони мають один суттєвий недолік: оскільки їхні операції базуються на розроблених правилах і шаблонах, не всі випадки вторгнень виявляються; крім того, вони не можуть "відповідати" шкідливому трафіку.

Крім того, слід зазначити, що продукція вітчизняних виробників на цьому ринку не представлена. Крім того, описані раніше методи завжди доступні як комерційна таємниця [14]. Тому нам потрібно розробити нову систему, яку можна використовувати в наборі мережевих інструментів.

Розглянутий матеріал дозволяє вивчити проблеми забезпечення безпеки мережі та вивчити сучасні методи аналізу мережевого трафіку.

2 ЗАВДАННЯ ПРОЕКТУВАННЯ СИСТЕМИ ТА МОДЕЛЬ МЕРЕЖЕВОГО ТРАФІКА

Система аналітики повинна охоплювати 100% трафіку та забезпечувати ефективні методи аналітики з навігацією за результатами.

Що стосується комплексного вирішення задачі аналізу мережевого трафіку, то її спочатку слід розділити на три незалежні підзадачі (рис. 2): перехоплення, зберігання та аналіз трафіку [8].

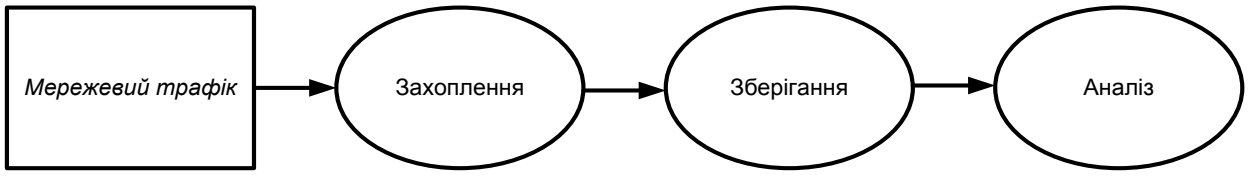

Рисунок 2.1 — Завдання системи аналізу мережного трафіку

2.1 Завдання перехоплення трафіку

Захоплення трафіку здійснюється за допомогою сніфферів. У загальному випадку, сніффер (англ. sniffer — дослівно перекладається як нюхач або винюхувач) — це програма або програмно-апаратний пристрій, призначений для перехоплення трафіку. В рамках конкретних продуктів можуть бути реалізовані додаткові можливості, наприклад розбір заголовків мережевих протоколів, фільтрація за заданими критеріями, відновлення сесій. Перехоплення мережевого трафіку може здійснюватися:

за допомогою «прослуховування» мережного інтерфейсу;

підключенням сніффера Sn до розриву каналу;

відгалуженням трафіку («дзеркалюванням») і копіюванням на сніффер (наприклад, Network tap);

аналізом бічних електромагнітних випромінювання;

атаками на канальному чи мережевому рівні, якіі змушують об'єкт нападу перенаправляти трафік на зловмисника.

Є два сніфера залежно від їх розташування:

— на маршрутизаторі (шлюзі);

— на кінцевому вузлі мережі.

В першому випадку перехоплюватиметься весь трафік, що проходить через інтерфейси пристрою, у другому — або лише трафік вузла мережі, якщо мережева карта працює в нормальному режимі, або пакети всіх пристроїв цього сегменту мережі (для цього у мережевої картки виставляється режим «promiscuous» нерозбірливий).

Створюються такі програми, ґрунтуючись на бібліотеці Pcap, що вільно розповсюджується (англ. «packet capture»). Вона призначена для використання спільно з мовами C/C++, а для роботи з бібліотекою іншими мовами, як-от Java, .NET, використовують обгортки. Для Unix-подібних систем це бібліотека libpcap, а Microsoft Windows — WinPcap.

Програмне забезпечення мережного моніторингу може використовувати libpcap або WinPcap, щоб захопити пакети у мережі, передачі пакетів у мережі. Крім того, підтримується збереження захоплених пакетів у файл і читання файлів, що містять збережені пакети.

Оскільки система, що розробляється, підключається до вихідного маршрутизатора, що зв'язує мережу провайдера і мережу Інтернет, відповідно, щоб не створювати затримки в роботі обладнання при такому обсязі оброблюваного трафіку, розумно використовувати метод «дзеркалування» трафіку, тобто дублювати його на інший мережевий інтерфейс (рисунок 2.2).

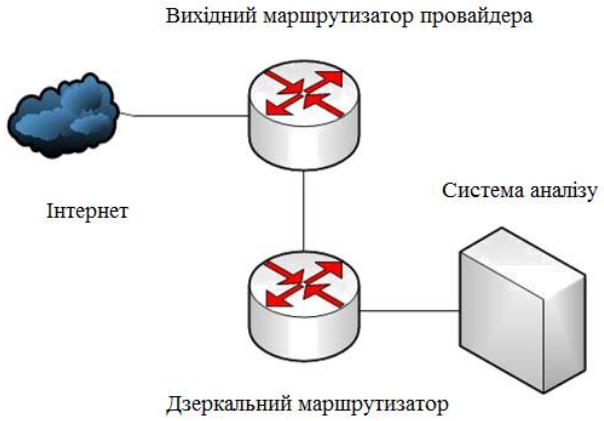

Рисунок 2.2 — Схема встановлення системи збирання інформації

2.2 Завдання аналізу трафіку

Що ж до завдання аналізу, то перевагу тому чи іншому інструменту надається з специфіки підзадач, які потрібно вирішити [9]. Більшість існуючих інструментів, зазвичай, проводять розбір заголовків мережевих протоколів, і навіть відновлюють сесії (базовий аналіз). У той же час існують досить специфічні завдання, для яких може не знайтися готового інструменту, наприклад:

аналіз протоколів тунелювання довільної глибини;

аналіз сеансів на прикладному рівні (виявлення зв'язків між потоками даних, що передаються по мережі);

вконання певних сценаріїв (скриптів), якщо в трафіку зустрічаються попередньо визначені підписи.

Існує два способи роботи аналізаторів:

у реальному часі;

з попередньо збереженим трафіком.

Аналіз у реальному часі вимагає, щоб інструмент працював безперервно з достатньою продуктивністю для аналізу вхідного трафіку. Він повинен мати можливість обробляти потенційно необмежений вхідний потік.

Коли виконується відкладений аналіз, інструмент отримує вхідні дані з файлу, що дозволяє більш детально проаналізувати трасування мережі порівняно з аналізом того самого трафіку в реальному часі.

Для оцінки продуктивності та тестування прототипу системи було взято мережевий трафік із шириною каналу 1,5 Гбіт/с. Якщо подивитися на статистику навантаження (рис. 2.3, табл. 2.1), то загальне середнє навантаження становить 591 Мбіт/с. Відповідно, обробити такий потік в режимі реального часу складно, ефективніше буде складувати сміття і вже з ним розбиратися. Звідси і необхідність зберігання трафіку.

.

Таблиця 2.1 — Статистика трафіку за добу

Канал	Максимальна завантаженість каналу, (Мбіт/с)	Середня завантаженість, (Мбіт/с)	Поточна завантаженість(Мбіт/с)
Вхідний	810.0	478.7	526.8
Вихідний	206.7	112.3	150.2

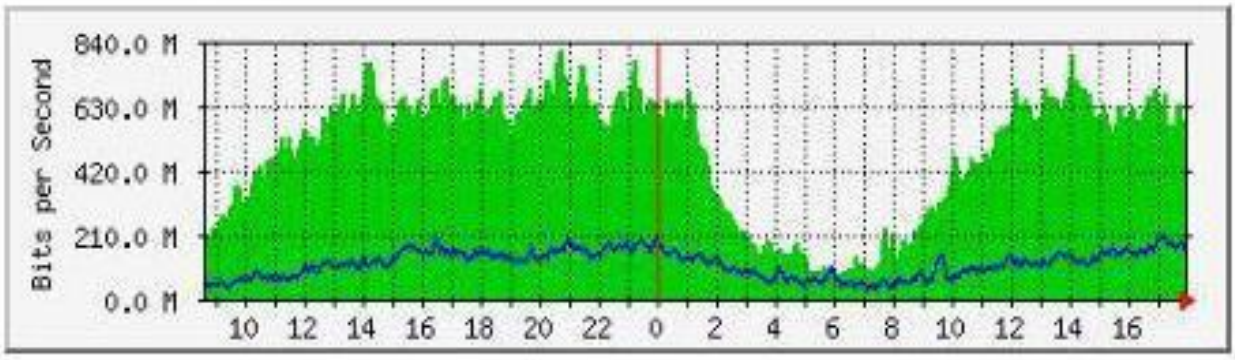

Рисунок 2.3 — Завантаженість зовнішнього Інтернет-каналу

2.3 Завдання зберігання трафіку

Надана частина трафіку займає 110 Гб на 28,190358 секунд реального часу передачі інформації; можна припустити, що з провайдера ці цифри буде збільшено у рази. З отриманих відомостей випливає необхідність обробки та перетворення дампа з метою зменшення займаного простору та оптимізації вибірки даних. Для цього добре підходять бази даних.

По-перше, це дає нам можливість швидко знаходити серед всього дампа відповідні умовам пакети, тому що кожен з них зберігається окремим записом у таблиці. Крім цього, для прискорення вибірок великої кількості даних застосовується індексування.

Індекс — це база даних, створена зі стовпців та індексів однієї чи кількох таблиць до відповідних рядків таблиці. Відповідно, прискорення роботи досягається в першу чергу за рахунок того, що структура покажчика оптимізована для пошуку.

Існує два типи індексів: кластерні та некластерні. За наявності кластерного індексу рядки таблиці впорядковуються за значенням ключа цього індексу.

Якщо таблиці немає кластерного індексу, таблиця називається купою, і індекс, створений такої таблиці, містить лише покажчики на записи. Це другий тип індексів. Кожна таблиця може мати лише один кластеризований індекс, але кожна таблиця може мати кілька різних некластеризованих індексів, кожен із яких визначає порядок власних записів.

Індекси можуть бути реалізовані в різних структурах, зазвичай використовуються дерева B*, дерева B+, дерева B і хеші.

Для оптимальної продуктивності запитів індекси зазвичай створюються для стовпців таблиці, які найчастіше використовуються в запитах. Тобто для однієї таблиці можна створити кілька індексів. Однак збільшення кількості індексів уповільнює операції додавання, оновлення та видалення рядків таблиці, оскільки самі індекси повинні оновлюватися.

Оскільки вони займають додаткову пам'ять, перед їх створенням необхідно переконатися, що запланований дохід від виконання запиту перевищить додаткові витрати комп'ютерних ресурсів на підтримку індексу.

По-друге, використання баз даних має забезпечувати зменшення розміру займаного простору, оскільки в продуманій архітектурі залишаться лише письмові дані.

Тому перед запуском системи слід дослідити: які бази даних найбільше підходять для цієї мети.

2.4 Вибір математичної моделі мережового трафіка

З розвитком сфери інформаційних технологій виникла потреба у організації передачі великого обсягу різних даних. У свою чергу це призвело до розвитку мережевих технологій. Однак з розширенням можливостей мереж передачі даних навантаження на мережі також зросла, внаслідок чого виникла необхідність розробки програмних та апаратних засобів для контролю та регулювання мережевої активності [26].

Однією з основних проблем для вирішення зазначеної задачі стало відсутність універсальної математичної моделі, здатної адекватно описати структуру мережевого трафіку. В основному це пов'язано зі збільшенням обсягів переданих даних, а також із застосуванням різних механізмів їх передачі (різні протоколи, способи маршрутизації і т. д.) [27]. Більшість математичних моделей, що застосовуються в даний час, може використовуватися для опису окремих видів переданих даних, а загальної теорії трафіку досі не існує.

В рамках теорії масового обслуговування пакети даних, що передаються розглядаються як випадкові події, не пов'язані один з одним.

Найбільш поширеною є класична модель Пуассона, в якій потік даних розглядається як незалежна випадкова величина, що експоненційно залежить від часу сеансу t:

$$f(t) = \frac{(\lambda t)^n}{n!} e^{-\lambda t},$$

де f(t) щільність розподілу;

λ — середня інтенсивність потоку за сеанс;

n 0,1, 2, ... — число потоків трафіку (подій) за період часу t.

Ця модель досить проста і може застосовуватися для аналізу трафіку, що містить незначний обсяг даних [28]. Однак в зв'язку з розширенням можливостей мереж передачі даних, а також з необхідністю передачі даних різних видів (аудіо, відео та інші) виявлено, що модель Пуассона не підходить для їх опису через низьку адекватність одержуваних результатів [29, 30].

Спроби модифікації розподілу Пуассона призвели до появи кількох математичних моделей, в рамках яких могли бути описані відхилення в структурі трафіку від стандартної моделі Пуассона, наприклад:

— ON/OFF-модель, згідно з якою трафік являє собою перерваний пуасонівський процес [31]. Відмінна риса — облік періодів неактивного стану

системи (коли пакети даних не передаються), що дозволяє досить добре описувати процес комутації пакетів;

— модель Пуассона, модульована марківським процесом, що дозволяє враховувати не тільки неактивний стан системи, а й кількість активних користувачів [32, 33]. У цьому випадку потік даних від одного користувача являє собою перервний пуассонівський процес, а результуючий потік розглядається як марківський процес.

Особливістю перелічених моделей і те, що пакети переданих даних розглядаються у яких як випадкові події, незалежні одна від одної, тобто розмір кожного наступного пакета даних залежить від розміру попереднього.

Теза про взаємну незалежність пакетів даних, що надходять, спростована в [34], де сформульовані основи теорії самоподібності. Відповідно до цієї теорії, під час роботи мережі має місце не тільки взаємозалежність як між окремими пакетами даних, а й між цілими фрагментами трафіку. В результаті було створено принципово новий клас моделей трафіку – самоподібні моделі.

Відмінна риса самоподібних процесів — наявність у них розподілів «важких хвостів» і залежності автокореляційної функції (АКФ), що повільно зменшується. Такі розподіли досить добре описуються моделями Парето та Вейбулла [35]. Для них характерна наявність степінної залежності функції щільності від швидкості передачі даних, проте вид функції для них дещо різниться:

— для моделі Парето $f(t) = \beta\alpha^{\beta}t^{-\beta-1}$;

— для розподілу Вейбулла $f(t) = \alpha\beta^{-\alpha}t^{\alpha-1}e^{-(t/\beta)\alpha}$,

де α, β — коефіцієнти рівнянь.

Однією з найпоширеніших математичних моделей для самоподібних процесів є класичний броунівський рух (БР), а також його аналог — модель фрактального броунівського руху (ФБР). У цьому випадку потік подій

розглядається як випадкова величина, яка залежить не тільки від часу, а й від величини попереднього потоку:

$$X_t = X_{t-1} + \varepsilon_t.$$

Тут X_t — значення випадкової величини в момент часу t; ε_t — білий шум.

Щільність розподілу приростів такої величини підпорядковується закону Гауса [36], для якого математичне очікування дорівнює нулю, а дисперсія:

$$\sigma^2(X_2 - X_1) = C(t_2 - t_1)^{2H},$$

Де X_1, X_2 — рначення випадкової величини в моменти часу t_1 і t_2;

C — деяка позитивна константа;

H – параметр Херста.

Параметр Херста дозволяє оцінювати міру самоподібності трафіку.

При $0{,}5 \leq H \leq 1{,}0$ процес є строго самоподібним і описується моделлю ФБР, при $H = 0{,}5$ — моделлю класичного БД, а при $0 \leq H < 0{,}5$ має стохастичний характер [37].

Ще одна група моделей, з використанням яких можливе моделювання самоподібних процесів, це так звані моделі часових рядів. Найбільш поширеними з них є моделі авторегресії та ковзного середнього.

Модель авторегресії AR передбачає, що поточне значення функції визначається лінійною залежністю від кількох попередніх значень цієї функції, а похибка — як білий шум:

$$X_t = C + \sum_{i=1}^{p} \alpha_i X_{t-1} + \varepsilon_t,$$

де p — порядок моделі AR;

D_i — коефіцієнти рівняння авторегресії;

X_{t-1} — лаговий оператор.

В свою чергу, модель ковзного середнього MA передбачає, що значення функції в даний час коливається біля деякого середнього значення, а величина відхилення залежить від значення функції в попередні періоди часу:

$$X_t = C + \varepsilon_t + \sum_{i=1}^{q} \beta_i X_{t-1},$$

де q — порядок моделі MA;

β_i — коефіцієнти рівняння ковзного середнього.

На базі моделей AR та MA сформовано два види моделей:

1) Модель ARMA, у якій попередні значення функції впливають як на поточне значення функції, а й у його відхилення:

$$X_t = C + \sum_{i=1}^{p} \alpha_i X_{t-i} + \varepsilon_t + \sum_{i=1}^{q} \beta_i \varepsilon_{t-i},$$

де ε_{t-I} — лаговий оператор помилки;

2) Модель ARIMA (модель Бокса — Дженкінса, або модель інтегрованої авторегресії — ковзного середнього), яка дозволяє працювати з залежностями, що мають тренд:

$$X_t = C + \sum_{i=1}^{p} \alpha_i \Delta^d X_{t-i} + \varepsilon_t + \sum_{i=1}^{q} \beta_i \Delta^d \varepsilon_{t-i},$$

де d — порядок моделі ARIMA, що характеризує ступінь інтегрування;

Δ^d — оператор взяття кінцевої різниці порядку d.

Незважаючи на те, що моделі часових рядів отримали досить широке поширення при вивченні трафіку, завдання отримання математичного опису

трафіку все ще залишається досить складном. У тому числі це пов'язано з тим, що:

— мережевий трафік, як правило, поєднує в собі різні види даних, які описуються різними моделями;

— залежно від типу переданих даних ступінь самоподібності трафіку може змінюватись.

У зв'язку з цим актуальним завданням є аналіз мережного трафіку, і навіть підбір математичної моделі, здатної адекватно описувати різні види трафіку та його поєднання.

Для виконання дослідження використовувалися дані, отримані в мережі Інтернет з різних видів трафіку:

1) потокове відео (протокол UDP);

2) YouTube-трафік (протокол QUIC);

3) трафік змішаного типу, що включає в себе одночасну передачу даних YouTube та потокового відео;

4) torrent-трафік (протоколи GVSP, UDP, BitTorrent);

5) комбінований трафік, що характеризується поєднанням різних видів трафіку.

Збір даних здійснювався як із окремих вузлів так, і зі всієї мережі загалом.

Мережевий трафік, що надходить з центральних комутаторів, дублювався на окремий порт, звідки здійснювалася передача даних на комп'ютер з працюючим аналізатором трафіку WireShark. Далі проводилося угруповання отриманої інформації або за видами трафіку, або за адресами окремих ПК. Отримані таким чином дані оброблялися в пакеті MATLAB.

Процедура обробки полягала в наступному. Для отримання даних, більш однорідних за часом, проводилася процедура їх групи шляхом обчислення середнього значення обсягу пакетів (байт), що передаються за певний період часу, з: 0,1; 0,5; 1,0; 5.0. Далі на основі згрупованих даних розраховувалися відносні частоти розподілу швидкості передачі даних (Байт/с). Потім здійснювалося порівняння отриманих функцій щільності розподілів зі стандартними моделями.

Як моделі порівняння обрані: класична модель Пуассона, моделі логнормального та експоненційного розподілу, а також моделі Парето і Вейбулла, що характеризують розподіли з «важким хвостом».

Перевірка на самоподібність трафіку включала розрахунок АКФ, визначення типу залежності між членами ряду даних (повільно або швидко спадаюча залежність), а також знаходження параметра Херста за R/S-методом [37].

Автокореляційна функція визначалася шляхом знаходження коефіцієнта кореляції між вихідним рядом даних і рядом, в якому всі елементи зміщені на один крок. Потім методом найменших квадратів знаходилися коефіцієнти для експоненційної та статечної залежності, до яких АКФ найбільш близька.

Якщо АКФ може бути описана експоненційною функцією, то має місце залежність, що швидко зменшується. Якщо функція носить статечний характер, то йдеться про повільно спадаючу залежність, яка характерна для самоподібних процесів. Оцінка типу залежності проводилася за допомогою показника LRD, який розраховувався як відношення дисперсій різниць фактичного значення АКФ та розрахункового, отриманого для статечної та експоненційної моделей.

Завдання підбору моделі для опису мережного трафіку полягало в порівнянні емпірично отриманих даних з моделями часових рядів. Як такі моделі обрані моделі БД, ФБД, а також AR, MA, ARMA та ARIMA. Для кожної моделі підібрано параметри рівнянь, на основі яких далі отримано розрахункові значення швидкостей передачі даних. Порівняння фактичних і розрахункових значень швидкості здійснювалося за допомогою коефіцієнта детермінації. Процедура повторювалася для різних порядків моделей (p і q), після чого розглядалася залежність коефіцієнта детермінації від відповідного порядку моделі.

Були проаналізовані дані, зібрані для різних видів трафіку — мультимедійного, пірингового та потокового (відео-конференції). Дані для аналізу трафіку отримані як окремих пристроїв, так мережі в цілому.

Зважаючи на нерівномірність появи запитів у мережі за часом проводилося угруповання отриманих даних з різними часовими інтервалами, з: 0,1; 0,5; 1,0; 5.0. Для згрупованих даних розраховувалася середня швидкість передачі. Далі виконувалася оцінка густини розподілу отриманих значень швидкості (рис. 2.4).

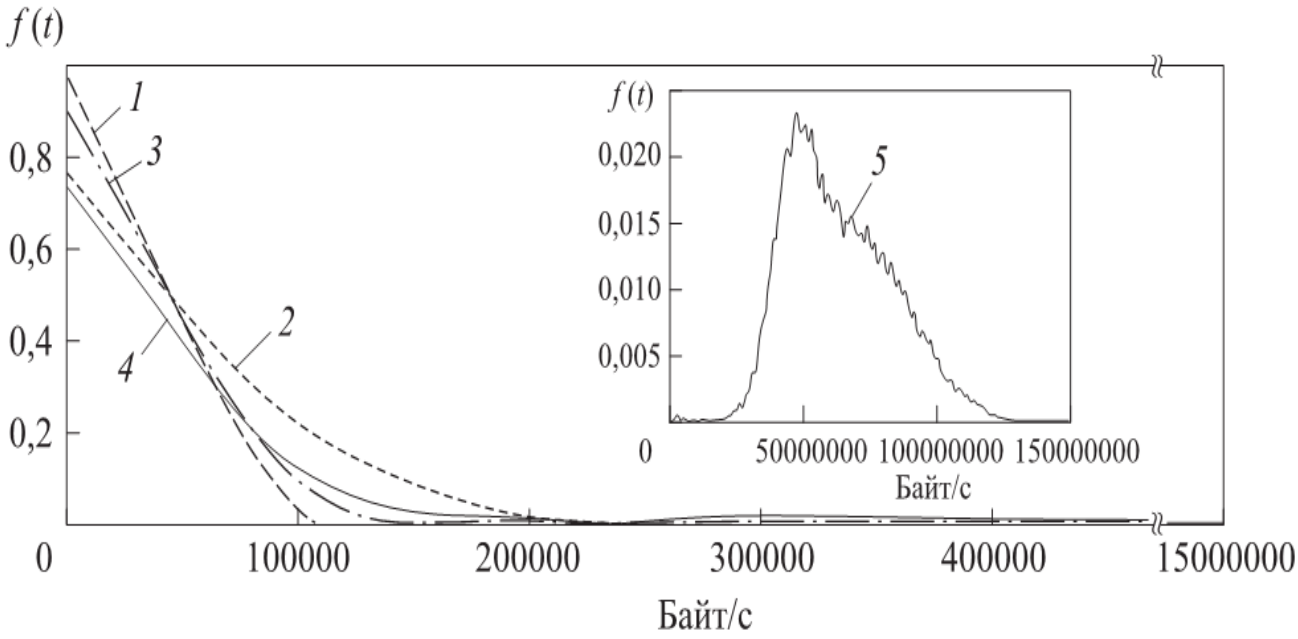

Рисунок 2.4 — Щільність розподілу швидкостей передачі даних (інтервал 0,1 с): для YouTube-трафіку (1), потокового відео (2), трафіку змішаного типу (потокове відео + YouTube-трафік) (3), комбінованого (4) та torrent-трафіку (5)

Вид отриманих розподілів дозволяє припустити, що вони можуть мати «важкий хвіст».

Далі на підставі експериментальних даних визначалися параметри досліджуваних моделей для отримання рівняння функції щільності розподілу. Для кожної моделі розраховувалися коефіцієнти детермінації. Отримані результати наведено у табл. 2.1.

Таблиця 2.1 — Порівняння математичних моделей для різних видів мережевого трафіку (часовий інтервал 0,1 с)

Вид переданих даних	Коефіцієнт детермінації R2 для розподілу					LRD	H
	Пуассона	Експоненціального	Логнормального	Парето	Вейбулла		
Потокове відео (UDP)	0,880	0,271	0,985	0,732	0,553	0,530	0,752
YouTube(QUIC)	-	0,212	0,974	0,908	0,825	0,956	0,289
Змішаний (Потокове відео + YouTube)	-	0,258	0,9401	0,908	0,917	0,8997	0,458
Torrent (GVSP, UDP, BitTorrent)	0,012	-	0,957	0,0070	0,038	0,553	0,805
Комбінований	0,857	0,489	0,980	0,958	0,920	2,173	0,825

Для перевірки трафіку на самоподібність розраховувалася АКФ, яка відображає залежність поточних значень швидкості від її попередніх (рис. 2). При цьому залежність між пакетами даних, що передаються, спостерігається для всіх вивчених видів трафіку. Однак для YouTube-трафіку та змішаного трафіку ця залежність виражена слабше.

Наступним кроком стала оцінка типу залежності між членами ряду даних. Для цього розраховувався показник LRD, значення якого має бути менше 1 для монотонно спадаючих залежностей та більше 1 для швидко спадаючих залежностей.

Розраховане щодо різних видів трафіку значення показника LRD (див. табл.2.1) свідчить у тому, що з більшості типів даних АКФ характеризується наявністю монотонно спадаючої залежності. Для комбінованого трафіку

спостерігається коливальний характер АКФ (період коливань становить 8 с), що дещо знижує точність розрахунку.

Перевірка самоподібності трафіку також оцінювалася за допомогою параметра Херста H [37]. Результати розрахунку параметра Херста R/S методом наведені в табл. 2.1.

Далі виконувалося порівняння експериментальних даних із моделями часових рядів: БД, ФБД, AR, MA, ARMA, ARIMA. Для кожної моделі підібрано відповідні порядки (p і q), за яких вона показувала найкращу надійність.

При порівнянні отриманих функцій щільності для різних видів трафіку з класичною моделлю Пуассона (див. табл. 2.1) виявлено, що адекватність отриманої моделі знижується зі збільшенням часу роботи мережі. Це пов'язано з тим, що вид функції густини розподілу для моделі Пуассона істотно залежить від таких параметрів, як час сеансу зв'язку, інтервал між пакетами, а також від типу даних, що передаються.

Зважаючи на те, що розподіл Пуассона є практично симетричним щодо середнього значення швидкості процесу, що позначається як λ, а значення параметра, що розраховується, змінюються в межах ±λ, з його допомогою неможливо описати угруповання даних в області низьких значень швидкості, а також наявність «довгого хвоста». Це не дозволяє використовувати модель Пуассона для моделювання мережевого трафіка системи.

Моделі розподілу з «важкими хвостами» показали досить високу адекватність математичного опису лише при малому часовому інтервалі (0,1 с), а при великих часових інтервалах (1…5 с) такі моделі виявилися непридатні. Це може бути обумовлено тим, що ці моделі є монотонно спадаючими, а експериментально отримані дані характеризуються наявністю «збурень», і чим більше часовий інтервал, тим сильніше виражені ці «збурення». Аналогічний висновок може бути зроблений для експоненційного розподілу.

Коефіцієнт детермінації, розрахований для функції щільності логнормального розподілу (див. табл. 2.1), свідчить про досить високий рівень

достовірності отриманого математичного опису, а також дозволяє застосовувати дану модель для опису різних видів мережного трафіку.

Аналізуючи вид АКФ, отриманих для різних видів трафіку (див. рис. 2.4), можна відзначити, що в YouTube і змішаний трафіки характеризуються короткочасною залежністю: значення АКФ для них наближається до нуля менш ніж після 15 кроків часової затримки. При цьому для інших видів трафіку (потокове відео, torrent-трафік, комбінований) залежність носить більш довготривалий характер.

При порівнянні АКФ потокового відео, YouTube і змішаного трафіків можна відзначити, що значення АКФ змішаного трафіку дещо вище, ніж YouTube-трафіку, але менше, ніж у потокового відео.

Особливість АКФ комбінованого трафіку — наявність вираженої циклічної структури з періодом коливань 8 с. Зниження інтенсивності коливань зі збільшенням часового інтервалу пов'язані з усередненням значень швидкості у процесі процедури агрегування.

Дані результати дозволяють зробити висновок про наявність автокореляції для всіх типів переданих даних, а також про її залежність від співвідношення видів трафіку, що входять до складу загального потоку даних.

Результати, отримані при оцінці показників LRD і H (див. табл. 2.1), свідчать про те, що для потокового відео, torrent і комбінованого трафіку характерна наявність фрактальності між членами ряду, а для YouTube-трафіку ступінь самоподібності досить низька. Це підтверджує зроблені раніше висновки про його несамоподібність. Цікаво, що значення параметра Херста для змішаного трафіку знаходиться в інтервалі значень між результатами відео та YouTube-трафіку. На підставі цього можна припустити таке: на ступінь самоподібності впливає не тільки тип переданих даних, але і їх співвідношення в загальному обсязі даних.

Під час вивчення моделей часових рядів виявлено таке. Для моделі AR загалом точність зростає зі збільшенням числа елементів ряду. Модель MA також

характеризується збільшенням збіжності результатів при збільшенні числа кроків, проте коефіцієнт детермінації в такому випадку не перевищує 0,0055.

Порівняння експериментальних даних з розрахованими за моделлю ARMA показало, що збільшення кроку за будь-яким з параметрів (p, q) призводить до зниження точності.

Інтегрування вихідного ряду даних та розрахунок моделі ARIMA призвели до незначного зниження збіжності результатів. Це можна пояснити відсутністю будь-якого тренду (зростаючого чи спадаючого) у вихідному часовому ряді і, як наслідок, зниженням надійності результатів подібного усереднення.

Для порівняння різних моделей прогнозування трафіку обрані параметри рівнянь, у яких дана модель показала найкращу збіжність результатів (табл. 2.2).

Таблиця 2.2 — Порівняння моделей прогнозування трафіку (часовий інтервал 0,1 с)

Модель	p	d	q	R2
AR	90	0	0	0,8144
MA	0	0	90	0,00533
ARMA	6	0	1	0,7973
ARIMA	10	1	1	0,7972
БД	1	0	0	0,7895
ФБД	1	0,8251	0	0,8067

Найкраща збіжність результатів спостерігається у моделі AR. Однак для використання такої моделі в режимі реального часу потрібні великі обчислювальні потужності, пов'язані зі складністю рівняння моделі. У свою чергу модель ФБД характеризується меншими витратами обчислювальної потужності при незначному зниженні точності (менше 1%). Це робить її більш привабливою не тільки для моделювання трафіку, але і для впровадження різні системи моніторингу і контролю. Таким чином, шляхом оцінки отриманих результатів можна зробити висновок про те, що для прогнозування мережного трафіку найбільш підходящою є модель ФБД.

Проведені дослідження показали, що вибір математичної моделі мережевого трафіку залежить від виду даних, що передаються, які присутні в його складі, а також від їх співвідношення. У більшості випадків розподіл даних підпорядковується логнормальному закону.

Для всіх досліджених видів трафіку розподіл даних має «важкий хвіст» і характеризується залежно, що повільно спадає. Це, з урахуванням розрахованих значень параметра Херста, свідчить про самоподібності вивчених видів трафіку (крім YouTube), а також про залежність ступеня самоподібності від співвідношення даних різних видів.

Отримані результати можна використовувати для математичного моделювання мережевого трафіку різних видів, у тому числі комбінованого, а також для розробки систем його моніторигу та контролю.

3 РОЗРОБКА ПРОТОТИПУ СИСТЕМИ МОНІТОРИНГУ ТА КОНТРОЛЮ МЕРЕЖНОГО ТРАФІКУ

Спроектована архітектура системи моніторингу та контролю, її розгорнуту схему можна побачити рисунку 3.1.

Вихідний трафік маршрутизатора копіюється в систему у вигляді файлів фіксованого розміру, які потім зчитуються та декодуються програмою аналізатора. Ми збираємо та записуємо в нашу базу даних нищенаведену інформацію від клієнтів:

— довжину пакета;

— час отримання;

— про протоколи мережного та транспортного рівня;

— ip-адреси джерела та одержувача;

— прапори SYN, ACK, FIN протоколу TCP;

— значення портів протоколу UDP.

Отримуючи запит від користувача, web-сервер завантажує з бази необхідні дані, аналізує їх у запитаному вигляді та відображає.

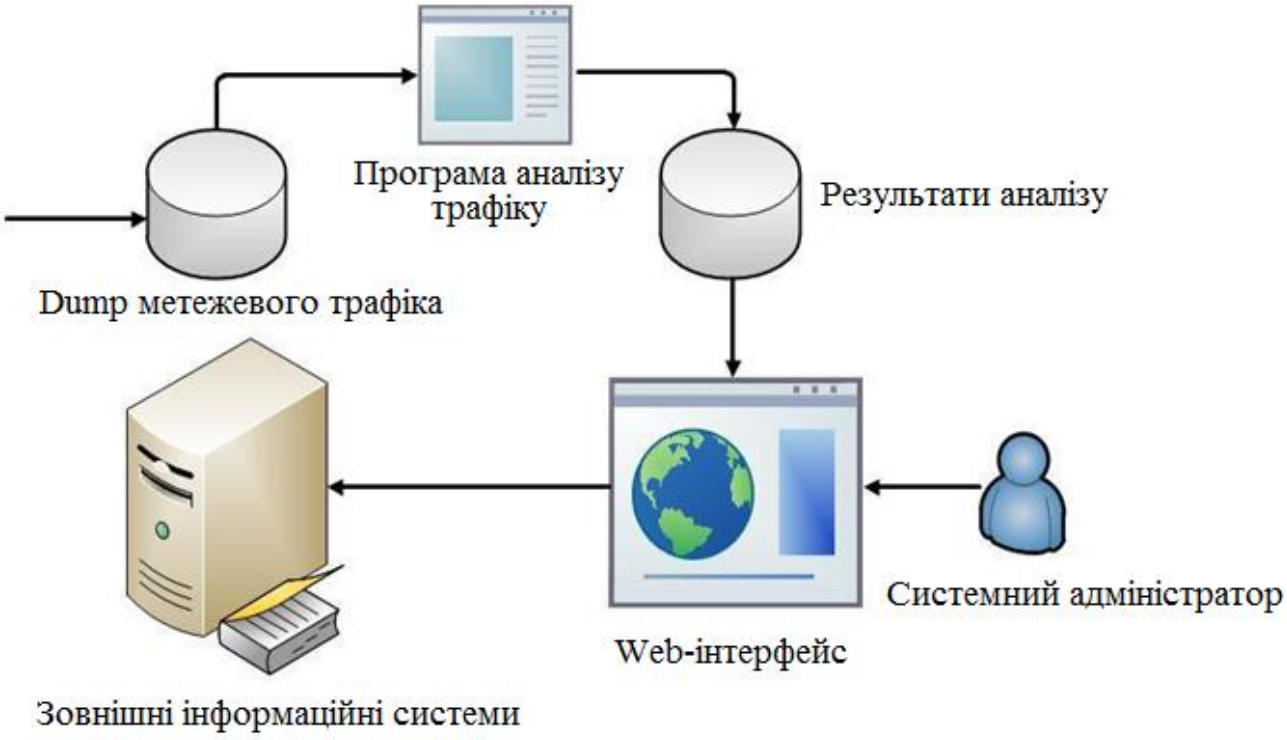

Рисунок 3.1 — Структурна схема системи аналізу мережного трафіку

3.1Модуль перетворення дампа мережевого трафіку

Цей модуль призначений для читання та перетворення на інший формат збережених даних. Він постійно моніторить стан файлової системи сервера: щойно з'являється новий файл дампа, відразу надсилаємо його на обробку.

На даний момент йде розбір пакетів виключно на мережному та транспортному рівні, тому що для прототипу це той максимум інформації, який потрібен у подальшому аналізі [25]. Функціонал системи можна легко розширити в міру виникнення потреби.

З мережевого рівня зберігаємо інформацію про:

— довжину пакета;

— час надходження пакета (секунди, мілісекунди, рядок з датою та часом);

— протокол мережевого рівня (ідентифікатор та назва);

— протокол транспортного рівня (ідентифікатор та назва).

— транспорт З рівня транспорту:

— IP-адреса та вихідний порт;

— IP-адреса та порт одержувача.

Цей модуль реалізовано на C++ за допомогою бібліотеки WinPcap для належного керування трафіком і драйвера бази даних.

3.1.1 Обґрунтування вибору програмних засобів

C++ — це структурована статично типізована мова програмування загального призначення. Будучи дуже потужною мовою, він містить інструменти для створення ефективних програм практично для будь-яких цілей: від низькорівневих утиліт і драйверів до складних програмних пакетів.

— конкретний C++:

— підтримує різні технології програмування: стандартне директивне програмування, метапрограмування (шаблони, макроси), ООП, узагальнене

програмування та операторні функції дозволяють користувачам писати короткі та ефективні вирази для типів користувачів у природній формі алгебри;

— Objects автоматично викликає деструктори об’єктів, коли їх знищує зворотні виклики конструктора, спрощуючи генерацію коду, роблячи вільні ресурси (пам’ять, семафори, файли тощо) безпечними та дозволяючи гарантоване виконання будь-якого переходу стану програми з обов’язковим звільненням ресурсу. , запис у журналі);

— мова попередньо виконує програми, важливі для побудови системного коду в реальному часі, створеного компілятором для використання можливостей мови, описаних у стандарті (наприклад, коли ви перетворюєте змінну в інший тип), який також визначає розташування код. виконана така дія дозволяє виміряти час реакції програми на зовнішню подію;

— Log підтримує як логічну (змінну), так і фізичну (постійну) концепції послідовності, що покращує надійність програми, оскільки дозволяє програмісту виявляти помилкові спроби змінити значення змінної, тоді як оператор послідовності надає додаткове розуміння правильного використання класів і функції програмістом, який читає текст програми. , що, у свою чергу, може бути сигналом для оптимізації та дозволяє регулярно виявляти внутрішньо перевантаження функцій належності; заперечення проти об’єкта виклику методу (зміна константи на читання зміни константи);

— може створювати об'єктно-орієнтовані мови програмування на основі Object. Щоб проілюструвати цю тезу, ви можете звернутися до бібліотеки Boost.Spirit, яка дозволяє парсерам визначати граматики EBNF у коді C++;

— він використовує шаблони для створення узагальнених контейнерів і алгоритмів для різних типів даних, а також для спеціалізації та обчислень на етапі агрегації;

— emp може імітувати класи домішок і комбінаторну параметризацію бібліотек за допомогою шаблонів і множинного успадкування, прикладом такого використання мови є бібліотека Loki, де клас SmartPrt може контролювати та

створювати лише кілька параметрів комп’ютера. близько 300 видів розумних індикаторів для управління ресурсами;

— Direct може імітувати мовні розширення для підтримки парадигм, які підтримуються компіляторами: наприклад, бібліотека Boost.Bind дозволяє зв’язувати аргументи функції;

— Platform-to-platform: стандарт мови, який встановлює мінімальні вимоги до обчислювального пристрою для запуску скомпільованих програм, стандартна бібліотека має можливість визначати фактичні характеристики системи виконання., слід зазначити, що компілятори для цієї мови доступні для багатьох платформ;

— C++: дуже сумісний з мовою C; дозволяє використовувати існуючий код C: код C може бути скомпільований компілятором C++ з мінімальними змінами; Бібліотеки, написані на C, можна викликати з C++ без додаткових витрат, у тому числі на рівні функцій зворотного виклику, які дозволяють бібліотекам, написаним на C, викликати код, написаний на C++;

— вона ефективна у використанні, тому що розроблена таким чином, щоб забезпечити максимальний контроль над усіма аспектами структури та компонування програми;

— для забезпечення максимальної продуктивності вона дозволяє вимкнути всі мовні параметри, які викликають додаткові витрати, оскільки їх використання не є обов’язковим;

— надає можливість використання низькорівневі інструменти для роботи з пам'яттю та адресами.

Незважаючи на те, що C++ має свої недоліки, вони не настільки важливі в нашому випадку, оскільки швидкість використання цієї мови вище, ніж незручності, які виникають під час розробки.

3.2 Модуль зберігання мережного трафіку

Через драйвер програма-сніффер заповнює базу даних записами про отримані пакети (один запис — один пакет). Призначення даного модуля повз

зберігання інформації — це зменшити обсяг займаний дампом і прискорити вибірку даних при запиті користувача.

Для нашої системи сніфер записує лише певні частини пакетів, які на поточному етапі цікаві нам для аналізу. У таблиці 3.1 відображено схему запису про один пакет.

Таблиця 3.1 — Схема даних, що зберігаються в базі даних

	Поле	Тип	Опис
id		ObjectId	Ідентифікатор запису у документі
time		Object	Збірне поле для часу отримання пакету
	seconds	Int32	Час у секундах
	usecond	Int32	Час у мілісекундах
	data	String	Дата та час отримання пакету
internetLayer		Object	Дані про протокол мережного рівня
	numbe	Int32	Ідентифікатор
	name	String	Назва
transportLayer		Object	Дані про протокол транспортного рівня
	numbe	Int32	Ідентифікатор
	name	String	Назва
length		Int32	Довжина пакета в октетах, включаючи заголовок та дані
source		Object	Дані про відправника пакету
	ip	String	IP-адреса
	port	Int32	Порт
destination		Object	Дані про одержувача пакету
	ip	String	IP-адреса
	port	Int32	Порт

3.2.1 Обґрунтування вибору програмних засобів

За прототип було прийнято MongoDB — документоорієнтовану базу даних з відкритим вихідним кодом, яка не потребує інтерпретації схем таблиць.

Ключові особливості:

— документоорієнтоване сховище (схема даних, наприклад JSON);

— використання JavaScript як мови для виконання запитів Java;

— широкий спектр атомарних операцій над даними (умовний пошук, складна вставка/оновлення тощо);

— різні типи даних (особливо підтримка масивів);

— підтримка індексів (B-дерево);

— запити профілю;

— відстеження операцій, що змінюють дані в базі даних;

— підтримка відмовостійкості та масштабованості: асинхронна реплікація, кластеризація екземплярів і розподіл баз даних між вузлами;

— ефективне обслуговування великих об'єктів, адміністративний інтерфейс, серверні функції, Map/reduce та інші;

— повнотекстовий пошук із підтримкою морфології. На відміну від реляційних баз даних, MongoDB пропонує орієнтовану на документ модель даних, яка робить документи швидшими, масштабованими та простішими у використанні.

Але враховуючи всі недоліки традиційних баз даних і переваги MongoDB, важливо розуміти, що проблеми різні і способи їх вирішення різні. У деяких випадках, якщо вам потрібно підтримувати складні структури даних, MongoDB дійсно може покращити продуктивність вашої програми. В іншому випадку було б краще використовувати власні бази даних посилань. Крім того, ви можете використовувати гібридний підхід: зберігати один тип даних у MongoDB, а інший – у традиційних базах даних.

Система MongoDB може представляти кілька баз даних, розташованих на одному фізичному сервері. Функціональність MongoDB дозволяє розміщувати

кілька баз даних на кількох фізичних серверах, і ці бази даних можуть легко обмінюватися даними та підтримувати цілісність.

Однією з важливих властивостей є формат даних у MongoDB. Перш за все, слід визначитися з популярним стандартом обміну та зберігання даних – JSON (JavaScript Object Notation). JSON ефективно описує складні структури даних. Хоча сховище даних MongoDB офіційно не використовує JSON, у цьому відношенні воно схоже на JSON. MongoDB використовує для зберігання формат під назвою BSON або бінарний JSON.

BSON дозволяє працювати з даними швидше: швидше пошук і обробка. Слід зазначити, що BSON має невеликий недолік порівняно з форматом JSON: загалом дані у форматі JSON займають менше місця, ніж формат BSON, але, з іншого боку, цей недолік з лишком компенсується швидкістю.

Кросплатформенність MongoDB досягається тим фактом, що MongoDB написана мовою C++, тому її легко переносити на різні платформи. MongoDB можна розгорнути на платформах Windows, Linux, MacOS і Solaris. Ви також можете завантажити вихідний код і зібрати MongoDB, але рекомендується використовувати бібліотеки з офіційного сайту.

Тоді як стандартний світ SQL складається з таблиць, світ MongoDB складається з колекцій. Однак у реляційних базах даних, хоча таблиці містять об’єкти з однаковою жорсткою структурою, колекція може містити різні об’єкти з різними структурами та іншими властивостями.

Система зберігання даних MongoDB представляє велику кількість екземплярів. Цей набір містить первинний вузол і може містити набір вторинних вузлів. Усі вторинні вузли зберігають цілісність і автоматично оновлюються, коли оновлюється основний вузол. Якщо основний вузол виходить з ладу з будь-якої причини, один із вторинних вузлів стає основним вузлом.

Відсутність схеми бази даних і незначна зміна концепції в цьому відношенні, необхідність зберігання даних, змінені схеми полегшують роботу з базами даних MongoDB і пізнішими масштабами. Крім того, економиться час

розробників. Вам більше не потрібно мати справу з перетворенням баз даних і витрачати час на створення складних запитів.

Однією з проблем під час роботи з будь-якою системою баз даних є зберігання великих обсягів даних. Ви можете зберігати дані у файлах за допомогою різних мов програмування. Деякі СУБД надають спеціальні типи даних для зберігання двійкових даних БД (наприклад, BLOB в MySQL).

На відміну від реляційної СУБД, MongoDB дозволяє зберігати різні типи документів, але розмір документа обмежений 16 Мб. І MongoDB пропонує рішення – спеціальну технологію GridFS, яка дозволяє зберігати дані розміром понад 16 Мб.

Система GridFS складається з двох колекцій. Перша колекція, яка називається файлами, зберігає такі метадані, як імена та розміри файлів. В іншій колекції, яка називається сегментами, дані файлу зберігаються в невеликих сегментах, зазвичай сегменти розміром 256 Кб.

3.3 Модуль аналізу та відображення

До заповненої бази підключається web-сервер, який перебирає на себе функціонал:

— розмежування доступу до інформації — взаємодія дозволена лише провайдеру;

— проведення заданої аналітики;

— відображення даних та графіків;

— забезпечення запитів на сторонні ресурси — отримання даних про автономні системи, яким належать IP-адреси, на які йдуть запити.

3.3.1 Аналітика мережного трафіку

3.3.1.1 Аналіз часових інтервалів між пакетами

Із заданого часового проміжку вибираються пакети, які відправляються з локальної мережі у зовнішню, вимірюється часовий проміжок між сусідніми записами пакетами, а потім підраховується кількість пакетів, що потрапили в

заданий часовий інтервал. Тобто, наприклад, якщо пакет прийшов з інтервалом 10 мілісекунд від попереднього, він записується в проміжок від 0 до 10 мілісекунд.

Для демонстрації роботи було встановлено проміжок 200 мілісекунд, це означає, що підраховувалися пакети, які потрапляють в інтервал кроком 200 мілісекунд. Також було встановлено обмеження виведення отриманих результатів: відображаються перші 50 мілісекунд (рисунок 3.2).

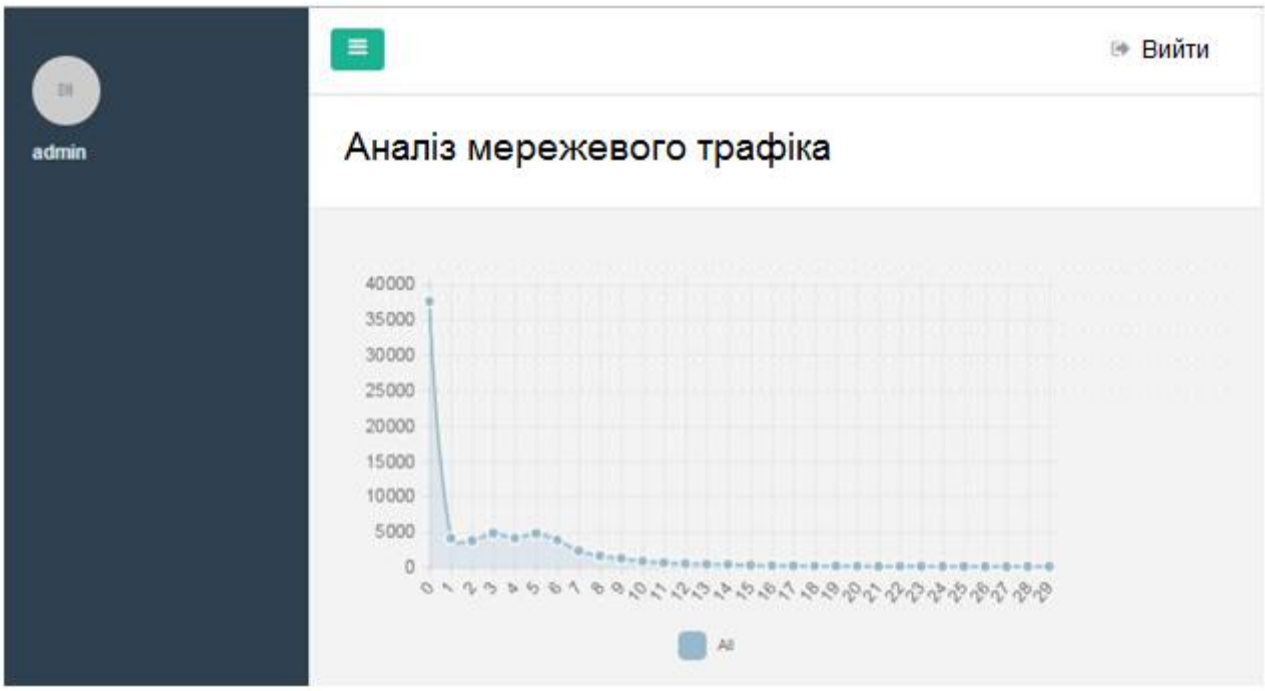

Рисунок 3.2 — Відображення графіка часових інтервалів між відправленими пакетами з кроком 200 мілісекунд

Крім цього, має фільтр, що дозволяє отримувати тимчасову статистику для всіх типів мережевих пакетів, так і для окремих протоколів.

Наприклад, на рисунках 3.3, 3.4 виводяться графіки часових інтервалів для пакетів, що встановлюють сесію протоколу TCP, для пакетів протоколу UDP.

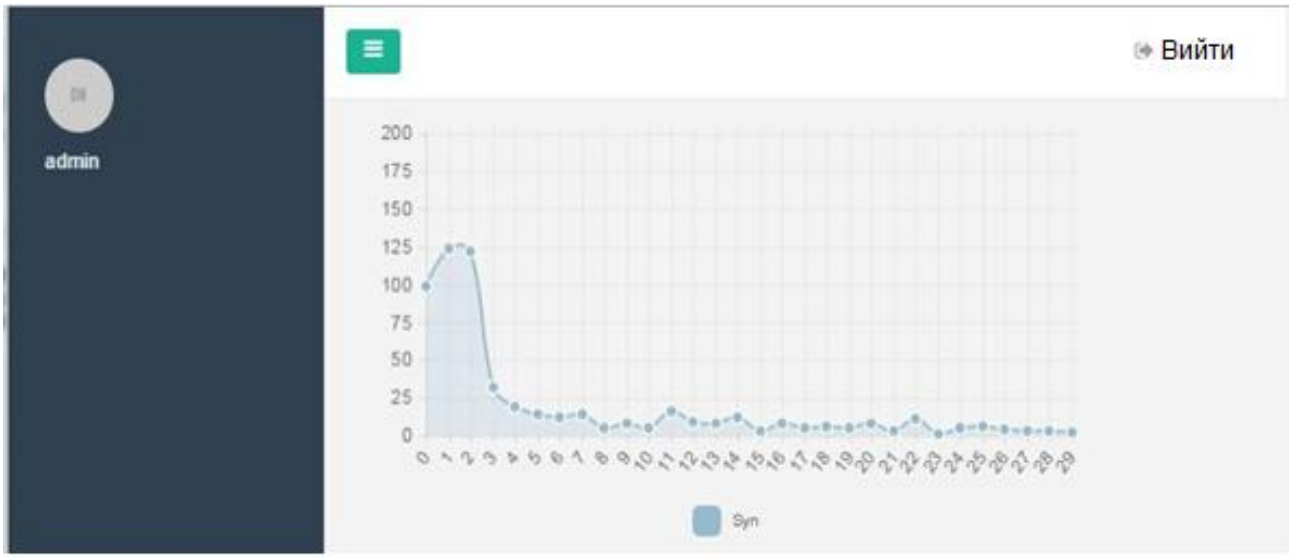

Рисунок 3.3 — Графік тимчасових інтервалів між відправленими пакетами із встановленням сесії

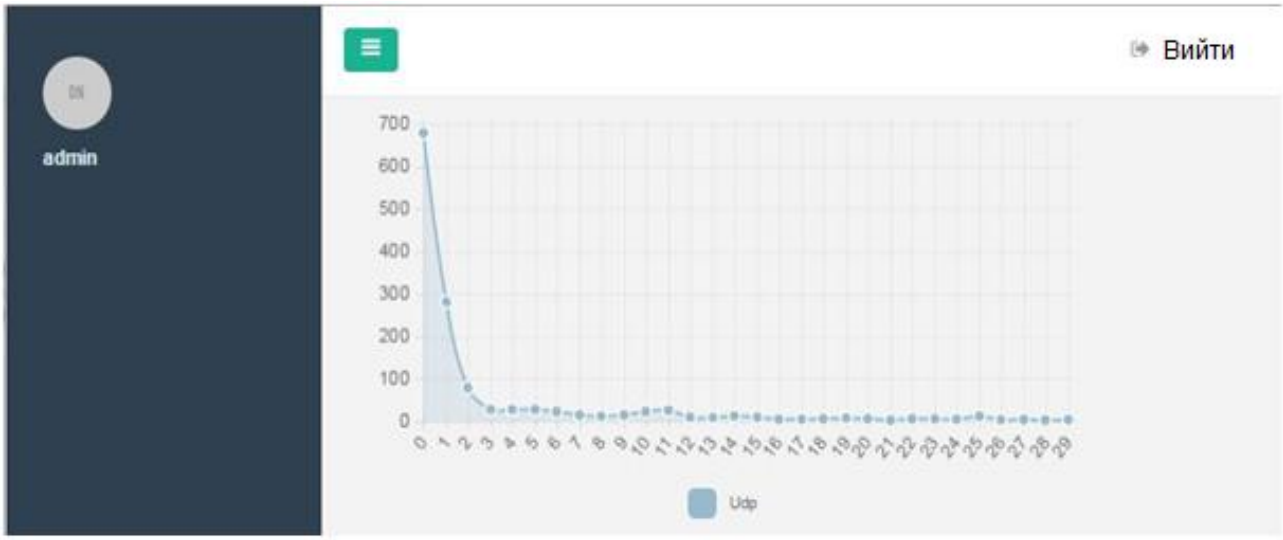

Рисунок 3.4 — Графік тимчасових інтервалів між відправленими пакетами протоколу UDP

Рисунок 3.5 — Графік тимчасових інтервалів для порівняння аналітики з різних протоколів

3.3.1.2 Аналіз частоти запитів за IP-адресами

Наступною частиною аналітики є виведення статистики частоти запитів різні IP-адреси (рисунок 3.6). Для демонстрації вивід був обмежений до 30 ресурсів, що часто запитуються.

Таким чином, можна визначати популярні ресурси, і у разі появи нової адреси у списку швидко перевірити доступну інформацію про його автономну систему.

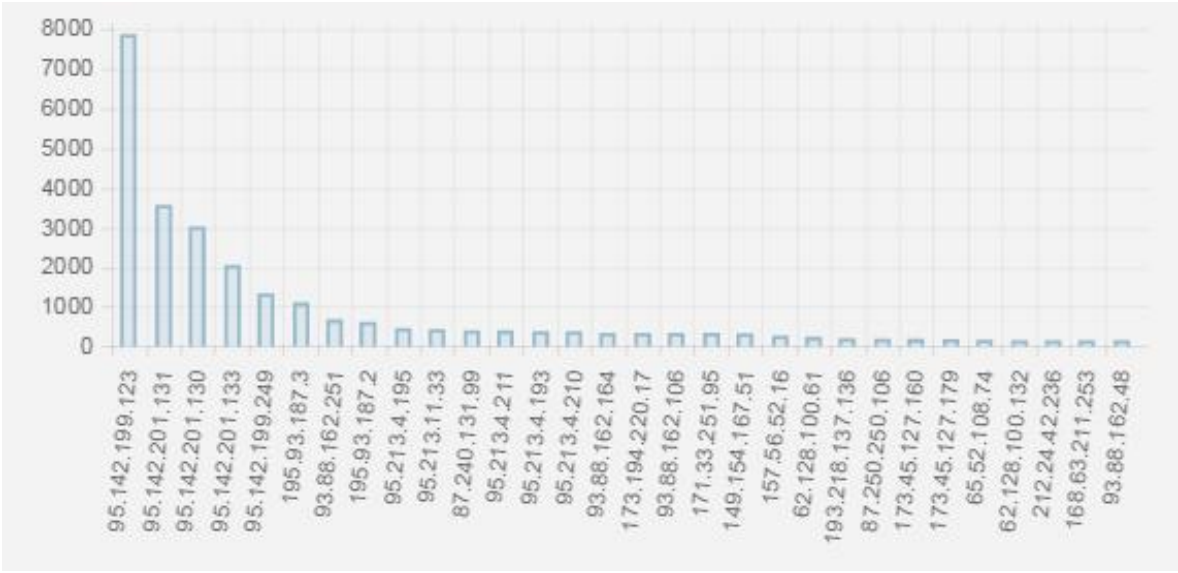

Рисунок 3.6 — Графік частоти запитів на IP-адреси

3.3.2 Обґрунтування вибору програмних засобів

Сервер реалізований на технологічному комплексі платформи MEAN (рис. 3.7):

— MongoDB — база даних;

— Express.js — це фреймворк веб-додатків, який працює поверх Node.js;

— Angular.js — фреймворк MVC для інтерфейсу веб-програми на основі браузера;

— Node.js — це платформа JavaScript для розробки серверів.

Використання Node.js дозволяє автоматично включати в проект готовий веб-сервер. У результаті процес розгортання значно спрощується, оскільки потрібна версія веб-сервера чітко визначена разом з іншими залежностями середовища виконання.

Express дозволяє вам керувати маршрутизацією/рендерингом сторінок на стороні сервера, але зосереджується на візуалізації на стороні клієнта за допомогою AngularJS. Крім того, AngularJS однаково добре працює на настільних комп’ютерах і ноутбуках, смартфонах і планшетах.

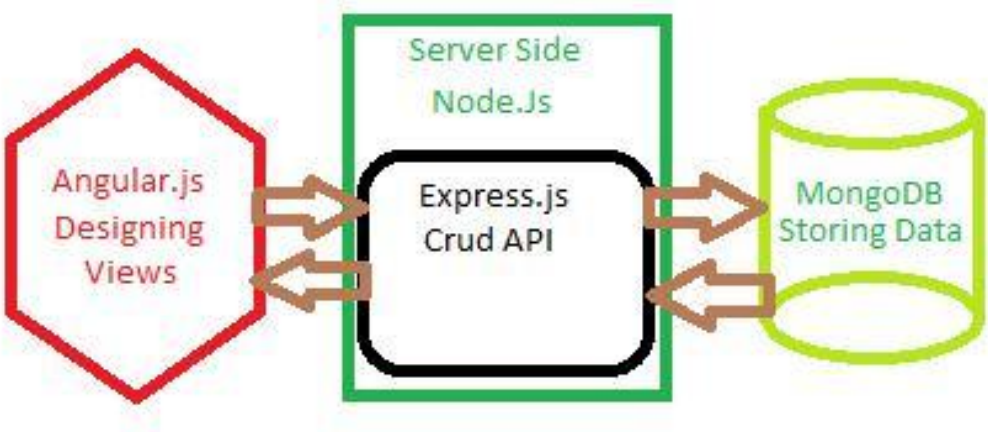

Рисунок 3.7 — Стек технологій MEAN

Слід зазначити, що клієнтська частина збирається за допомогою потокового збирача проектів. Це дозволяє вам автоматизувати створення та мінімізацію

файлів CSS і JS, тестування тощо, тим самим прискорюючи й оптимізуючи процес веб-розробки.

Зв’язок із цією базою даних здійснюється через інструмент Mongoose, який інтегрує базу даних із концепціями об’єктно-орієнтованих мов програмування. Це прискорює розробку, оскільки позбавляє програміста від написання великої кількості інтегрованого коду та дозволяє представляти та взаємодіяти з даними з бази даних у вигляді набору об’єктів.

4 ДОСЛІДЖЕННЯ ПРОГРАМНОГО ЗАСОБУ МОНІТОРИНГУ ТА КОНТРОЛЮ МЕРЕЖНОГО ТРАФІКУ

4.1 Етапи розробки програмного забезпечення

Розробка будь-якого програмного забезпечення складається з декількох етапів, грамотне виконання яких є обов'язковою умовою отримання хорошого результату. Коротко розглянемо кожен етап методології розробки програмного забезпечення.

Першим етапом розробки програмного забезпечення є процедура проведення комплексного аналізу вимог до програмного забезпечення, що створюється, що дозволяє визначити основні цілі та функції кінцевого продукту. На цьому етапі обговорюються деталі проекту, що допомагає більш чітко розробити вимоги до програмного забезпечення. Результатом проведеного аналізу є формування основного документа, на який спирається виконавець у своїй роботі — технічної інструкції (ТІ) з розробки програмного забезпечення. ТІ має [26] повністю описувати завдання, поставлені перед розробником та описувати кінцеву мету проекту.

Наступним етапом розробки програмного забезпечення є етап проектування, тобто моделювання теоретичної основи майбутнього продукту. Найсучасніші засоби програмування дозволяють частково поєднати етапи проектування та кодування, тобто технічну реалізацію проекту, яка орієнтована на об’єктний підхід, але повне планування вимагає ретельного та точного моделювання. Якісний аналіз перспектив і можливостей створюваного продукту стає основою його повноцінної роботи та виконання всіх завдань, поставлених програмою. Одним із компонентів фази проектування, наприклад, є вибір інструментів і операційних систем, яких сьогодні на ринку дуже багато. У рамках цього етапу слід виконати:

— оцінку результатів первинного аналізу та виявлених обмежень;

— пошук важливих напрямків проекту;

— формування остаточної архітектури побудованої системи;

— аналіз необхідності використання програмних модулів або готових рішень від сторонніх розробників;

— проектування основних елементів продукту — бази даних, процесів і кодових моделей;

— вибір середовища програмування та засобів розробки, узгодження інтерфейсу програми, включаючи елементи графічного представлення даних;

— визначення ключових вимог безпеки для програмного забезпечення, що розробляється.

Наступним кроком буде робота безпосередньо з кодом на основі мови програмування, обраної в процесі підготовки. Неможливо описати особливості та нюанси самого трудомісткого та складного етапу, досить сказати, що успіх реалізації будь-якого проекту безпосередньо залежить від якості попереднього аналізу та оцінки конкурентних рішень. Створена програма повинна «боротися» за те, щоб називатися найкращою серед подібних продуктів. Кодування може відбуватися разом із наступним етапом розробки — тестуванням програмного забезпечення, коли зміни вносяться безпосередньо під час написання коду. Рівень і ефективність взаємодії всіх елементів у виконанні розроблених розробником завдань є найважливішим на поточному етапі — від цього залежить якість реалізації проекту.

Перевірка та налагодження дозволяє усунути помилки програмування і досягти кінцевої мети — повноцінної роботи розробленої програми. Процес тестування дозволяє змоделювати ситуації, коли програмний продукт (ПП) перестає працювати. Фаза налагодження локалізує та виправляє виявлений код і приводить його до стану, близького до ідеального. Ці дві фази займають не менше 30% часу, витраченого на весь проект.

Розгортання програмного забезпечення є завершальним етапом розробки і часто відбувається разом із налагодженням системи. Зазвичай впровадження програмного забезпечення відбувається в три етапи:

— початкове завантаження даних;

— поступовий збір інформації;

— випуск створеної програми на проектну потужність.

Основною метою поступового впровадження скомпільованої програми є поступове виявлення раніше невиявлених помилок і недоліків у коді. На цьому етапі розробки програмного забезпечення можуть виникнути дуже «вузькі» помилки, з частковою невідповідністю даних при їх завантаженні в базу даних, а також з перервами у виконанні програмних процедур через використання багатокористувацького доступу. Саме на цьому етапі викристалізовується остаточний образ взаємодії користувача з програмою, а також визначається ступінь лояльності до розробленого інтерфейсу.

Створення навіть невеликого і технічно простого програмного забезпечення залежить від чіткого виконання кожного етапу розробки. Невід'ємною частиною роботи розробників стає чіткий план реалізації необхідних заходів із зазначенням кінцевої мети. Тільки правильно складене ТЗ дозволить досягти бажаного результату та розробити дійсно якісне та конкурентоспроможне програмне забезпечення для будь-якої платформи — серверної, стаціонарної чи мобільної. Подальша технічна підтримка створеного програмного забезпечення в процесі його експлуатації також є невід’ємною частиною завершального етапу розробки програмного забезпечення.

4.2 Графічний інтерфейс користувача

Проект WinForms створено для побудови графічного інтерфейсу користувача у Visual Studio 2022. Консольна версія програми не підходить для поставленого завдання через відсутність засобів управління програмним забезпеченням. Структурно програма складатиметься з трьох основних компонентів:

— інтерфейс програми реалізовано у вигляді графічного вікна, де розміщені елементи керування програмою, а також поля введення та виведення інформації;

— логічна частина програми — це програмний код, який відповідає логіці програми, іншими словами, всі функції, що відповідають за функціонування та роботу компонентів програми;;

— допоміжні бібліотеки потрібні для доступу до мережевих інтерфейсів і перехоплення трафіку (пакетів).

Проект використовуватиме допоміжні бібліотеки (Pcap.dll, SharpPcap.dll, Pcap.NET.dll). Для цього у вкладці «Посилання» додано посилання на вищезазначені бібліотеки. Щоб додати посилання, клацніть правою кнопкою миші на пункті «Анотація» і виберіть у меню пункт «Додати посилання...», як показано на рисунку 4.1 Після цього на екрані з’явиться менеджер пошуку та вибору посилань, як показано на рисунку. 4.2.

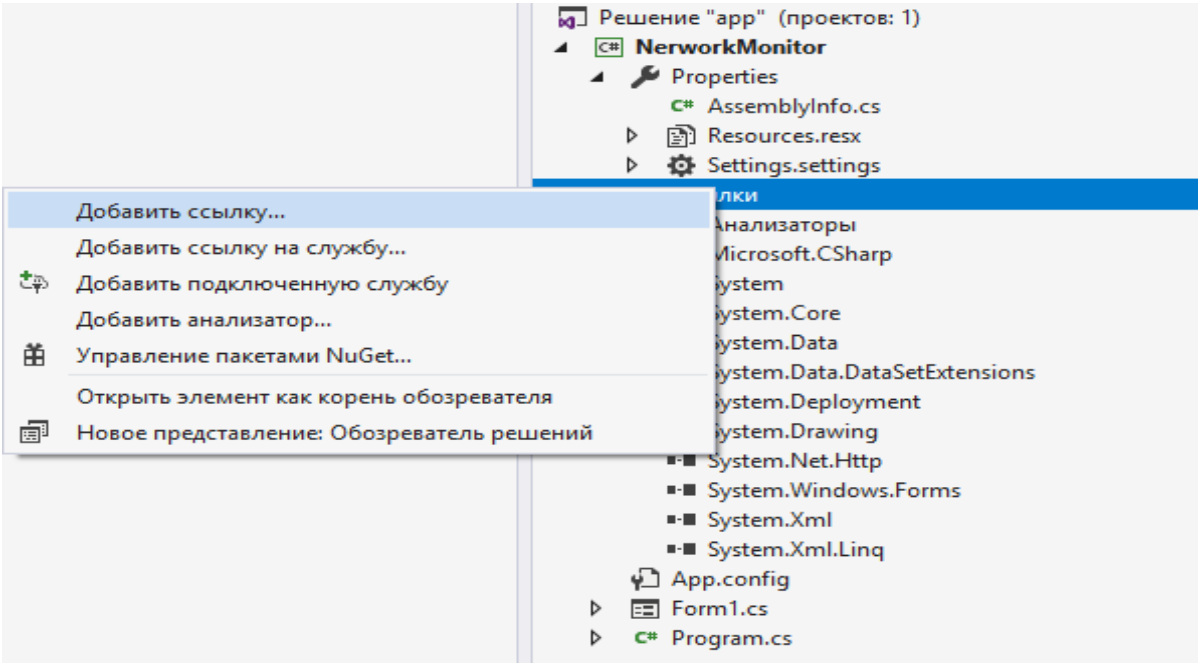

Рисунок 4.1 — Крок додавання посилань на допоміжні бібліотеки

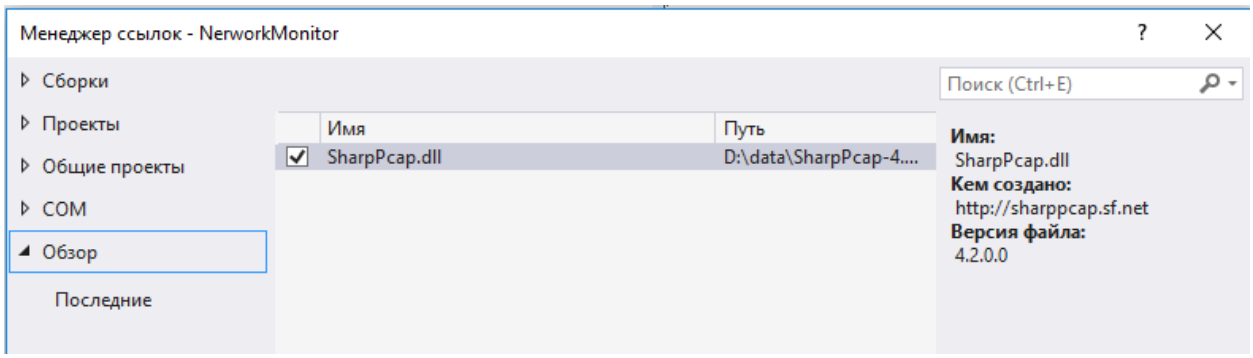

Рисунок 4.2 — Вибір та додавання посилань

На форму були додані наступні компоненти:

cbFirstDev — компонент comboBox для вибору підключення до мережі клієнтами;

cbLastDev — компонент comboBox, для вибору підключення до мережі з другої підмережі (маршрутизатор з Інтернету) tbDest — компонент textBox, для введення MAC-адреси маршрутизатора, підключеного до Інтернету або іншої підмережі;

dgOther — компонент dataGridView, для відображення списку всіх мережевих пристроїв, з яких надсилаються трансляції в мережу (через другий інтерфейс головної машини);

dgClients — компонент dataGridView для відображення списку всіх авторизованих мережевих пристроїв (клієнтів). Термін включає: MAC-адресу, IP-адресу, обсяг трафіку (в байтах);

btRefresh — компонент кнопки для оновлення списків dgOther і dgClients;

AddClient — компонент кнопки для додавання вибраного клієнта зі списку dgOther;

button2 — компонент кнопки для видалення вибраного клієнта зі списку dgClients;

btstart — компонент кнопки для запуску процесу сканування мережі, зупинки пакетів та інших процесів;

btstop — компонент кнопки для зупинки всіх процесів.

Крім того, для ясності та опису всіх елементів додано етикетки. До проекту додано компонент таймера для періодичного оновлення списків dgOther і dgClients. Після додавання всіх необхідних компонентів і позначення їх компонентами ярликів, форма набуде вигляду, показаного на рисунку 4.3.

4.3 UML діаграми класів

Універсальна мова моделювання (UML) — це мова нотації або діаграм, призначена для визначення, візуалізації та документування моделей об’єктно-

орієнтованих програмних систем. Це спосіб візуалізації програмного забезпечення за допомогою набору діаграм. UML не є методом розробки, тобто конструкції цієї мови повинні надавати інформацію про послідовність етапів моделювання, а не надавати інструкцій для створення системи, але ця мова допомагає візуально переглянути макет системи і полегшує співпрацю з іншими розробниками. Розробкою UML керує Група керування об’єктами (OMG). Ця мова є загальновизнаним стандартом для графічного опису програмного забезпечення [26-28].

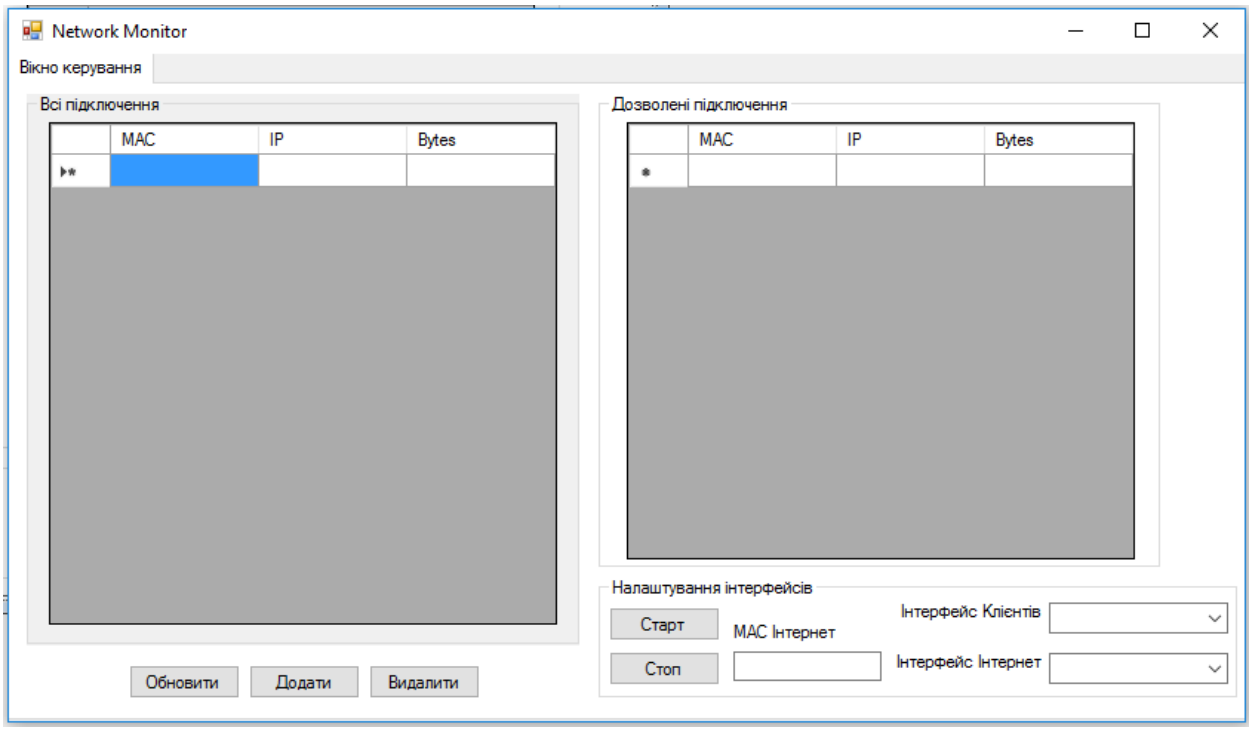

Рисунок 4.3 — Інтерфейс програми

Розроблений для структурування об’єктно-орієнтованого програмного забезпечення, UML є дуже обмеженою мовою для програмування на основі інших парадигм.

Проекти UML складаються з багатьох елементів моделі, які представляють різні частини програмної системи. Елементи UML [28] використовуються для створення діаграм, які відповідають певній частині системи або представленню системи.

Діаграма класів у термінології UML — це діаграма, яка представляє набір класів, які не мають чіткого відношення до дизайну бази даних і зв’язків між цими класами. Крім того, діаграма класів може містити коментарі щодо обмежень. Обмеження можуть бути неофіційно визначені природною мовою або написані мовою OCL (мова обмежень об’єктів). Іншими словами, діаграма класів може представляти лише імена класів або імена класів і відповідні атрибути класу або імена класів, атрибути та операції (методи).

Клас — це опис набору об’єктів, які мають однакові властивості, поведінку, зв’язки та семантику. Кожен клас повинен мати назву, яка відрізнятиме його від інших класів. Ім’я є звичайним текстом, ім’я класу може містити будь-які літери, цифри та знаки пунктуації (за винятком стовпців і крапок) і може бути написане у кількох рядках.

Атрибут — це іменована властивість класу, яка описує діапазон значень, які може приймати екземпляр атрибута. Клас може мати або не мати будь-яку кількість властивостей. У другому випадку блок атрибутів залишається порожнім. Атрибут представляє деяку характеристику змодельованого об'єкта, яка існує в усіх об'єктах цього класу. Ім'я атрибута може бути текстовим, як ім'я класу. На практиці одне або кілька скорочень використовуються для назви атрибута, який представляє деяку властивість відповідного класу атрибута. Типи атрибутів можуть бути визначені в UML, наприклад: розмір, площа, кут, видимість. Останній атрибут може мати такі значення:

— публічний означає, що операції класу доступні для кожного системного об’єкта з будь-якого місця програми; захищений означає, що доступ до операцій класу можливий лише для системних об’єктів, які є екземплярами цього класу або його спадкоємцями;

— private означає, що до операцій класу можуть отримати доступ лише системні об’єкти, які є екземплярами цього класу, тобто визначеного класу.

Діаграма класів, розроблена для програмного засобу, показана на рисунку 4.4.

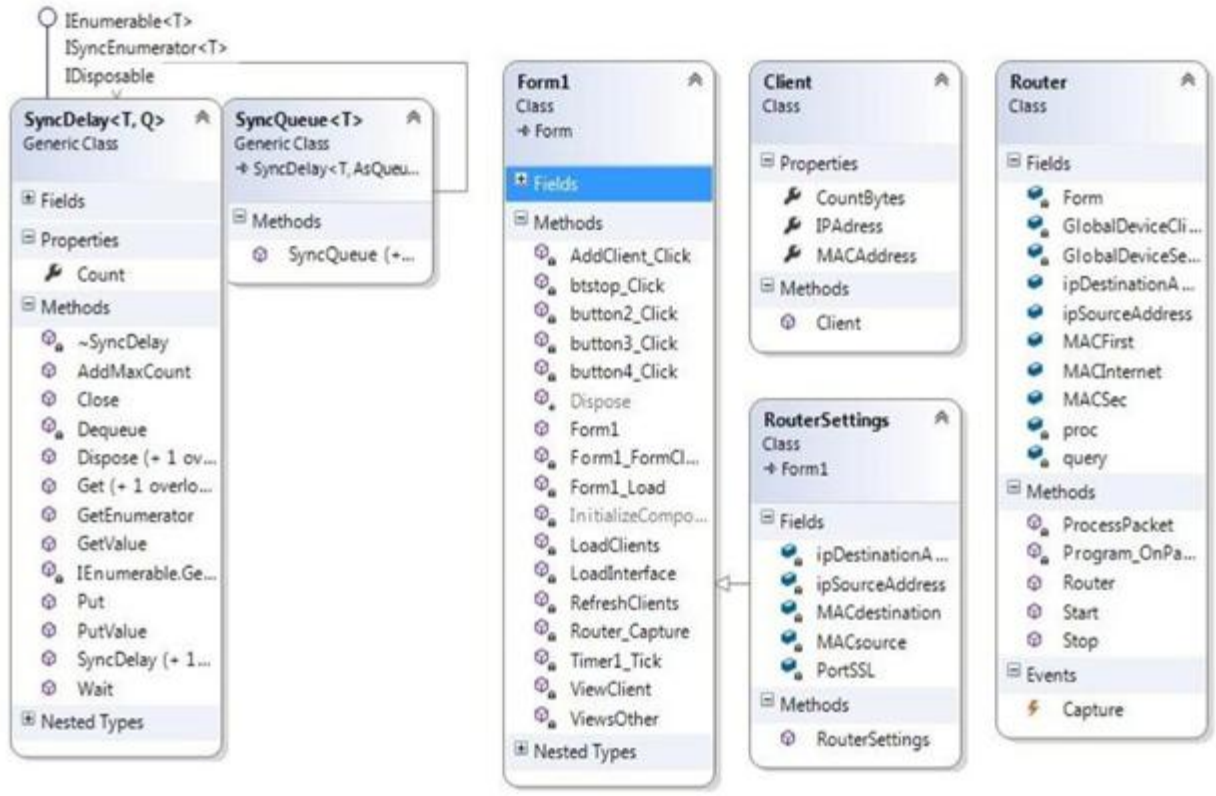

Рисунок 4.4 — Діаграма класів

В ході роботи виділено такі класи:

— Form1 — клас screenView;

— Client — клас даних про клієнта;

— Router — клас відповідає за захоплення та обробку пакетів;

— SyncDelay, SyncQueue — класи опрацювання черг.

4.4 Логіка роботи програми

Перш за все, до програми були підключені бібліотеки SharpPcap.dll і Pcap.NET.dll, які забезпечують доступ до мережевих підключень, пристроїв, а також можливість перехоплювати мережевий трафік і обробляти і модифікувати пакети. Також було підключено бібліотеку System.Net.NetworkInformation. З’єднання вищезгаданих бібліотек із командою use показано на рисунку 4.5.

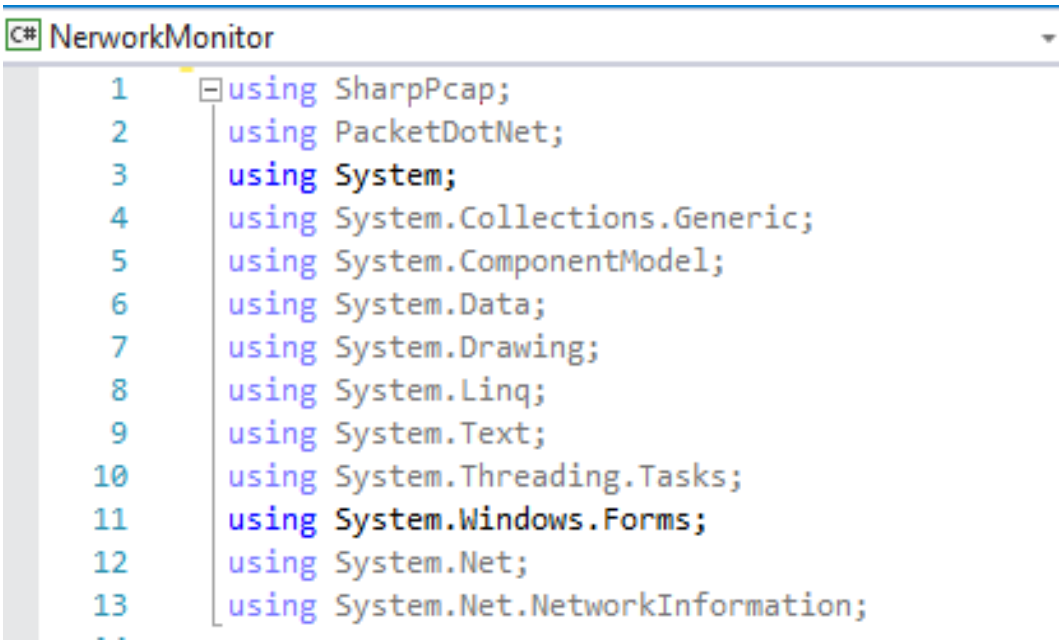

Рисунок 4.5 — Підключення бібліотек в проекті

Спеціальний текстовий файл «Clients.txt» використовується для зберігання інформації про мережеві інтерфейси (клієнти) для пересилання пакетів. Формат введення файлу: [MAC-адреса]: [IP-адреса]: [обсяг трафіку].

Основні розроблені функції програми представлені в Додатку Б. Розглянемо призначення окремих функцій докладніше.

Функція Form1_Load() — це обробник події завантаження форми, тобто функція, яка виконується, коли форма завантажується вперше. Тіло функції забезпечує перевірку наявності файлу «Clients.txt». Якщо такого файлу не існує, у каталозі, де знаходиться файл, буде створено двійковий файл (exe). Якщо такий файл існує, функція виконується LoadInterface(), потім функція LoadClients() і, нарешті, функція ViewClient().

Використовуючи метод CaptureDeviceList.Instance(), ви можете отримати список мережевих підключень (пристроїв) і записати список пристроїв у компоненти cbFirstDev і cbLastDev. Далі виконується функція LoadClients().

В першу чергу інформація про клієнтів зчитується з файлу "Clients.txt" і записується в об'єкт класу Client.

Після читання та запису всіх записів із файлу "Clients.txt" до списку клієнтів, функція ViewClient() викликається для відображення списку клієнтів у формі в компоненті dgClients.

Далі, коли натиснуто кнопку Пуск, викликається функція обробки події button3_Click().

Після запуску функція, яка підключається до вибраних мережевих інтерфейсів, починає процес перехоплення та обробки пакетів. Список клієнтів оновлюється за таймером.

Функція ViewsOther() записує в компонент dgOther список усіх клієнтів, від яких буде отримано трафік під час виконання призупиненого процесу, працює при періодичному виборі за таймером, таймер встановлено на 20 секунд.

Функція AddClient_Click() є обробником подій для натискання кнопки «Додати», копіює вибраний рядок із компонента dgOther у файл «Clients.txt», а звідти — у компонент dgClients.

Функція button4_Click (), коли натиснуто кнопку button4, оновлює список dgClients, dgOther і Clients шляхом виконання функції RefreshClients(). Вона оновлює список клієнтів у файлі "Clients.txt".

Коли форма закрита, виконується функція this.Dispose(). Для завершення (зупинки) процесу призупинення необхідно натиснути на кнопку «Зупинити». Для цього реалізовано функцію обробки кліків btstop_Click(). Функція завершує процес арешту.

Програма реалізує власні стеки та черги для спрощення зберігання та обробки пакетів. Клас Router реалізовано в окремому файлі "Router.cs". Для цього написані такі спеціальні конструктори та методи:

public void Start(first ICaptureDevice, second ICaptureDevice) — приймає два мережеві пристрої як вхідні дані для читання та надсилання мережевого трафіку. Цей спосіб змінює MAC-адреси і перенаправляє пакети (виконує роль маршрутизатора), завдяки чому можна відправити пакети з однієї підмережі в іншу.

public void Stop(Router router) — приймає об'єкт типу Router як вхідні дані, зупиняє реєстрацію пакетів.

void Program_OnPacketArrival10 (відправник об'єкта, CaptureEventArgs e) — метод, який обробляє події отримання нового пакету через інтерфейс.

private void ProcessPacket () — обробка пакетів рівня IP.

4.5 Тестування роботи програми

Перед запуском програми необхідно переконатися, що на локальній машині є два мережеві пристрої, які підключені та працюють належним чином. З'єднання з цими клієнтськими пристроями повинні бути перевірені, інформація про них буде відфільтрована і для них буде реалізована політика обмеження доступу до зовнішньої мережі. Зовнішня мережа може бути локальною мережею з іншою підмережею, ніж перша, і може бути маршрутизатором, який отримує доступ до Інтернету, як показано на рисунку 4.6.

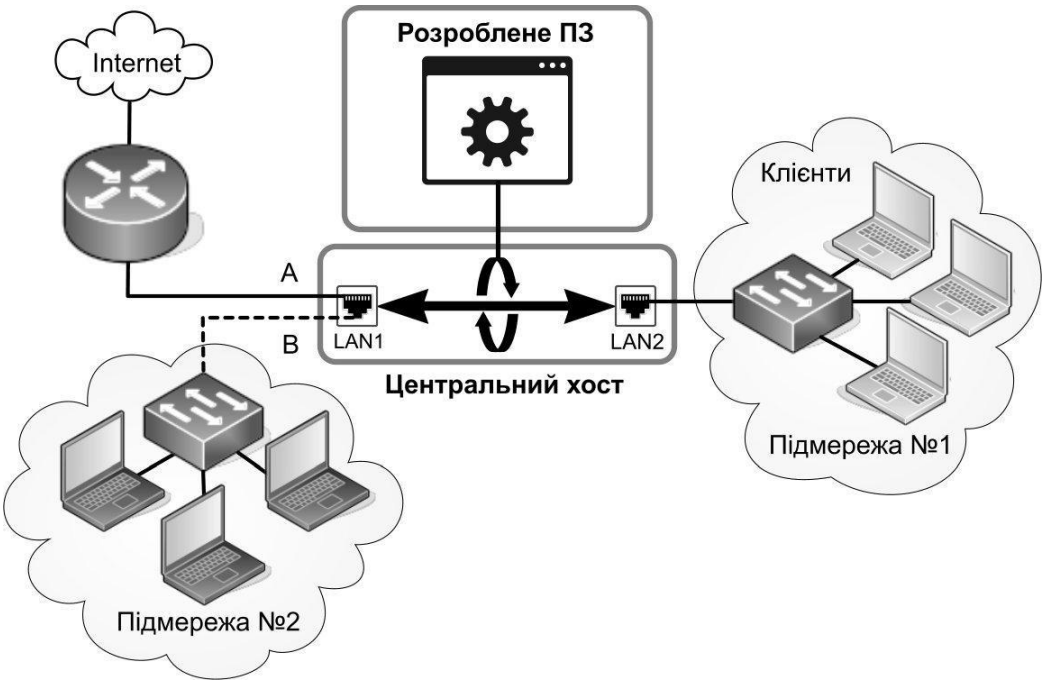

Рисунок 4.6 — Схема варіантів структури мережі

Наступним кроком є перевірка підключення підмережі №1 до відповідного інтерфейсу центральної машини та підмережі №2 з підключеним до неї мережевим пристроєм на центральній машині. Це робиться за допомогою

команди «ping». Після того як всі вузли з’єднані, можна приступати до роботи над програмою.

До мінімальної конфігурації центрального ПК є певні вимоги:

— процесор: Intel Core 2 Duo, AMD Sempron або вище;

— обсяг ОП: не менше 256-512 МБ вільної пам'яті;

— простір на жорсткому диску: мінімум 128 Мб вільного місця;

— принаймні два мережевих інтерфейси, що підтримують режим прослуховування мережі;

— ОС не нижче Windows 7.

Обов'язковими вимогами є:

— версія NET.FrameWork не нижче 4.5.2;

— версія WinPcap не нижче 4.1.3;

— наявність бібліотек SharpPcap.dll, Pcap.NET.dll в директорії з виконуваним модулем програми;

— мережеві екрани на центральній машині, повинні бути відключені в обов'язковому порядку.

Переконавшись, що всі необхідні бібліотеки встановлені, можна запускати програму. Після початку першим кроком є запис MAC-адреси вашого пристрою, на який буде спрямовано трафік (підмережа №1). Це потрібно зробити в текстовому полі «MAS Internet». Потім у списках «Інтерфейс клієнта» та «Інтерфейс Інтернету» слід вибрати відповідні мережеві пристрої. Їх імена в списках відповідають іменам, які вони мають в ОС. Інтерфейс програми під час її роботи представлено на рисунку 4.7.

Натисніть «Пуск», щоб почати процес моніторингу та фільтрації. Після ініціалізації чекаємо появи перших адрес мережевих пристроїв, пакети з яких будуть проходити через мережевий інтерфейс поруч з підмережею №2. Інформація про ці мережеві пристрої відображається у списку «Усі підключення». І запис про сам пристрій у вигляді списку MAC-адреси, IP-адреси та кількості байт.

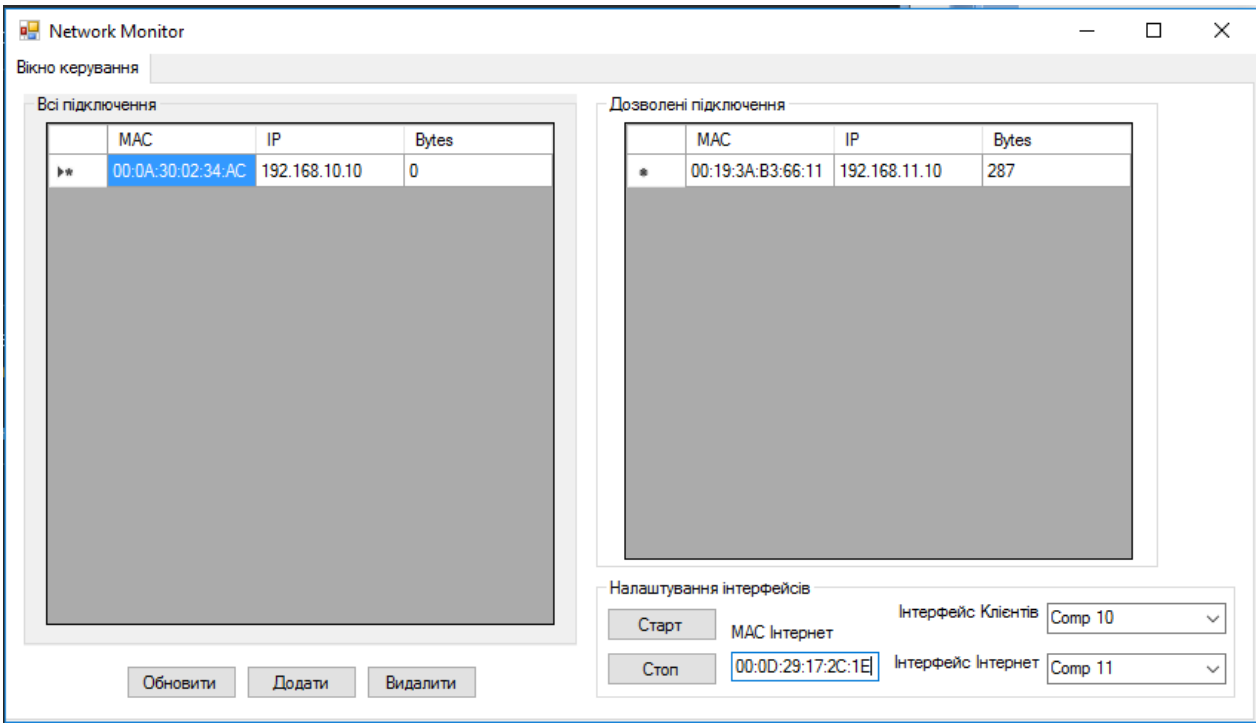

Рисунок 4.7 — Вікно програми під час роботи

Щоб додати мережевий інтерфейс із загального списку в список «Дозволені підключення», необхідно вибрати його в списку «Усі підключення» та натиснути кнопку «Додати»., після цього мережевий пристрій буде включено в список «Дозволених підключень». Так само після внесення пристрою в список «Дозволені підключення» буде розрахований трафік для нього. Для того, щоб не очікувати оновлення інформації в списку за таймером, ви можете скористатися кнопкою «Оновити» і списки оновляться відразу.

Натисніть кнопку «Видалити», щоб видалити пристрій зі списку дозволених підключень. І вибраний мережевий пристрій знову з’явиться в списку «Всі підключення». Щоб зупинити процес моніторингу та фільтрації, натисніть кнопку «Зупинити». Щоб його перезапустити, достатньо буде ще раз натиснути кнопку «Пуск». Якщо закрити програму та запустити її знову, список дозволених клієнтів буде завантажено з файлу. Коли програма закривається, вона запам'ятовує список дозволених клієнтів.

Тестування ПЗ – це [18]:

— процес пошуку програмного забезпечення для завантаження

— інформація про якість продукції;

— процес перевірки відповідності вимогам, запропонованим до виробу, і практично застосовній ознаці, що здійснюється шляхом спостереження за його роботою у штучно створених ситуаціях та обмеженому наборі певним чином підібраних випробувань;

— оцінка системи, щоб знайти різницю між тим, якою система має бути, і тим, чим якою вона є.

У широкому розумінні тестування — це один із методів контролю якості (Quality Control), який включає планування, підготовку тестів, безпосереднє тестування та аналіз отриманих результатів [18].

Системний тест перевіряє програму в цілому, для невеликих проектів це, зазвичай, ручне тестування.

При тестуванні була зроблена спроба підключитися до іншої підмережі за допомогою команди "ring" перед запуском програмного забезпечення. Для цього з машини 192.168.11.10 на вузол 192.168.10.10 була відправлена команда «дзвінок». Але відповіді від цього вузла не було, і це правильно, оскільки, ці хости знаходяться в різних підмережах. Після цього починає працювати програма. Вводиться MAC-адреса зовнішнього вузла та вибираються два мережевих інтерфейси, клієнтський і зовнішній. Потім знову була надіслана команда «дзвінок». Тепер це було необхідно для того, щоб наш клієнт з'явився в загальному списку мережевих пристроїв. Як тільки він з’явився, після вибору та натискання кнопки «Додати», «кільцеві» пакети почали надходити на вузол 192.168.10.10, а відповіді поверталися на 192.168.11.10.

Тестування всіх програмних модулів проводилось на етапі налагодження з усуненням виявлених помилок.

Для перевірки інтерфейсу користувача та практичних функцій (логіки) розроблено набір можливих регулярних і нестандартних дій користувача:

— спроба підключення до зовнішнього вузла мережі з недоступною MAC-адресою;

— спробувати запустити програму з невибраними мережевими інтерфейсами на локальній машині;

— запускати програму без клієнтського файлу;

— відображення некоректного формату даних у файлі клієнта;

- перевірка роботи всіх кнопок;

— перевірка правильності відображення списків;

— перевірка точності таймера;

— перевірка правильності перехоплення та доставки посилок;

— перевірка правильності обчислення та показу об’єму руху;

— перевірка виходу програмного забезпечення; - проходження всього терміну використання програмного забезпечення.

В результаті перевірки виявлені помилки були усунені, в результаті повторних тестів додаткових помилок виявлено не було. Однак, не виключено, що є помилки в інших системах ОС, мережевих пристроях. Це також може залежати від версій бібліотеки, типів мережевих пристроїв і деяких інших факторів.

В цілому, програмне забезпечення за результатами тестування показало стабільну роботу і готове до використання.

4.6 Застосунок для мобільних пристроїв

Також був розроблений за стосунок для мобільних пристроїв, який надає прості та розширені способи блокування доступу до Інтернету. Програмам і адресам можна окремо дозволити або заборонити доступ до вашого Wi-Fi та/або мобільного з’єднання.

Блокування доступу до Інтернету може допомогти:

— зменшити використання даних;

— бережіть акумулятор;

— збільшити конфіденційність.

Особливості розробленого за стосунку наступні:

— простий у використанні;

— root не потрібен;

— 100% відкритий код;

— без відстеження чи аналітики;

— без реклами;

— активно розвивається та підтримується;

— підтримка Android 5.1 і новіших версій;

— підтримка IPv4/IPv6 TCP/UDP;

— підтримка модему;

— підтримка кілька користувачів пристрою;

— опціональне блокування в роумінгу;

— додаткове блокування системних програм;

— додаткове повідомлення, коли програма отримує доступ до Інтернету;

— додатково можна записувати використання мережі для кожної програми;

— дизайн зі світлою та темною темами.

Жоден інший брандмауер без root-доступу не пропонує всі ці функції.

Також в додатку доступні такі професійні функції, які доцільно в подальшому доцільно перевести на комерційну основу:

— перегляд журналу трафіку — дозволяє відображати та експортувати докладний журнал вихідного IP-трафіку для всіх програм , щоб мати можливість точно бачити, що робить пристрій у будь-який момент;

— фільтрувати мережевий трафік — дозволяє вибірково блокувати мережевий трафік для кожної адреси для кожної програми, тож можете заблокувати програмам дзвонити додому тощо, зберігаючи доступ до Інтернету; увімкнути/вимкнути режим фільтрації (доступні з Android 7 Nougat);

— сповіщення про нові програми — вмикає сповіщення про нові програми, за допомогою яких можна безпосередньо заблокувати або дозволити програму;

— сповіщення на графіку швидкості мережі — дозволяє відображати поточну швидкість мережі у вигляді графіка в сповіщенні рядка стану (три

найпопулярніші програми відображаються на Android 5 Lollipop або старіших версіях);

— зовнішній вигляд (тема, кольори) — дозволяє вибрати з п’яти додаткових тем програми, усі доступні у світлому та темному варіантах;

— підтримка розробки — підтримка поточної розробки, як-от додавання нових функцій, покращення наявних функцій, виправлення помилок і адаптація застосунку до нових версій Android.

Вікна роботи застосунку показано рисунках 4.7 -4.13.

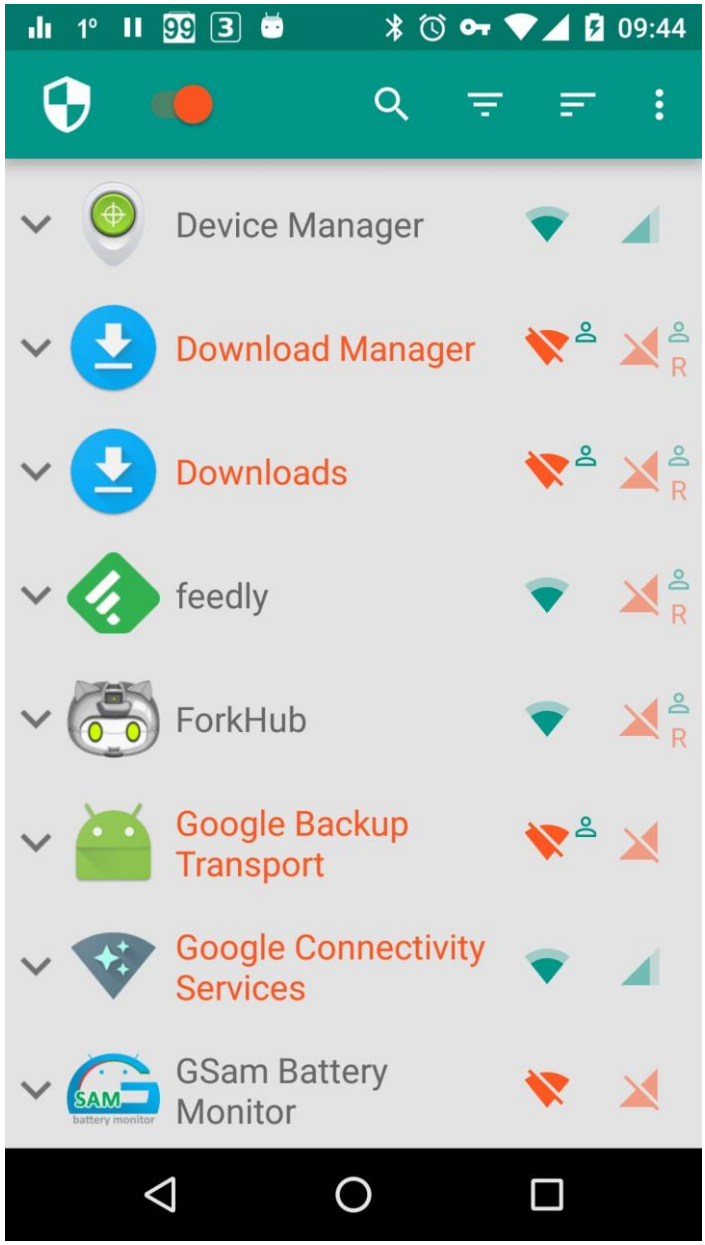

Рисунок 4.7 — Головне вікно застосунку

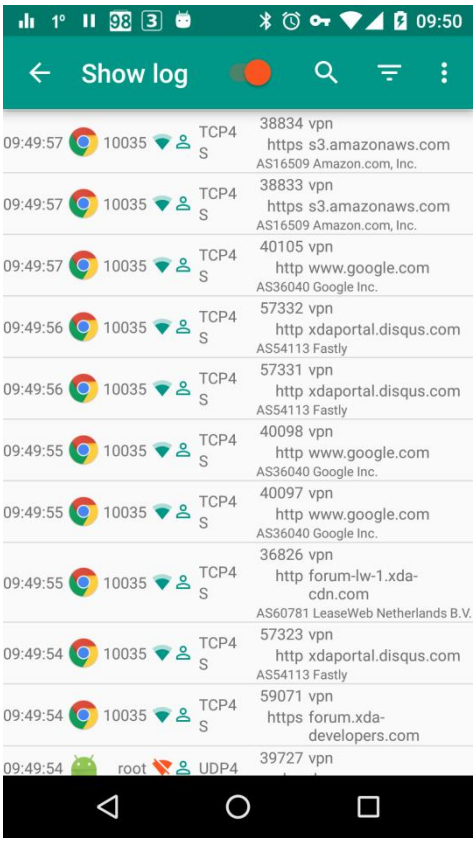

Рисунок 4.8 — Глобальний журнал трафіку

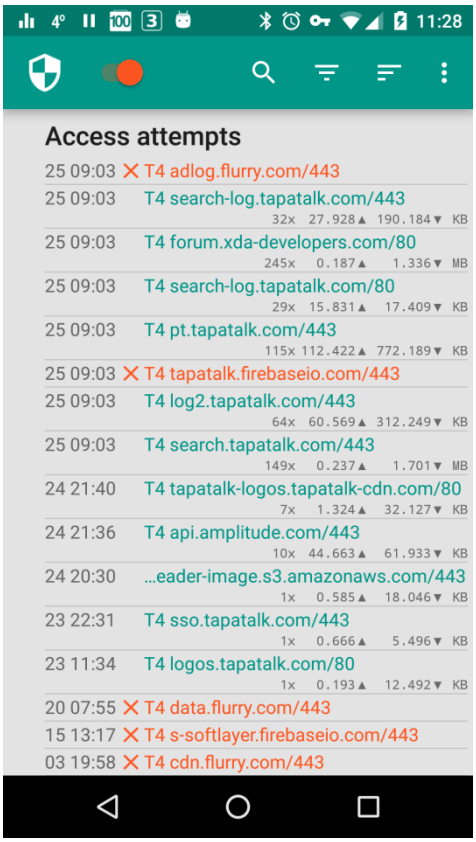

Рисунок 4.9 — Журнал трафіку програми

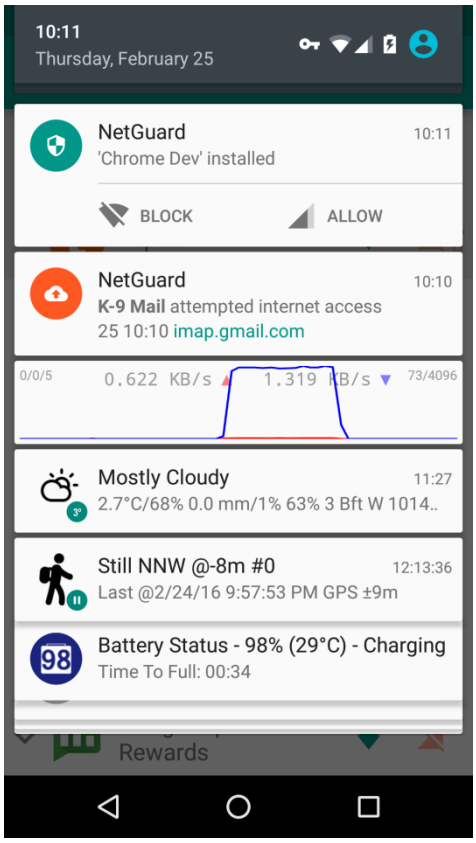

Рисунок 4.10 — Вікно сповіщень

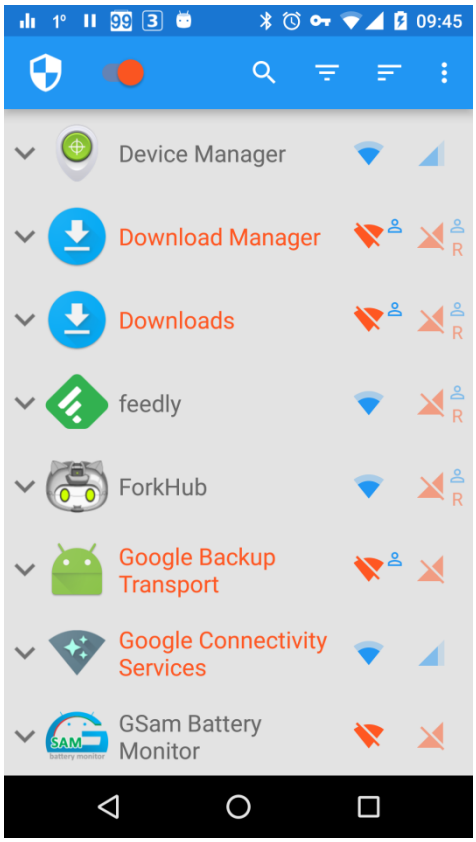

Рисунок 4.11 — Синьо-помаранчева світла тема

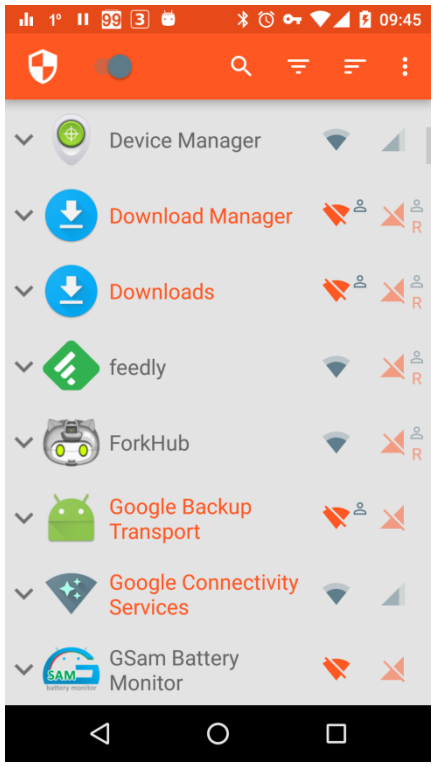

Рисунок 4.12 — Помаранчево-сіра світла тема

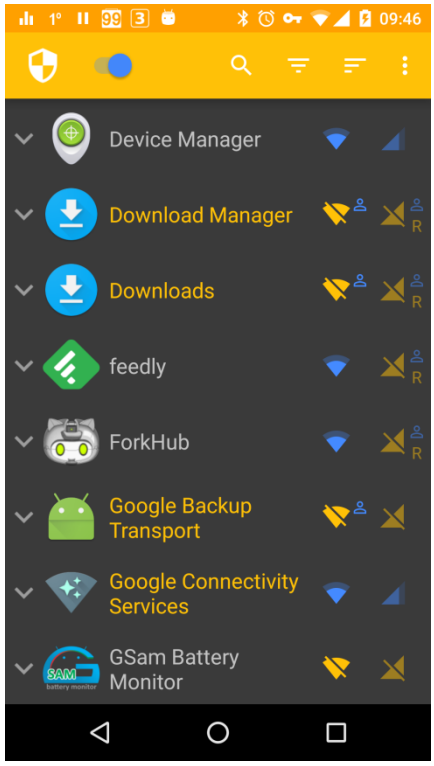

Рисунок 4.13 — Жовто-синя темна тема

В цілому, програмне забезпечення за результатами тестування показало стабільну роботу і готове до використання.

5 ЕКОНОМІЧНА ЧАСТИНА

5.1 Комерційний та технологічний аудит науково-технічної розробки

Метою даного розділу є проведення технологічного аудиту, в даному випадку нового програмного засобу для моніторингу та контролю мережевого трафіку мобільних пристроїв. Розроблюваний продукт має розширення функціональних можливостей засобу моніторингу та контролю мережевого трафіку. Метою розробки є вдосконалення системи збору мережного трафіку мобільних пристроїв та розширення її функціональних можливостей для аналізу та виявлення несанкціонованої активності.

Актуальність зумовлена тим, що на даний момент активно розробляються і застосовуються різні методи моніторингу та контролю трафіку, але вони не завжди є ефективними на практиці. Внаслідок цього всі технології захисту постійно вивчаються та покращуються.

Аналогом може бути комп'ютерна програма програма 3G Watchdog за ціною 3000 $ (117000 тис грн.)

Для проведення комерційного та технологічного аудиту залучають не менше 3-х незалежних експертів. Оцінювання науково-технічного рівня розробки та її комерційного потенціалу рекомендується здійснювати із застосуванням п'ятибальної системи оцінювання за 12-ма критеріями, у відповідності із табл. 5.1.

Таблиця 5.1 — Рекомендовані критерії оцінювання комерційного потенціалу розробки та їх можлива бальна оцінка

Бали (за 5-ти бальною шкалою)					
Крите-рій	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 роботоздат-ність в реальних умовах

Продовження табл. 5.1

Ринкові переваги					
2	Багато аналогів на малому ринку	Ринкові п 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 властивості 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 .	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає
Практик на 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 із 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 використовуються у виробництві

Продовження табл. 5.1

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

Усі дані по кожному параметру занесено в таблиці 5.2

Таблиця 5.2 — Результати оцінювання комерційного потенціалу розробки

Критерії оцінювання	ПІБ експертів		
	Експерт 1	Експерт 2	Експерт 3
	Бали		
Технічна здійсненність концепції	3	3	3
Наявність аналогів на ринку	3	3	4
Цінова політика	4	4	4
Технічні та споживчі властивості виробу	4	3	3
Експлуатаційні витрати	3	4	3
Ринок збуту	4	3	4
Конкурентоспроможність	3	3	3
Фахівці з технічної і комерційної реалізації	4	3	4
Фінансування	4	4	3
Матеріально-технічна база	3	3	3
Термін реалізації ідеї	4	4	4
Супровідна документація	3	3	3
Сума	42	40	41
Середньоарифметична сума балів	(42+40+41) / 3 = 41		

За даними таблиці 5.2 можна зробити висновок щодо рівня комерційного потенціалу даної розробки. Для цього доцільно скористатись рекомендаціями, наведеними в таблиці 5.3.

Таблиця 5.3 — Рівні комерційного потенціалу розробки

Середньоарифметична сума балів СБ , розрахована на основі висновків експертів	Рівень комерційного потенціалу розробки
0-10	Низький
11-20	Нижче середнього
21-30	Середній
31-40	Вище середнього
41-48	Високий

Як видно з таблиці, рівень комерційного потенціалу розроблюваного нового програмного продукту є високим, що досягається за рахунок того, що програмний продукт відрізняється від існуючих тим, що дана технологія має розширення функціональних можливостей засобу моніторингу та контролю мережевого трафіку, а також вдосконалену систему збору мережного трафіку мобільних пристроїв.

5.2 Прогнозування витрат на виконання науково-дослідної (дослідно-конструкторської) роботи

5.2.1 Основна заробітна плата розробників, яка розраховується за формулою:

$$З_о = \frac{M}{T_р} \cdot t \text{,} \qquad (5.1)$$

де М — місячний посадовий оклад конкретного розробника (дослідника), грн.;

$T_р$ — число робочих днів за місяць, 20 днів;

t — число днів роботи розробника (дослідника).

Результати розрахунків зведемо до таблиці 5.4.

Таблиця 5.4 — Основна заробітна плата розробників

Найменування посади	Місячний посадовий оклад, грн.	Оплата за робочий день, грн.	Число днів роботи	Витрати на заробітну плату, грн.
Керівник проекту	40000	2000,00	42	84000,000
Програміст	35000	1750,00	42	73500,000
Всього				157500,00

Так як в даному випадку розробляється програмний продукт, то розробник виступає одночасно і основним робітником, і тестувальником розроблюваного програмного продукту.

5.2.2 Додаткова заробітна плата розробників, які приймали участь в розробці обладнання.

Додаткова заробітна плата прийнято розраховувати як 13 % від основної заробітної плати розробників та робітників:

$$З_д = З_о \cdot 13\ \% / 100\ \% \qquad (5.2)$$

$$З_д = (157500{,}00 \cdot 13\ \% / 100\ \%) = 20475{,}00 \ \text{(грн.)}$$

5.2.3 Нарахування на заробітну плату розробників.

Згідно діючого законодавства нарахування на заробітну плату складають 22 % від суми основної та додаткової заробітної плати.

$$Н_з = (З_о + З_д) \cdot 22\ \% / 100\% \qquad (5.3)$$

$$Н_з = (157500{,}00 + 20475{,}00) \cdot 22\ \% / 100\ \% = 39154{,}50 \ \text{(грн.)}$$

5.2.4. Оскільки для розроблювального програмного продукту не потрібно витрачати матеріали та комплектуючи, то витрати на матеріали і комплектуючі дорівнюють нулю.

5.2.5 Амортизація обладнання, яке використовувалось для проведення розробки.

Амортизація обладнання, що використовувалось для розробки в спрощеному вигляді амортизація обладнання, що використовувалась для розробки розраховується за формулою:

$$A = \frac{Ц}{Тв} \cdot \frac{t_{вик}}{12} \text{ [грн.].} \qquad (5.4)$$

де Ц — балансова вартість обладнання, грн.;

Т — термін корисного використання обладнання згідно податкового законодавства, років

$t_{вик}$ — термін використання під час розробки, місяців

Розрахуємо, для прикладу, амортизаційні витрати на комп'ютер балансова вартість якого становить 22000 грн., термін його корисного використання згідно податкового законодавства – 2 роки, а термін його фактичного використання – 2,10 міс.

$$A_{обл} = \frac{22000}{2} \times \frac{2{,}1}{12} = 1925 \; грн.$$

Аналогічно визначаємо амортизаційні витрати на інше обладнання та приміщення. Розрахунки заносимо до таблиці 5.5.

Таблиця 5.5 — Амортизаційні відрахування матеріальних і нематеріальних ресурсів для розробників

Найменування обладнання	Балансова вартість, грн.	Строк корисного використання, років	Термін використання обладнання, місяців	Амортиза-ційні відрахуван-ня, грн.
Комп'ютер та комп'ютерна периферія (Ноутбук Acer)	22000	2	2,10	1925,000
Офісне обладнення (меблі)	25000	4	2,10	1093,750
Приміщення	1000000	20	2,10	8750,000
Ліцензійне ПЗ (Microsoft Windows 10, Java 17)	31000	2	2,10	2712,500
Всього				14481,25

Тарифи на електроенергію для непобутових споживачів (промислових підприємств) відрізняються від тарифів на електроенергію для населення. При цьому тарифи на розподіл електроенергії у різних постачальників (енергорозподільних компаній), будуть різними. Крім того, розмір тарифу залежить від класу напруги (1-й або 2-й клас). Тарифи на розподіл електроенергії для всіх енергорозподільних компаній встановлює Національна комісія з регулювання енергетики і комунальних послуг (НКРЕКП). Витрати на силову електроенергію розраховуються за формулою:

$$В_е = В \cdot П \cdot Ф \cdot К_п, \quad (5.5)$$

де В — вартість 1 кВт-години електроенергії для 1 класу підприємства, В = 6,2 грн./кВт;

П — встановлена потужність обладнання, кВт. П = 0,4 кВт;

Ф — фактична кількість годин роботи обладнання, годин.

$К_п$ — коефіцієнт використання потужності, $К_п$ = 0,9.

$$В_е = 0,9 \cdot 0,5 \cdot 8 \cdot 42 \cdot 6,2 = 937,44 \text{ (грн.)}$$

5.2.6 Інші витрати та загальновиробничі 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» розраховуються як 50…100% від суми основної заробітної плати дослідників:

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

де $Н_{ів}$ — норма нарахування за статтею «Інші витрати».

$$I_в = 157500{,}00 * 75\% / 100\% = 118125 \text{ (грн.)}$$

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» розраховуються як 100…150% від суми основної заробітної плати дослідників:

$$Н_{нзв} = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%}, \quad (5.7)$$

де $Н_{нзв}$ — норма нарахування за статтею «Накладні (загальновиробничі) витрати».

$$Н_{нзв} = 157500{,}00 * 115\ \% / 100\ \% = 181125 \text{ (грн.)}$$

5.2.7 Витрати на проведення науково-дослідної роботи.

Сума всіх попередніх статей витрат дає загальні витрати на проведення науково-дослідної роботи:

$$B_{заг} = 157500,00+20475,00+39154,50+14481,25+31000+937,44+118125+$$
$$+181125 = 562798,19 \text{ грн.}$$

5.2.8 Розрахунок загальних витрат на науково-дослідну (науково-технічну) роботу та оформлення її результатів.

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

$$ЗВ = \frac{B_{заг}}{\eta} \text{ (грн),} \qquad (5.8)$$

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

Так, якщо науково-технічна розробка знаходиться на стадії: науково-дослідних робіт, то η=0,1; технічного проектування, то η=0,2; розробки конструкторської документації, то η=0,3; розробки технологій, то η=0,4; розробки дослідного зразка, то η=0,5; розробки промислового зразка, то η=0,7; впровадження, то η=0,9. Оберемо η = 0,5, так як розробка, на даний момент, знаходиться на стадії дослідного зразка:

$$ЗВ = 562798,19 / 0,5 = 1125596 \text{ грн.}$$

5.3 Розрахунок економічної ефективності науково-технічної розробки за її можливої комерціалізації потенційним інвестором

В ринкових умовах узагальнювальним 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. Саме зростання чистого прибутку забезпечить потенційному інвестору надходження додаткових коштів, дозволить покращити фінансові результати його діяльності, підвищить конкурентоспроможність та може позитивно вплинути на ухвалення рішення щодо комерціалізації цієї розробки.

Для того, щоб розрахувати можливе зростання чистого прибутку у потенційного інвестора від можливого впровадження науково-технічної розробки необхідно:

— вказати, з якого часу можуть бути впроваджені результати науково-технічної розробки;

— зазначити, протягом скількох років після впровадження цієї науково-технічної розробки очікуються основні позитивні результати для потенційного інвестора (наприклад, протягом 3-х років після її впровадження);

— кількісно оцінити величину існуючого та майбутнього попиту на цю або аналогічні чи подібні науково-технічні розробки та назвати основних суб’єктів (зацікавлених осіб) цього попиту;

— визначити ціну реалізації на ринку науково-технічних розробок з аналогічними чи подібними функціями.

При розрахунку економічної ефективності потрібно обов’язково враховувати зміну вартості грошей у часі, оскільки від вкладення інвестицій до отримання прибутку минає чимало часу. При оцінюванні ефективності інноваційних проектів передбачається розрахунок таких важливих показників:

— абсолютного економічного ефекту (чистого дисконтованого доходу);

— внутрішньої економічної дохідності (внутрішньої норми дохідності);

— терміну окупності (дисконтованого терміну окупності).

Аналізуючи напрямки проведення науково-технічних розробок, розрахунок економічної ефективності науково-технічної розробки за її можливої комерціалізації потенційним інвестором можна об'єднати, враховуючи визначені ситуації з відповідними умовами.

5.3.1 Розробка чи суттєве вдосконалення програмного засобу (програмного забезпечення, програмного продукту) для використання масовим споживачем.

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

$$\Delta П_i = (\pm \Delta Ц_0 \cdot N + Ц_0 \cdot \Delta N)_i \cdot \lambda \cdot \rho \cdot (1 - \frac{\vartheta}{100}), \qquad (5.9)$$

де $\pm\Delta Ц_o$ — розрахунок економічної ефективності науково-технічної розробки за її можливої комерціалізації потенційним інвестором

зміна вартості програмного продукту (зростання чи зниження) від впровадження результатів науково-технічної розробки в аналізовані періоди часу;

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

$Ц_o$ — основний оціночний показник, який визначає діяльність підприємства у даному році після впровадження результатів наукової розробки, $Ц_o = Ц_б \pm \Delta Ц_o$;

$Цб$ — вартість програмного продукту у році до впровадження результатів розробки;

ΔN — збільшення кількості споживачів продукту, в аналізовані періоди часу, від покращення його певних характеристик;

λ — коефіцієнт, який враховує сплату податку на додану вартість. Ставка податку на додану вартість дорівнює 20%, а коефіцієнт $\lambda = 0{,}8333$.

p — коефіцієнт, який враховує рентабельність продукту;

ϑ — ставка податку на прибуток, у 2023 році ϑ =18%.

Припустимо, що при прогнозованій ціні 12000 грн. за одиницю виробу, термін збільшення прибутку складе 3 роки. Після завершення розробки і її вдосконалення, можна буде підняти її ціну на *500* грн. Кількість одиниць реалізованої продукції також збільшиться: протягом першого року – на 3000 шт., протягом другого року – на 4000 шт., протягом третього року на 5000 шт. До моменту впровадження результатів наукової розробки реалізації продукту не було:

$$\Delta П_1 = (0*500 + (12000 + 500)*3000)* 0{,}8333* 0{,}35) * (1 - 0{,}18) = 8609999{,}656 \text{ грн.}$$

$$\Delta П_2 = (0*500 + (12000 + 500)*(3000+4000)* 0{,}8333* 0{,}35) * (1 - 0{,}18) = 20927082{,}496 \text{ грн.}$$

$$\Delta П_3 = (0*500 + (12000 + 500)*(3000+4000+5000)* 0{,}8333* 0{,}35) * (1 - 0{,}18) = 35874998{,}565 \text{ грн.}$$

Отже, комерційний ефект від реалізації результатів розробки за три роки складе 65412080,72 грн.

5.3.2 Розрахунок ефективності вкладених інвестицій та періоду їх окупності.

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

$$ПП = \sum_{1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, \qquad (5.10)$$

де ΔП — збільшення чистого прибутку у кожному із років, протягом яких виявляються результати виконаної та впровадженої науково-дослідної (науково-технічної) роботи, грн;

Т — період часу, протягом якою виявляються результати впровадженої науково-дослідної (науково-технічної) роботи, роки;

τ — ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні, τ = 0,05…0,15;

t — період часу (в роках).

Збільшення прибутку ми отримаємо починаючи з першого року:

$$ПП=(8609999,656/(1+0,1)^1)+(20927082,496/(1+0,1)^2)+(35874998,565/(1+0,1)^3)=$$
$$=7827272,41 + 17295109,5 + 26953417,4 = 52075799,32 \text{ грн.}$$

Далі розраховують величину початкових інвестицій *PV*, які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної розробки. Для цього можна використати формулу:

$$PV = k_{інв} * ЗВ, \quad (5.11)$$

де $k_{інв}$ — коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію. Це можуть бути витрати на підготовку приміщень, розробку технологій, навчання персоналу, маркетингові заходи тощо; зазвичай $k_{інв}=2...5$, але може бути і більшим;

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

$$PV = 2 * 1125596 = 2251192,76 \text{ грн.}$$

Тоді абсолютний економічний ефект $Е_{абс}$ або чистий приведений дохід (NPV, Net Present Value) для потенційного інвестора від можливого впровадження та комерціалізації науково-технічної розробки становитиме:

$$E_{абс} = ПП – PV, \quad (5.12)$$

$$E_{абс} = 52075799{,}32 - 2251192{,}76 = 49824606{,}56 \text{ грн.}$$

Оскільки $E_{абс} > 0$ то вкладання коштів на виконання та впровадження результатів даної науково-дослідної (науково-технічної) роботи може бути доцільним.

Для остаточного прийняття рішення з цього питання необхідно розрахувати внутрішню економічну дохідність або показник внутрішньої но-рми дохідності *(IRR, Internal Rate of Return)* вкладених інвестицій та порівняти її з так званою бар'єрною ставкою дисконтування, яка визначає ту мінімальну внутрішню економічну дохідність, нижче якої інвестиції в будь-яку науково-технічну розробку вкладати буде економічно недоцільно.

Розрахуємо відносну (щорічну) ефективність вкладених в наукову розробку інвестицій $E_в$. Для цього використаємо формулу:

$$E_в = \sqrt[T_ж]{1 + \frac{E_{абс}}{PV}} - 1, \tag{5.13}$$

$T_ж$ — життєвий цикл наукової розробки, роки.

$$E_в = \sqrt{з\ (1 + 49824606{,}56/2251192{,}76} - 1 = 1{,}849$$

Визначимо мінімальну ставку дисконтування, яка у загальному вигляді визначається за формулою:

$$\tau = d + f, \tag{5.14}$$

де d — середньозважена ставка за депозитними операціями в комерційних банках; в 2023 році в Україні d = (0,09…0,14);

f — показник, що характеризує ризикованість вкладень; зазвичай, величина f = (0,05...0,5).

$$\tau_{\min} = 0{,}14 + 0{,}05 = 0{,}19.$$

Так як $E_в > \tau_{min}$, то інвестор може бути зацікавлений у фінансуванні даної наукової розробки.

Розрахуємо термін окупності вкладених у реалізацію наукового проекту інвестицій за формулою:

$$T_{ок} = \frac{1}{E_в}, \qquad (5.15)$$

$$T_{ок} = 1 / 1{,}849 = 0{,}54 \text{ р.}$$

Оскільки $T_{ок}$ < 3-х років, а саме термін окупності рівний 0,54 роки, то фінансування даної наукової розробки є доцільним.

Висновки до розділу: економічна частина даної роботи містить розрахунок витрат на розробку нового програмного продукту, сума яких складає 1125596 гривень. Було спрогнозовано орієнтовану величину витрат по кожній з статей витрат. Також розраховано чистий прибуток, який може отримати виробник від реалізації нового технічного рішення, розраховано період окупності витрат для інвестора та економічний ефект при використанні даної розробки. В результаті аналізу розрахунків можна зробити висновок, що розроблений програмний продукт за ціною дешевший за аналог і є висококонкурентоспроможним. Період окупності складе близько 0,54 роки.

ВИСНОВКИ

Вивчення сучасних методів та засобів моніторингу і контролю мережевого трафіку показало, що статистичне дослідження провайдерського трафіку, що надходить з локальної мережі в зовнішню мережу, в даний час не використовується в системах безпеки. Основним напрямком таких систем є моніторинг роботи обладнання, навантаження на канали; а для локальних мереж — це захист від загроз та несанкціонованої активності із зовнішньої мережі.

Розроблена архітектура добре показала себе в рамках прототипу, але для подальшого розвитку необхідно перевірити інші бази даних, щоб вибрати найкращу для роботи в даних умовах і продумати архітектуру збережених даних.

Надалі слід враховувати, що отримані дані мають великий обсяг, і для цього необхідно додати оптимізацію, а також консультації з цільовими користувачами системи для впровадження відповідних алгоритмів дослідження трафіку.

В рамках магістерської роботи:

— досліджено існуючі систем моніторингу та контролю мережевого трафіку, що дозволило визначити їх переваги та недоліки;

— обґрунтувано вибір інструментів розробки, що забезпечить ефективне проектування із застосуванням сучасних програмних засобів і технологій;

— запропоновано оптимальну архітектуру системи та створено алгоритми обробки даних для її функціонування;

— реалізовано програмний засіб системи виявлення шкідливого трафіку з урахуванням розробленої архітектури та алгоритмів і проведено його тестування.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Machine Learning Algorithms for Network Intrusion Detection [Електронний ресурс]: Режим доступу https://link.springer.com/chapter/10.1007/978-3-319-98842-9_6. Дата доступу 01.11.2023
2. What is the difference between functional and non functional requirement? [Електронний ресурс]: Режим доступу: https://stackoverflow.com/questions/16475979/what-is-the-difference-between- functional-and-non-functional-requirement. Дата доступу 01.11.2023
3. Monster Logs [Електронний ресурс]: Режим доступу: https://zeek.org/2012/01/04/monster-logs/. Дата доступу 01.11.2023
4. B. A. Forouzan, Data Communications and Networking, 5:th ed. Mcgroy-Hill, 2013
5. H. Alaidaros і M. Mahmuddin, " Flow-Based approach on Bro Intru-Sion Detection, " Journal of Telecommunication, Electronic and Computer Engineering (JTEC), vol. 9, № 2-2, С. 139-145, 2017
6. G. James, D. Witten, T. Hastie, and R. Tibshirani, An Introduction toStatistical Learning, 1:st ed.Springer, 2017
7. Support-vector networks [Електронний ресурс]: Режим доступу: https://link.springer.com/article/10.1007/BF00994018/. Дата доступу 01.11.2023
8. Ben-Hur, Asa, Horn, David, Siegelmann, Hava, and Vapnik, Vladimir; "Support vector clustering" (2001) Journal of Machine Learning Research, 2: 125–137
9. Vapnik, V.: Invited Speaker. IPMU Information Processing and Management of Uncertainty in Knowlendge-Based Systems (2014)
10. Barghout, Lauren. "Spatial-Taxon Information Granules as Used in Iterative Fuzzy- Decision-Making for Image Segmentation." Granular Computing and Decision- Making. Springer International Publishing, 2015. 285-318.
11. Bilwaj Gaonkar, Christos Davatzikos “Analytic estimation of statistical significance maps for support vector machine based multi-variate image analysis and classification”, April 2013.

12. R. Cuingnet, C. Rosso, M. Chupin, S. Lehéricy, D. Dormont, H. Benali, Y. Samson and O. Colliot, “Spatial regularization of SVM for the detection of diffusion alterations associated with stroke outcome, Medical Image Analysis”, 2011, 15 (5): 729–737
13. Statnikov, A., Hardin, D., & Aliferis, C., “Using SVM weight-based methods to identify causally relevant and non-causally relevant variables”. 2006.
14. Piryonesi S. Madeh; El-Diraby Tamer E. (2020-06-01). "Role of Data Analytics in Infrastructure Asset Management: Overcoming Data Size and Quality Problems". Journal of Transportation Engineering, Part B: Pavements.
15. Hastie, Trevor. The elements of statistical learning : data mining, inference, and prediction : with 200 full-color illustrations. Tibshirani, Robert., Friedman, J. H. (Jerome H.). New York: Springer. 2001
16. Terrell, George R.; Scott, David W. (1992). "Variable kernel density estimation". Annals of Statistics, 1992. – 1236–1265.
17. Mills, Peter. "Efficient statistical classification of satellite measurements". International Journal of Remote Sensing, 2012.
18. Deng,H.; Runger, G.; Tuv, E. Bias of importance measures for multi-valued attributes and solutions Proceedings of the 21st International Conference on Artificial Neural Networks (ICANN), 2011.- 293–300 с.
19. Altmann A, Tolosi L, Sander O, Lengauer T. Permutation importance:a corrected feature importance measure. Bioinformatics, 2010.
20. Tolosi L, Lengauer T, Classification with correlated features: unreliability of feature ranking and solutions.. Bioinformatics. Volume 27, Issue 14, 15 July 2011, 1986-1994 с..
21. Machine Learning Benchmarks and Random Forest Regression. Center for Bioinformatics & Molecular Biostatistics [Електронний ресурс]: Режим доступу: https://escholarship.org/uc/item/35x3v9t4
22. Principal Component Analysis [Електронний ресурс]: Режим доступу: https://www.statistixl.com/features/principal-components/
23. Source code for pyod.models.pca [Електронний ресурс]: Режим доступу:

https://pyod.readthedocs.io/en/latest/_modules/pyod/models/pca.html

24. Abdi H., Williams L.J. (2010). Principal component analysis.. Wiley Interdisciplinary Reviews: Computational Statistics, 2: 433–459.
25. Korniyenko B., Yudin A., Galata L. Risk estimation of information system.Wschodnioeuropejskie Czasopismo Naukowe. 2016. № 5. P. 35 - 40.
26. Корнієнко Б.Я., Юдін О.К., Снігур О.С. Безпека аутентифікації у web-ресурсах. Захист інформації. 2012. № 1 (54). С. 20 -25. DOI: 10.18372/2410-7840.14.2056
27. Корнієнко Б.Я. Безпека інформаційно-комунікаційних систем та мереж. Навчальний посібник для студентів спеціальності 125 «Кібербезпека». – К.:НАУ, 2018. – 226 с.
28. Chen Th.M. Network traffic modeling. In: The Handbook of Computer Networks. Vol. 3. Wiley, 2007, pp. 326-339.
29. Ihler A., Hutchins J., Smyth P. Написання для визначення результатів з Марковим-модуляованими поісонними процесами. ACM Trans. Knowl. Discov. Data, 2007, vol. 1, no. 3, art. 13.DOI: https://doi.org/10.1145/1297332.1297337
30. Paxson V., Floyd S. Wide area traffic: failure of Poisson modeling. IEEE/ACM Trans. Netw., 1995, vol. 3, iss. 3, pp. 226-244. DOI: https://doi.org/10.1109/90.392383
31. Корнієнко Б.Я., Максімов Ю.О., Марутовська Н.М. Прикладні програми управління інформаційними ризиками. Захист інформації. 2012. № 4 (57). С. 60 – 64. DOI: 10.18372/2410-7840.14.3493 (ukr).
32. Воробйов О.В., Можаєв А.А., Осколков А.П. Аналіз моделей опису трафіку. Наука та техніка Повітряних Сил Збройних Сил України, 2014, № 2 (15), с. 170-172.
33. Heffes H., Lucantoni D. A Markov modulated characterization of packetized voice and data traffic and related staticical multiplexer performance. IEEE J. Sel. Areas Commun., 1986, vol. 4, iss. 6, pp. 856-868. DOI: https://doi.org/10.1109/JSAC.1986.1146393

34. Singh LN, Dattatreya GR A novel approach до параметра estimation в Markov-модуляції Пойсон процесів. Proc. IEEE (ETC), 2004, pp. 1–6.
35. Leland WE, MS Taqqu, Wilinger W., et al. Ha Self-Similar Nature of Ethernet Traffic (Extended version). IEEE/ACM Trans. Netw., 1994, vol. 2, iss. 1, pp. 1–15.DOI: https://doi.org/10.1109/90.282603
36. Adas A. Traffic models in broadband networks. IEEE Commun. Mag., 1997, vol. 35, iss. 7, pp. 82–89. DOI: https://doi.org/10.1109/35.601746
37. Едемська О.М., Бєльков Д.В. Дослідження мережевого трафіку за допомогою функції Херста. Інформатика та кібернетика, 2015, № 2, с. 39-46.

ДОДАТОК А

Технічне завдання

Міністерство освіти та науки України

Вінницький національний технічний університет

Факультет інформаційних технологій та комп'ютерної інженерії

Кафедра обчислювальної техніки

ЗАТВЕРДЖУЮ

Завідувач кафедри ОТ

_____проф., д.т.н. О. Д. Азаров

«___» _____ 20__ р.

ТЕХНІЧНЕ ЗАВДАННЯ

на виконання магістерської кваліфікаційної роботи

«Програмний засіб для моніторингу та контролю мережевого трафіку мобільних пристроїв»

08-54.КМКР.033.00.000 ПЗ

Науковий керівник

к.т.н., доц. каф. ОТ

_____ Муращенко О.Г.

виконав:

магістрант 2 курсу,

_____Любецький С.І.

Вінниця 2023

1 Підстава виконання магістерської кваліфікаційної роботи

1.1 Підвищення рівня автоматизації процесів обробки, зберігання та передачі даних також відображається на появі проблем безпеки, а витрати на відшкодування збитків, завданих злочинцями, постійно зростають, при цьому спостерігається постійна тенденція до збільшення кількості атак на комп'ютерні системи та мережі

Робота передбачає створення програмного забезпечення для аналізу внутрішнього трафіку, на основі якого можна вчасно вжити заходів і тим самим захистити пристрій і програми від впливу мережі.

1.2 Наказ про затвердження теми МКР.

2 Мета і призначенням МКР

2.1 Метою роботи є є вдосконалення системи збору мережного трафіку мобільних пристроїв та розширення її функціональних можливостей для аналізу та виявлення несанкціонованої активності.

2.2 Призначення розробки — виконання магістерської кваліфікаційної роботи.

3 Вихідні дані для виконання МКР

Вихідні дані для виконання МКР: бібліотека PCap, мови програмування C/C++, Java, .NET. Це бібліотека libpcap для Unix-подібних систем і WinPcap для Microsoft Windows.

4 Вимоги до виконання МКР

МКР повинна задовольняти таку вимогу —моніторити та контролювати мережевий трафік мобільних пристроїв, що дозволяє забезпечити прості та розширені способи блокування доступу до Інтернет.

5 Етапи МКР та очікувані результати

Етапи роботи та очікувані результати приведено в табл. А.1.

6 Матеріали, що подаються до захисту МКР

До захисту МКР подаються: пояснювальна записка МКР, ілюстративні та графічні матеріали, протокол попереднього захисту МКР на кафедрі, відзив наукового керівника, відзив опонента, протоколи складання державних екзаменів, анотації до МКР українською та іноземною мовами, довідка про відповідність оформлення МКР діючим вимогам.

Таблиця А.1 — Етапи МКР

з/п	Назва етапів виконання магістерської роботи	Строк виконання етапів роботи	.
	Постановка мети та задач роботи	21.10.23	
	Класифікація засобів моніторингу та контролю	25.10-30.10.23	
	Технології виявлення аномальної діяльності	31.10-08.11.23	
	Проектування системи	09.11-15.11.23	
	Розробка прототипу системи моніторингу мережного трафіку	16.11-.20.1123	
	Розробка прототипу системи контролю мережного трафіку	21.11-25.11.23	
	Обґрунтування вибору програмних засобів	26.11-31.11.23	
	Дослідження програмного засобу моніторингу та контролю мережного трафіку	01.12-04.12.23	
	Розрахунок економічної частини роботи	01.12-04.12.23	
	Оформлення пояснювальної записки та ілюстративного матеріалу	05.12.23	
	Аналіз виконання роботи, висновки, додатки		
	Перевірка якості виконання магістерської роботи та усунення недоліків		

7 Порядок контролю виконання та захисту МКР

Виконання етапів розрахункової та графічної 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

8.1 При оформлюванні МКР використовуються:

— ДСТУ 3008: 2015 «Звіти в сфері науки і техніки. Структура та правила оформлювання»;

— ДСТУ 8302: 2015 «Бібліографічні посилання. Загальні положення та правила складання»;

— Методичні вказівки до виконання магістерських кваліфікаційних робіт зі спеціальності 123 — «Комп’ютерна інженерія». Кафедра обчислювальної техніки ВНТУ 2022.

8.2 Порядок виконання МКР викладено в «Положення про кваліфікаційні роботи на другому (магістерському) рівні вищої освіти СУЯ ВНТУ–03.02.02 П.001.01:21.

ДОДАТОК Б

Існуючі технології СВВ

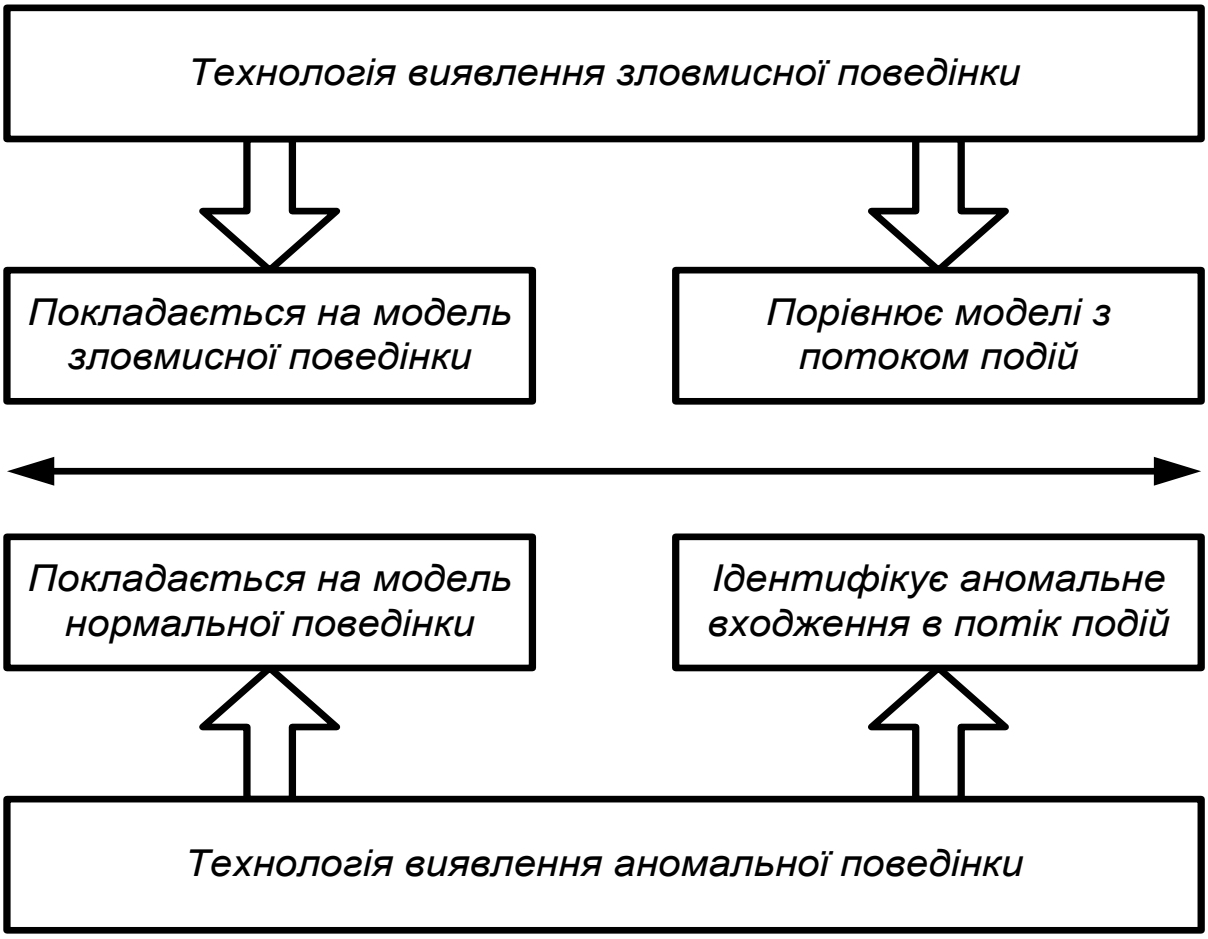

Рисунок Б.1 — Існуючі технології СВВ

ДОДАТОК В

Структурна схема системи аналізу мережного трафіку

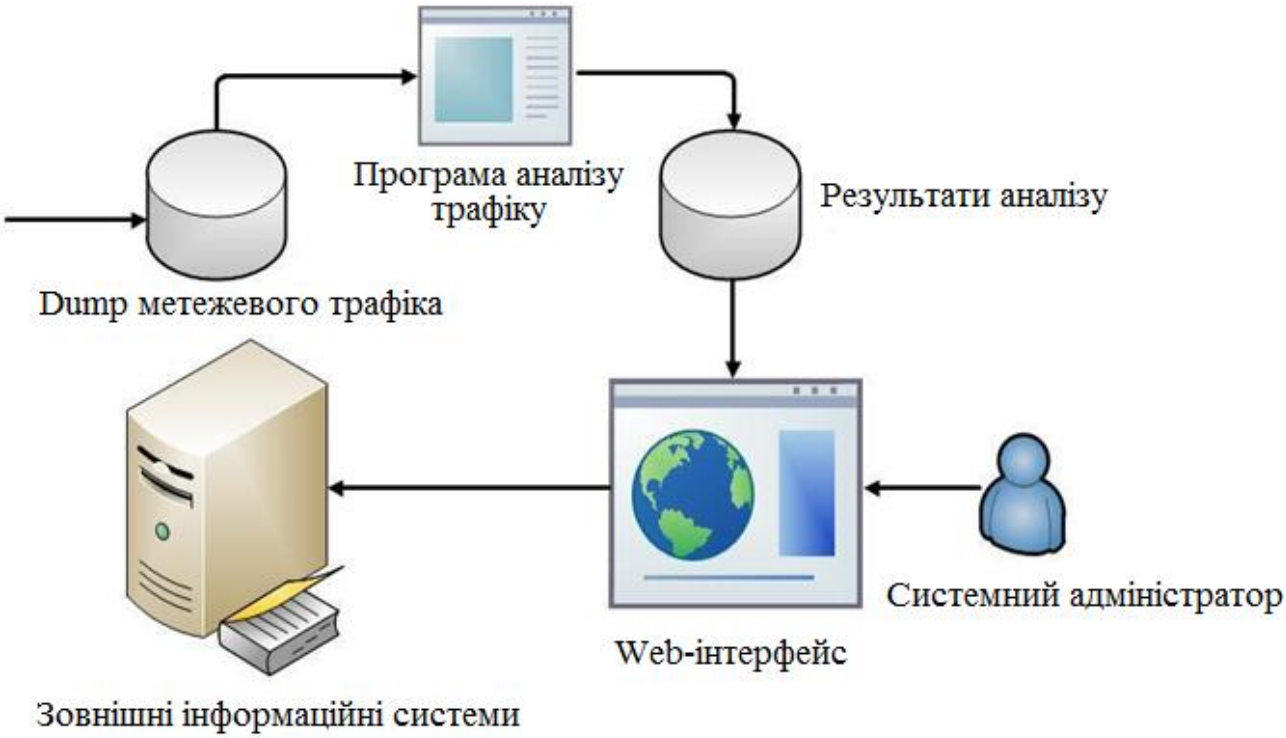

Рисунок В.1 — Структурна схема системи аналізу мережного трафіку

ДОДАТОК Г

Стек технологій MEAN

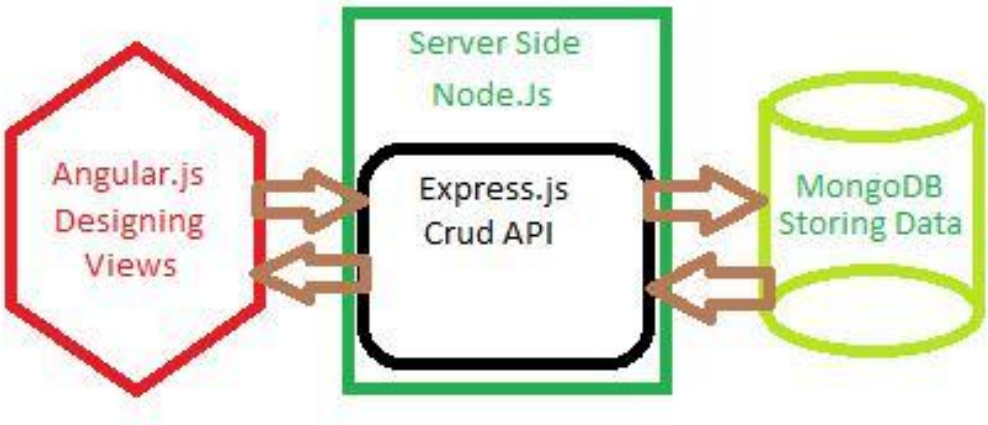

Рисунок Г.1 — Стек технологій MEAN

ДОДАТОК Д

Діаграма класів

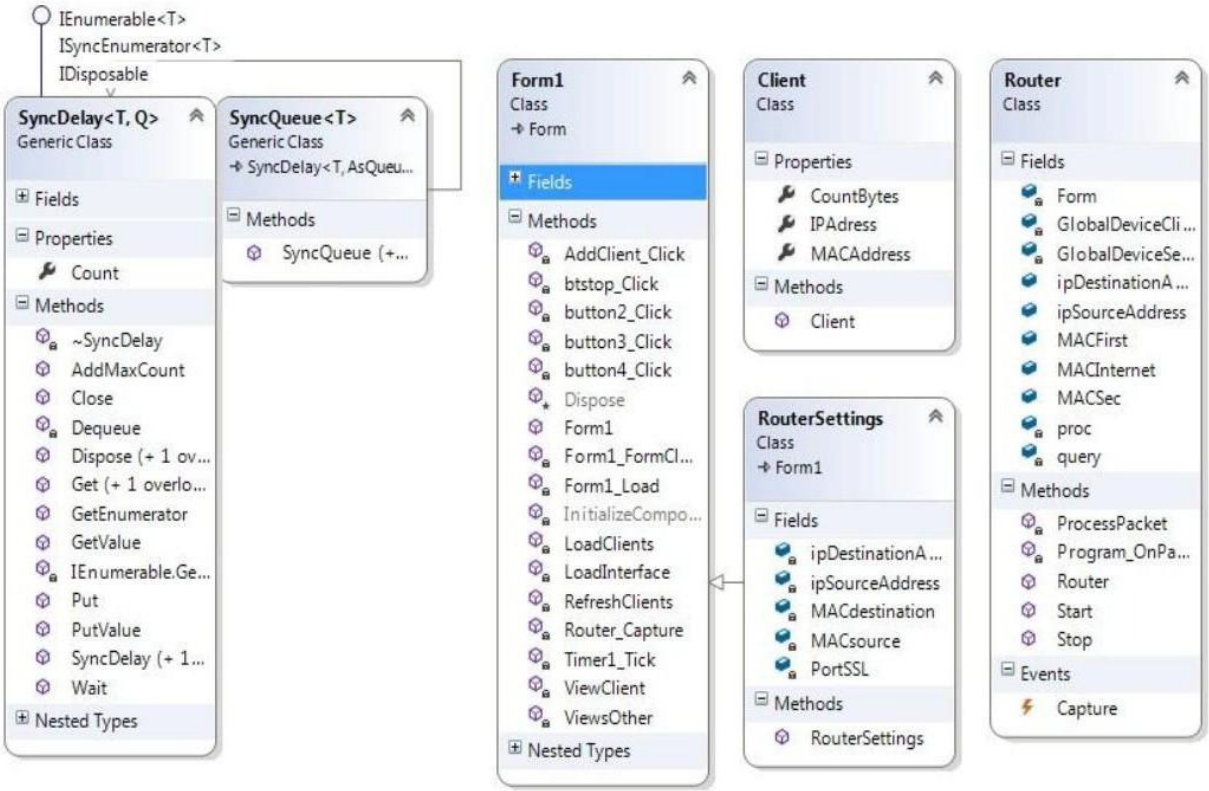

Рисунок Д.1 — Діаграма класів

ДОДАТОК Е
ПРОТОКОЛ
ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ
НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: Програмний засіб для моніторингу та контролю мережевого трафіку мобільних пристроїв
Тип роботи: магістерська кваліфікаційна робота
(БДР, МКР)
Підрозділ кафедра обчислювальної техніки
(кафедра, факультет)

Показники звіту подібності Unicheck
Оригінальність 87,6 % Схожість 12,4 %

Аналіз звіту подібності (відмітити потрібне):
✓ 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 роботи і відсутності самостійності її виконання автором. Роботу направити на розгляд експертної комісії кафедри.
□ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Особа, відповідальна за перевірку _____ Захарченко С.М.
(підпис) (прізвище, ініціали)

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

Автор роботи _____ Любецький С.І.
(підпис) (прізвище, ініціали)

Керівник роботи _____ Муращенко О.Г.
(підпис) (прізвище, ініціали)