Центральноукраїнський національний технічний університет
Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення

”Допущено до захисту”
Завідувач кафедри кібербезпеки
та програмного забезпечення
д.т.н., професор
_____ Олексій СМІРНОВ
“____” _____ 2025 р.

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА
за першим (бакалаврським) рівнем вищої освіти
на тему
“Програмне забезпечення системи перевірки та фільтрації пакетів у мережі на основі технології DPI”

Виконав здобувач вищої освіти
IV курсу, групи КІ-21-1
ОПП «Комп’ютерна інженерія»
спеціальності 123 «Комп’ютерна інженерія»
_____ Куліков О.С.
«____» _____ 2025 р.

Керівник проекту
доктор філософії (PhD)
_____ Усік П.С.
«____» _____ 2025 р.
Рецензент _____
_____

м. Кропивницький

Центральноукраїнський національний технічний університет

Факультет *Механіко-технологічний*

Кафедра *Кібербезпеки та програмного забезпечення*

Освітній ступінь *бакалавр*

Галузь знань ._12 _____*“*Інформаційні технології*”*_____

Спеціальність *123 “*Комп’ютерна інженерія*”*

Освітньо-професійна (освітньо-наукова) програма _*“*Комп’ютерна інженерія*”*_

ЗАТВЕРДЖУЮ
Завідувач кафедри
д.т.н., проф.
_____ Олексій СМІРНОВ
« 17 » січня 2025 року

ЗАВДАННЯ НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ ЗА ПЕРШИМ (БАКАЛАВРСЬКИМ) РІВНЕМ ВИЩОЇ ОСВІТИ ЗДОБУВАЧА ВИЩОЇ ОСВІТИ

Кулікову Олександру Сергійовичу
(прізвище, ім’я, по батькові)

1. Тема роботи *Програмне забезпечення системи перевірки та фільтрації пакетів у мережі на основі технології DPI*

2. Керівник роботи *Усік Павло Сергійович, доктор філософії (PhD)*
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу № 46-02 від 17.01.2025 року

3. Строк подання студентом роботи до захисту *23.05.2025 р.*

4. Мета та завдання випускної кваліфікаційної роботи*: Метою роботи є розробка програмного забезпечення системи перевірки та фільтрації пакетів у мережі на основі технології DPI*

5. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Призначення та область використання.

2.Перегляд аналогічних існуючих систем.

3. Опис і обґрунтування проектних рішень.

4. Етапи програмування системи.

5. Впровадження системи в промислову експлуатацію.

6. Висновки

6. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)

Структурна схема системи	*1 аркуш*
Функціональна схема системи	*1 аркуш*
Діаграма процесів	*1 аркуш*
Блок-схема алгоритму роботи додатку	*2 аркуша*

7. Дата видачі завдання « 17 » січня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти	Строк виконання етапів випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти	Примітка
1.	Аналіз існуючих систем	10.03.2025 р.	
2.	Постановка задачі, оформлення ТЗ	15.03.2025 р.	
3.	Розробка моделі компонента	20.03.2025 р.	
4.	Розробка структур даних	25.03.2025 р.	
5.	Розробка алгоритмів зв’язку та відображення	30.03.2025 р.	
6.	Програмування алгоритмів	10.04.2025 р.	
7.	Оформлення ПЗ	17.04.2025 р.	
8.	Попередній захист роботи	23.05.2025 р.	

Дата видачі завдання
« 17 » січня 2025 р.

Підпис керівника
_____ Усік П.С.
(прізвище та ініціали)

Завдання прийнято до виконання
« 17 » січня 2025 р.

Підпис здобувача
_____ Куліков О.С.
(прізвище та ініціали)

АНОТАЦІЯ

Куліков О.С. Програмне забезпечення системи перевірки та фільтрації пакетів у мережі на основі технології DPI. 123 Комп'ютерна інженерія. Центральноукраїнський національний технічний університет. Кропивницький. 2025.

В даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи перевірки та фільтрації пакетів у мережі на основі технології DPI.

Метою розробки є програмне забезпечення системи перевірки та фільтрації пакетів у мережі на основі технології DPI.

Результат роботи – програмна реалізація системи перевірки та фільтрації пакетів у мережі на основі технології DPI.

В процесі роботи над програмною моделлю виконано аналіз існуючих апаратних та програмних засобів. В повній мірі описані всі компоненти розробленого програмного забезпечення.

Розроблено зручний інтерфейс користувача. Наведені інструкції по роботі з програмними засобами.

Програма може використовуватися на ПЕОМ з ОС Windows 10/11.

Програму розроблено в середовищі Python.

Ключові слова: комп'ютерна інженерія, перевірка та фільтрація пакетів, DPI

ABSTRACT

Kulikov O.S. Software for a network packet inspection and filtering system based on DPI technology. 123 Computer Engineering. Central Ukrainian National Technical University. Kropyvnytskyi. 2025.

In this final qualification work for the first (bachelor's) level of higher education, software has been developed that is intended for a network packet inspection and filtering system based on DPI technology.

The purpose of the development is software for a network packet inspection and filtering system based on DPI technology.

The result of the work is a software implementation of a network packet inspection and filtering system based on DPI technology.

In the process of working on the software model, an analysis of existing hardware and software was performed. All components of the developed software are fully described.

A convenient user interface has been developed. Instructions for working with software are provided.

The program can be used on a PC with OS Windows 10/11.

The program was developed in the Python environment.

Keywords: computer engineering, packet inspection and filtering, DPI

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

ЕОМ	–	електрона обчислювальна машина
КВ	–	коефіцієнт варіації
КЗ	–	канал зв'язку
НСД	–	несанкціонований доступ
ПС	–	програмна середа
СВВ	–	система виявлення вторгнень
СеМО	–	експонентна мережа масового обслуговування
СМО	–	система масового обслуговування
СПД	–	система передачі даних

ВСТУП

Актуальність теми. Deep Packet Inspection (DPI, також complete packet inspection і Information eXtraction або IX) – технологія накопичення статистичних даних, перевірки й фільтрації мережних пакетів по їхньому вмісту. Глибока перевірка пакетів (DPI) відноситься до типу аналізу пакетів, який виходить за рамки інформації заголовка пакета та також аналізує корисне навантаження пакета. DPI можна використовувати для визначення надмірних рівнів неділового трафіку на підприємствах, таких як використання соціальних мереж, які потрібно фільтрувати або блокувати, для виявлення потоків даних, відеотрафіку, зашифрованого трафіку BitTorrent, зловмисної поведінки, зловмисного трафіку, вторгнення тощо, щоб виявити хости за NAT операторського рівня шляхом вилучення немаршрутизованих IP-адрес зі списків однорангових вузлів, отриманих шляхом сканування розподіленої хеш-таблиці BitTorrent (DHT), для класифікації зловмисного програмного забезпечення, щоб проаналізувати трафік honeypot, щоб полегшити моніторинг безпеки на основі криміналістики, а також увімкнути криміналістику за проектом промислових систем. Насправді глибока перевірка пакетів може виявити та записати онлайн-активність до такої міри, що це викликає занепокоєння щодо конфіденційності щодо масового стеження державними та урядовими установами (зокрема, відповідно до законодавства, яке вимагає «зручних для прослуховування» онлайн-сервісів, таких як CALEA у США), навіть якщо величезний обсяг трафіку робить непрактичним записувати всі сліди активності користувачів. З іншого боку, глибока перевірка пакетів дозволяє мережевим операторам формувати трафік і контролювати різні типи трафіку, такі як електронна пошта, VoIP і P2P. Такі компанії, як NETSCOUT і Sandvine надають послуги DPI для визначення пріоритетів мережевої активності, забезпечення дотримання політик і допомоги в розробці нових планів обслуговування.

Мета й завдання дослідження. Метою роботи є програмне забезпечення системи перевірки та фільтрації пакетів у мережі на основі технології DPI.

Для досягнення поставленої мети визначена програма дослідження, що складається з наступних завдань:

– Огляд існуючих систем перевірки та фільтрації пакетів у мережі на основі технології DPI.

– Дослідження системи перевірки та фільтрації пакетів у мережі на основі технології DPI.

– Програмна реалізація системи перевірки та фільтрації пакетів у мережі на основі технології DPI.

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми дозволяють успішно вирішувати задачі перевірки та фільтрації пакетів у мережі на основі технології DPI.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи перевірки та фільтрації пакетів у мережі на основі технології DPI, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти.

1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ

1.1 Призначення системи

Щоб усунути обмеження фільтрації пакетів, проксі-серверів програм і перевірки стану, була розроблена (або продана) технологія, відома як Deep Packet Inspection (DPI). DPI аналізує весь пакет і може буферизувати, збирати та перевіряти кілька пов’язаних пакетів як частину сеансу. DPI працює на рівнях L3-L7 стеку OSI.

Механізми DPI аналізують увесь IP-пакет і приймають рішення про пересилання за допомогою логіки на основі правил, що базується на відповідності підпису або регулярного виразу. Тобто вони порівнюють дані в пакеті з базою даних попередньо визначених сигнатур атаки (рядок байтів). Крім того, статистичні або історичні алгоритми можуть доповнювати зіставлення статичного шаблону. Проблема з DPI полягає в тому, що вміст пакетних даних є практично неструктурованим порівняно з високоструктурованими заголовками пакетів (додаткову інформацію див. у попередньому розділі про NAT). Аналіз заголовків пакетів можна зробити економічно, оскільки розташування полів заголовків пакетів обмежено стандартами протоколу. Однак вміст корисного навантаження здебільшого не має обмежень. Пошук у корисному навантаженні для кількох шаблонів рядків у потоці даних є обчислювально дорогим завданням. А зі збільшенням швидкості дроту вимогате, що ці пошуки виконуються на швидкості дроту, збільшує вартість. Крім того, оскільки база даних сигнатур загроз є динамічною, її потрібно легко оновлювати – це виключає використання звичайних ASIC. Перспективні підходи до вирішення цих проблем включають підхід на основі програмного забезпечення (Snort реалізує алгоритм Боєра-Мура) і підхід на основі апаратного забезпечення (ПЛІС, що працюють з алгоритмом фільтра Блума).

1.2 Область застосування

Ідентифікація BitTorrent

Клієнти BitTorrent з'єднуються із трекером за протоколом TCP. Для того, щоб виявити серед усього трафіку TCP такі пакети, досить перевірити, що вміст даних TCP пакета із другого байта збігається з «BitTorrent protocol» [7]

Subscriber Management

Важливим моментом є те, що правила, на підставі яких виконується шейпинг/блокування, можуть бути задані за допомогою двох основних базисів – per-service або per-subscriber. У першому випадку найпростішим образом обмовляється, що конкретному додатку дозволяється утилізувати певну смугу. У другому прив'язка додатка до смуги здійснюється для кожного передплатника або групи передплатників незалежно від інших, що виробляється через інтеграцію DPI з існуючими OSS/BSS системами оператора. Тобто можна настроїти систему таким чином, що передплатник, що за тиждень накачав торрентів на 100 гігабайт, до кінця місяця буде обмежений по швидкості закачування цих же торрентів на рівні 70% від купленого їм тарифу. А в передплатника, що купив додаткову послугу за назвою «Skype без проблем», трафік додатка Skype не буде блокуватися ні при яких умовах, але будь-який іншою – легко. Можна зробити прив'язку до User-Agent і дозволити браузинг тільки за допомогою браузерів, що рекомендуються, можна робити хитрі редиректи залежно від типу браузера або ОС. Іншими словами, гнучкість тарифних планів і опцій обмежена лише здоровим глуздом. Якщо ж мова йде про трафік мобільних операторів, то DPI дозволяє контролювати завантаження кожної базової станції окремо, справедливо розподіляючи ресурси таким чином, щоб всі користувачі залишилися задоволені якістю сервісу. Більшість виробників пакетного ядра EPC (Evolved Packet Core) для LTE інтегрує у свій PDN-GW функціонал DPI, пристосований для рішення завдань мобільних операторів.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи перевірки та фільтрації пакетів у мережі на основі технології DPI, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти.

2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ

2.1 Огляд існуючих систем, технологій, архітектур, програмних рішень за профілем теми випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти

У цьому огляді зупинимося на трьох популярних сніфферах належного рівня: SearchInform, TamoSoft NetResident і pTraffer.

SearchInform

SearchInform, мабуть, можна назвати одним із самих знаменитих сніфферів. Користувачами SearchInform є багато великих компаній, чия діяльність пов'язана з найрізноманітнішими сферами життя – від кораблебудування й банківської справи до виробництва побутової техніки й інших агрегатів.

Не можна заперечувати той факт, що SearchInform по праву є настільки популярною програмою, адже всі обов'язки, які на неї покладають, вона виконує чудово. І перед багатьма іншими сніфферами вона має масу переваг, серед яких, за словами розроблювачів, головними є наступні:

– легкість в інсталяції й експлуатації, що дозволяє працювати із програмою без якого-небудь співробітника компанії-розроблювача ПЗ;

– установка ПЗ не впливає на існуюче функціонування мережі;

– можливість простежувати всі канали витоку інформації, а також виконувати вибір лише деяких модулів;

– контроль над Skype;

– суміжність із операційною системою Windows;

– абсолютну інтеграцію з будь-якою доменною структурою;

– можливість налаштування системи оповіщення без залучення сторонніх програмістів і ефективний захист даних при мінімальних витратах;

– дозволяє розмежувати права доступу до перехопленої інформації;

– можливість простежувати місця, де була замічена поява конфіденційної інформації;

– можливість відстежити активність певного користувача й перегляду інформації, його що цікавить;

– збереження історії операцій, коли-або проведених у даній мережі.

Всі ці плюси дійсно мають місце бути, але є один великий мінус, що, на жаль, дуже часто перекреслює їх всі, роблячи їх непомітними в тіні жирної негативної оцінки – SearchInform програма далеко не дешева.

Правда, кожний користувач може одержати тріал-версію на 30 днів, але потім йому однаково прийде розщедритися, і обійдеться йому це в кругленьку суму. Тому для багатьох фірм, що не мають величезного бюджету, використання SearchInform стає нереальною мрією.

Другим помітним мінусом для SearchInform стала необхідність установки віддалених агентів на ПК користувачів, що певною мірою дозволяє відстежити факт спостереження й визначити ПЗ яких воно здійснюється.

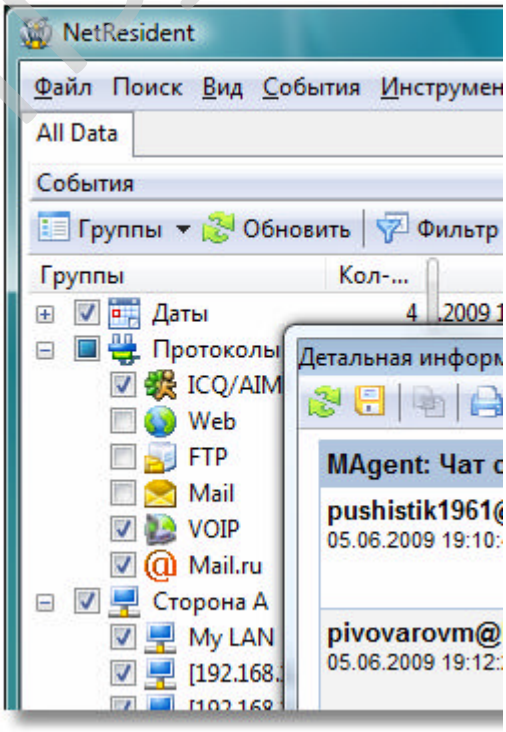


Рисунок 2.1 – Інтерфейс користувача SearchInform

TamoSoft NetResident

Втім, це не так страшно, тому що SearchInform – не єдине подібне рішення проблем, і TamoSoft NetResident також популярний серед українських підприємців. TamoSoft NetResident – це один із провідних лідерів на ринку перехоплення інформації в комп'ютерних мережах і мережних аналізаторах, що, безумовно, роблять йому велику честь. Особливо якщо врахувати, що програмне забезпечення таки незручне, і вимагає чималого розуму від користувача. Звичайно ж, подібний комплекс навряд чи буде використовувати людина з базовими знаннями системного адміністрування, однак легкість у використанні й експлуатації ще нікому не заважала. Так, багато хто із плюсів, які має SearchInform, у TamoSoft NetResident відсутні – не всі, але багато хто. Наприклад можливість індексування даних, щоб скажемо по відправнику листа знайти де він ще фігурував і з ким спілкувався можливим не представляється – у нет-резидента своя власна БД зі своїм форматом... Робота із цією програмою можлива тільки при наявності в штаті мізкуватого програміста, сісадміна, що і буде виконувати за вас всю роботу й пояснить вам усе, що буде потрібно, а інакше впоратися з TamoSoft NetResidentом не вийде ніяк. Зате на вибір пропонується відразу безліч продуктів на будь-який смак, які підійдуть у тім або іншому випадку. Який з них варто вибрати, вибір залежить тільки від того, що потрібно у вашім випадку, і тому потрібно особливо уважно стежити за описом кожного.

CommView

Глибокий аналіз інформації в мережі може бути проведений за допомогою таких інструментів, як CommView і CommViewforWiFi, які відповідно підійдуть для провідних і бездротових мереж. Віддалені агенти RA цих програм здатні проводити аналіз віддалені мережі – і для домашніх ПК існує полегшена версія CommViewHome. Знаходиться вона, до речі, теж, як і важить – менше звичайний базової версії.

Продукт NetResident дозволяє відслідковувати події, що відбуваються з усіма вашими файлами, папками й іншими даними, розміщеними в мережі. Якщо

буде потреба можна навіть прослухати розмови по телефоні, якщо вони проводилися через вашу мережу.

Для обліку трафіку на офісних або домашніх комп'ютерах придасться CommTraffic, напевно, кращий із всіх пропонованих цією компанією софт. Унікальна технологія перехоплення інформації в мережі дозволить визначити відправлений і отриманий трафік до байта, а особливі налаштування допоможуть розібратися в ній навіть новачкові, на відміну від багатьох інших програм TamoSoft NetResidenta.

pTraffer

Але все-таки кращим рішенням, на наш погляд стане, програмне забезпечення pTraffer, що і завоювало за зовсім невеликий проміжок часу величезну популярність серед користувачів різних мереж. Відрізняючись легкістю у використанні, pTraffer стане зручним інструментом, як для фахівців-знавців, так і новачків, яким нелегко управляти складним ПЗ на комп'ютері. Система захисту інформації, що стане для вас надійним помічником – це саме ptraffer.

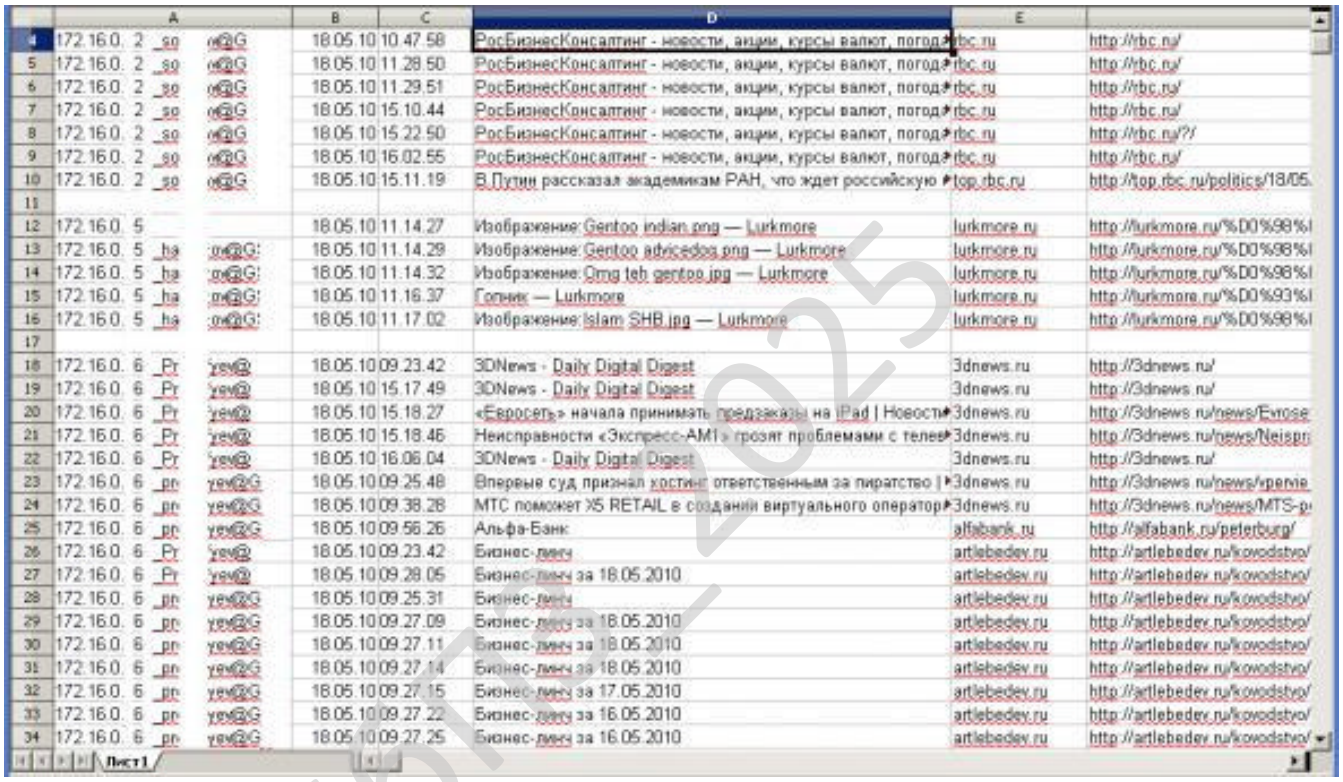
За допомогою pTraffer ви можете контролювати кожний потік інформації, як прийнятої, так і вихідної, від кожного користувача окремо й усього сегмента мережі в цілому.

За допомогою цього продукту можлива робота з файлами дампа пам'яті й у режимі он-лайн, що стало можливим завдяки драйверу WinPcap. Користувачі pTraffer також можуть у будь-яке зручне для себе час відновити переписку з кожним абонентом, а також використовувати кожної з доступних зараз безкоштовних індексаторів даних (Персональний яндекс, Google Desktop і т.п.) на свій смак, що звичайно виставляє продукт у вигідному кольорі.

У цьому програмному забезпеченні, на відміну від всіх інших, є можливість самостійної вказівки полів, наприклад пошукових машин або повідомлень на форуми. В інтернеті можна знайти ролик у якому показаний приклад використання цього інструментарію. Ми перевірили – дійсно можна

самостійно розширювати можливості функціонала системи. Раніше подібного підходу, простого й логічного, ми не зустрічали, тому очевидно це унікальний плюс для pTraffer.

Безсумнівним плюсом у скарбничку pTraffer'a буде система щоденного контролю всіх що зачіпаються web-ресурсів. Контроль може бути ручним і автоматичним, залежно від ваших умінь. По факті це виглядає от так:



	A	B	C	D	E	
4	172.16.0. 2 _so o@G	18.05.10	10.47.58	РосБизнесКонсалтинг - новости, акции, курсы валют, погод	rbc.ru	http://rbc.ru/
5	172.16.0. 2 _so o@G	18.05.10	11.28.50	РосБизнесКонсалтинг - новости, акции, курсы валют, погод	rbc.ru	http://rbc.ru/
6	172.16.0. 2 _so o@G	18.05.10	11.29.51	РосБизнесКонсалтинг - новости, акции, курсы валют, погод	rbc.ru	http://rbc.ru/
7	172.16.0. 2 _so o@G	18.05.10	15.10.44	РосБизнесКонсалтинг - новости, акции, курсы валют, погод	rbc.ru	http://rbc.ru/
8	172.16.0. 2 _so o@G	18.05.10	15.22.50	РосБизнесКонсалтинг - новости, акции, курсы валют, погод	rbc.ru	http://rbc.ru/?/
9	172.16.0. 2 _so o@G	18.05.10	16.02.55	РосБизнесКонсалтинг - новости, акции, курсы валют, погод	rbc.ru	http://rbc.ru/
10	172.16.0. 2 _so o@G	18.05.10	15.11.19	В Путин рассказал академикам РАН, что ждет российскую	top.rbc.ru	http://top.rbc.ru/politics/18/05
11						
12	172.16.0. 5	18.05.10	11.14.27	Изображение:Gentoo indian.png — Lurkmore	lurkmore.ru	http://lurkmore.ru/%D0%98%I
13	172.16.0. 5 _ha o@G	18.05.10	11.14.29	Изображение:Gentoo advicedog.png — Lurkmore	lurkmore.ru	http://lurkmore.ru/%D0%98%I
14	172.16.0. 5 _ha o@G	18.05.10	11.14.32	Изображение:Omg teh gentoo.jpg — Lurkmore	lurkmore.ru	http://lurkmore.ru/%D0%98%I
15	172.16.0. 5 _ha o@G	18.05.10	11.16.37	Гопник — Lurkmore	lurkmore.ru	http://lurkmore.ru/%D0%93%I
16	172.16.0. 5 _ha o@G	18.05.10	11.17.02	Изображение:Islam SHB.jpg — Lurkmore	lurkmore.ru	http://lurkmore.ru/%D0%98%I
17						
18	172.16.0. 6 _Pr yev@	18.05.10	09.23.42	3DNews - Daily Digital Digest	3dnews.ru	http://3dnews.ru/
19	172.16.0. 6 _Pr yev@	18.05.10	15.17.49	3DNews - Daily Digital Digest	3dnews.ru	http://3dnews.ru/
20	172.16.0. 6 _Pr yev@	18.05.10	15.18.27	«Евросеть» начала принимать предзаказы на iPad | Новости	3dnews.ru	http://3dnews.ru/news/Evrose
21	172.16.0. 6 _Pr yev@	18.05.10	15.18.46	Неисправности «Экспресс-АМ1» грозят проблемами с телев	3dnews.ru	http://3dnews.ru/news/Neispr
22	172.16.0. 6 _Pr yev@	18.05.10	16.06.04	3DNews - Daily Digital Digest	3dnews.ru	http://3dnews.ru/
23	172.16.0. 6 _pr yev@G	18.05.10	09.25.48	Впервые суд признал хостинг ответственным за пиратство |	3dnews.ru	http://3dnews.ru/news/vpervie
24	172.16.0. 6 _pr yev@G	18.05.10	09.38.28	МТС поможет X5 RETAIL в создании виртуального оператор	3dnews.ru	http://3dnews.ru/news/MTS-p
25	172.16.0. 6 _pr yev@G	18.05.10	09.56.26	Альфа-Банк	alfabank.ru	http://alfabank.ru/peterburg/
26	172.16.0. 6 _Pr yev@	18.05.10	09.23.42	Бизнес-линч	artlebedev.ru	http://artlebedev.ru/kovodstvo/
27	172.16.0. 6 _Pr yev@	18.05.10	09.28.05	Бизнес-линч за 18.05.2010	artlebedev.ru	http://artlebedev.ru/kovodstvo/
28	172.16.0. 6 _pr yev@G	18.05.10	09.25.31	Бизнес-линч	artlebedev.ru	http://artlebedev.ru/kovodstvo/
29	172.16.0. 6 _pr yev@G	18.05.10	09.27.09	Бизнес-линч за 18.05.2010	artlebedev.ru	http://artlebedev.ru/kovodstvo/
30	172.16.0. 6 _pr yev@G	18.05.10	09.27.11	Бизнес-линч за 18.05.2010	artlebedev.ru	http://artlebedev.ru/kovodstvo/
31	172.16.0. 6 _pr yev@G	18.05.10	09.27.14	Бизнес-линч за 18.05.2010	artlebedev.ru	http://artlebedev.ru/kovodstvo/
32	172.16.0. 6 _pr yev@G	18.05.10	09.27.15	Бизнес-линч за 17.05.2010	artlebedev.ru	http://artlebedev.ru/kovodstvo/
33	172.16.0. 6 _pr yev@G	18.05.10	09.27.22	Бизнес-линч за 16.05.2010	artlebedev.ru	http://artlebedev.ru/kovodstvo/
34	172.16.0. 6 _pr yev@G	18.05.10	09.27.25	Бизнес-линч за 16.05.2010	artlebedev.ru	http://artlebedev.ru/kovodstvo/

Лист1

Рисунок 2.2 – Інтерфейс користувача pTraffer

По кожному користувачі видно які саме сторінки він відвідував, його IP, логsн і інші необхідні дані. Важливим зауваженням є той факт, що аналіз іде незалежно від використання проксs-сервера і його портів (перевіряються всі порти).

Також програма pTraffer «уміє» обчислювати тих користувачів вашої компанії, що займаються замість роботи, закачуванням різних торрентів, при використанні Gmail, VNC, Google Talk, Jabber і інших Інтернет-сервісів. Це

можливо завдяки сигнатурному аналізу сесій, що працює повністю в автоматичному режимі.

Зручний інтерфейс і розширений пошуковий функціонал (як убудований, так і зовнішніх індексаторів) дозволяє проводити пошук у найкоротший термін по всім збереженим раніше даним за кілька років, при проведенні розслідувань.

Для кожної служби організації існує окремий самостійний звіт. Так, наприклад, служба безпеки одержує:

– діалоги по певному наборі ключових слів;

– загальний список повідомлень на всілякі форуми, сайти, пошукові системи;

– вихідний і вхідний трафік у мегабайтах;

– пошукові запити окремих співробітників компанії.

Або, скажемо, служба IT:

– звіти по біллінгу всього трафіку в цілому, або розбитого по портах або протоколам;

– звіти по співробітниках: хто і які мережні сервіси використовує по факту (Jabber, VNC, RDP, IRC інші);

– звіти по Інтернет-Сайтам (незалежно від проксі-сервера й портів).

pTraffer розроблений таким чином, що ви можете самостійно змінювати функціонала вашої копії системи, вносячи туди певні корективи й наробітки. Також на форумі системи викладено великий набір стандартних opensource-скрипів по розширенню функціонала.

Загальне враження від траффера створилося таке, що він цілком може виконувати роль частини системи глобального ешелонування населення, у всякому разі частина розробок і ідей продукту можуть брати початок із системи СОРМ2 або її аналогів.

2.2 Обґрунтування вибору засобів для побудови системи та мови програмування

Python – динамічна інтерпретована об'єктно-орієнтована скриптова мова програмування із строгою динамічною типізацією. Офіційний сайт мови програмування Python https://www.python.org/. Python – багатоцільова мова програмування, яка дозволяє писати код, що добре читається. Відносний лаконізм мови Python дозволяє створити програму, яка буде набагато коротше свого аналога, написаного на іншій мові. Python – багатоплатформова мова програмування. Це означає, що програми на Python можна запускати в різних операційних системах без будь-яких змін.

Ще однією перевагою Python є його стандартна бібліотека, яка встановлюється разом з Python і містить готові інструменти для роботи з операційною системою, веб-сторінками, базами даних, різними форматами даних, для побудови графічного інтерфейсу програм тощо. Програми, написані на мові програмування Python, можуть бути як невеликими скриптами, так і складними системами. Python абсолютно безкоштовний.

Швидкість виконання коду Python

Один з можливих недоліків Python – швидкість виконання коду. Python не є компільованою мовою. Код на Python спочатку компілюється у внутрішній байт-код, який потім виконується інтерпретатором Python. У більшості випадків при використанні Python виходять програми повільніші в порівнянні з такими мовами, як C.

Втім, сучасні комп'ютери мають таку обчислювальну потужність, що для більшості застосунків швидкість розробки важливіша швидкості виконання, а програми на Python зазвичай пишуться набагато швидше.

Окрім того, Python легко розширюється модулями, написаними на C або C++. Такі модулі можуть використовуватися для виконання частин програми, що створюють інтенсивне навантаження на процесор.

Використання Python

Python використовується для різних цілей: для створення ігор і веб-застосунків, розробки внутрішніх інструментів для різноманітних проектів. Мова також широко застосовується в науковій області для досліджень і розв'язування прикладних завдань.

Застосування мови програмування Python:

1. BitTorrent – протокол для обміну даними.

2. Ubuntu Software Center – вільне програмне забезпечення для пошуку, установки і видалення пакунків в системі Ubuntu Linux.

3. Blender – програма для створення тривимірної комп'ютерної графіки, що включає засоби моделювання, анімації, вимальовування, пост-обробки відео, а також створення відеоігор.

4. GIMP – растровий графічний редактор, із підтримкою векторної графіки.

5. World of Tanks.

6. Вільна енциклопедія Вікіпедія.

7. Пошукова система Google.

8. DropBox – файловий хостинг, що включає персональне хмарне сховище, синхронізацію файлів і програму-клієнт.

9. YouTube – популярне відеосховище.

Версії Python

Мови програмування з часом змінюються – розробники додають в них нові можливості, а також виправляють помилки. Так з'являються різні версії мови. Наприклад, код написаний на Python 2 у більшості випадків не буде працювати у версії Python 3 без внесення додаткових змін.

Процесор є найважливішим компонентом в комп'ютері. Одна з основних функцій процесора – це обробка даних згідно комп'ютерної програми, яка є списком інструкцій, шляхом виконання арифметичних і логічних операцій над фрагментами даних.

Кожна інструкція в програмі – це команда, яка «повідомляє» процесору, яку операцію він повинен виконати. Процесор комп'ютера може розуміти лише ті інструкції, які написані на машинній мові. Машинна мова – це штучна мова, створена для передачі команд комп'ютеру. За допомогою машинної мови створюються ефективні програми, оскільки розробник отримує доступ до всіх можливостей процесора. Машинна мова – мова низького рівня.

Інструкція машинної мови існує для кожної операції, яку процесор здатний виконати – є інструкція для додавання чисел, є інструкція для віднімання чисел і т.д. Увесь набір інструкцій, який центральний процесор може виконати, відомий як набір інструкцій процесора.

Наприклад, у вас є певна програма, яка зберігається на диску вашого комп'ютера. Для виконання програми, ви здійснюєте подвійний клік на значку програми. Це змушує програму копіюватися з диска в оперативну пам'ять, після чого процесор комп'ютера виконує копію програми, яка знаходиться в оперативній пам'яті.

Коли процесор виконує інструкції програми, він бере участь у процесі, який є відомим як цикл fetch – decode – execute (отримати – декодувати – виконати). Цей цикл виконується для кожної інструкції у програмі і складається з трьох кроків:

Отримати

Програма – це послідовність інструкцій на машинній мові. Першим кроком циклу є завантаження (отримання) наступної інструкції з пам'яті в процесор.

Декодувати

Інструкція машинної мови – це двійкове число, яке представляє команду, що повідомляє процесору виконати певну операцію. На цьому кроці процесор декодує інструкцію, яку було «витягнуто» з пам'яті, для визначення того, яка операція повинна виконуватись.

Виконати

Останній крок циклу – виконати операцію.

Хоча процесор комп'ютера розуміє тільки машинну мову, людині непрактично писати програми на машинній мові. Така програма може мати тисячі або навіть мільйони бінарних інструкцій, і написання такої програми буде дуже обтяжливим процесом.

З цієї причини була створена мова асемблера як альтернатива машинній мові. Замість використання двійкових чисел для написання інструкцій, мова асемблера використовує короткі слова, відомі як мнемокоди.

Незважаючи на те, що мова асемблера не вимагає двійкових інструкцій, як у випадку машинної мови, проте вона вимагає високих знань про процесор. Використовуючи мову асемблера, навіть для найпростішої програми, необхідно написати велику кількість інструкцій.

Мова програмування високого рівня дозволяє створювати складні програми, не знаючи, як працює процесор, і не записуючи великої кількості інструкцій низького рівня. Крім того, більшість мов програмування високого рівня використовують слова, які легко зрозуміти.

Python – одна із популярних сучасних мов програмування високого рівня. Python – інтерпретована мова програмування. Python – це високорівнева інтерпретована мова програмування, на відміну від C++, яка є прикладом компільованої мови програмування. Назва Python відноситься як до мови програмування, так і до інтерпретатора – комп'ютерної програми, яка зчитує початковий код (написаний на Python) і виконує інструкції (команди).

Для перекладу мови високого рівня на машинну мову доступні два типи програм:

1. Компілятор.

2. Інтерпретатор.

Завантаження Python

Версії інтерпретатора Python для різних операційних систем доступні для безкоштовного завантаження за адресою https://www.python.org/downloads.

Середовище програмування для Python

Для написання програм використовують текстові редактори або інтегровані середовища розробки, які включають в себе різні інструменти для роботи з кодом: засіб для написання коду (текстовий редактор), інтерактивний інтерпретатор, відлагоджувач тощо.

Текстові редактори та інтегровані середовища програмування для Python:

– IDLE – стандартний редактор Python. Встановлюється разом з Python для користувачів Windows, окремим пакунком для користувачів Linux.

– Notepad++ – безкоштовний текстовий редактор початкового коду, який підтримує велику кількість мов, в тому числі і Python. Лише для користувачів Windows.

– Visual Studio Code – це легкий, але потужний редактор початкового коду, який розповсюджується безкоштовно і доступний у версіях для платформ Linux, Windows і macOS.

– PyScripter – інтегроване середовище розробки для мови програмування Python. Для користувачів Windows. Поширюється безкоштовно.

– Wing IDE 101 – вільне інтегроване середовище для Python, розроблене для навчання програмістів-початківців. Для користувачів Linux, Windows і macOS. Поширюється безкоштовно.

– Geany – вільний текстовий редактор з базовими елементами інтегрованого середовища розробки, доступний для операційних систем Linux, Windows і macOS.

– PyCharm – інтегроване середовище розробки для мови програмування Python. PyCharm є власницьким програмним забезпеченням. Наявна безкоштовна версія Community з усіченим набором можливостей. Для користувачів Linux, Windows і macOS.

– Thonny – IDE для вивчення програмування мовою Python. Для користувачів Linux, Windows і macOS.

– Mu – редактор коду Python для програмістів-початківців. Для користувачів Linux, Windows і macOS.

2.3 Розгорнута постановка завдання

Згідно з технічним завданням на випускну кваліфікаційну роботу за першим (бакалаврським) рівнем вищої освіти, реалізації підлягає програмне забезпечення, яке призначено для системи перевірки та фільтрації пакетів у мережі на основі технології DPI.

В процесі розробки випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти необхідно виконати наступний обсяг роботи:

а) провести аналіз існуючих систем-аналогів для виявлення їх позитивних і негативних якостей. Результати аналізу врахувати в подальших розробках;

б) вибрати та обґрунтувати методику побудови системи контролю роботи технологічного обладнання на виробництві в автоматизованому режимі. Розробити функціональну та структурну схеми системи;

в) розробити програмне забезпечення системи, що дозволить реалізувати поставлену технічним завданням задачу. Побудувати блок-схеми алгоритмів програми та підпрограми;

г) організувати інтерфейс користувача з метою формування та виводу на екран ЕОМ повідомлень про некоректні дії користувача та нестандартні ситуації в роботі технологічного обладнання;

д) розробити рекомендації по організаційних та методичних заходах, які забезпечать впровадження системи в промислову експлуатацію та її подальшу успішну експлуатацію;

е) провести розрахунки по визначенню економічної ефективності розробленої системи;

ж) розробити заходи по охороні праці при впровадженні та експлуатації системи, а також розробити заходи з цивільного захисту;

з) сформувати висновки про виконаний обсяг робіт та одержані результати.

3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ

3.1 Опис функціонування системи

Яскравий приклад DPI – Bittorrent. Для їхньої ідентифікації здійснюється аналіз послідовності пакетів, що володіють однаковими ознаками, таким як Source_IP:port – Destination_IP:port, розмір пакета, частота відкриття нових сесій в одиницю часу й т.д., по поведінковим (евристичним) моделях, що відповідають таким додаткам. Із цієї причини чудовий NBAR ім'я Cisco хоч і дозволяє детектувати і здійснювати контроль трафіку по додатках, повноцінним рішенням DPI не є, тому що в ньому відсутній ряд важливих компонентів.

Система DPI, як правило, установлюється на границі мережі оператора в розрив існуючих up-link, що йдуть від прикордонних маршрутизаторів. Тим самим, весь трафік, що залишає або входить у мережу оператора, проходить через DPI, що дає можливість його перевірки та фільтрації пакетів й контролю. Для рішення специфічних завдань можна встановлювати цю систему не на границі мережі, а спускати її нижче, ближче до кінцевих користувачів, на рівень BRAS/CMTS/GGSN/... Це може бути корисно тим операторам, які з ряду причин крім утилізації зовнішніх каналів також хочуть вирішувати завдання контролю внутрішніх. Природно, тут мова йде про досить великі сервіси-провайдери з великою розподіленою мережею масштабів країни й з досить дорогими канальними ємностями.

На ринку DPI є моделі на самий різний гаманець. Продуктивність представлених на ринку пристроїв плаває в межах від сотень Мбит/с до 160 Гбит/с FDX у рамках однієї окремо взятої коробки, які, як правило, можна поєднувати в кластери. Відповідно, і вартість плаває досить серйозно – від декількох тисяч до мільйонів доларів США. У випадку з корпоративним сегментам рішення припускають низькошвидкістні підключення по мідних

інтерфейсах типів 10/100/1000. Операторські рішення розраховані на підключення безлічі лінків 1GE і 10GE. Що стосується зовсім дорослих рішень, те поки що ринок 100GE інтерфейсів на мережному встаткуванні досить убогий і дорогий, але як тільки з'явиться перший реальний бізнес-кейс, вендори DPI запропонують відповідні рішення, тому що в деяких з них заготівлі вже є.

Звучить це все, звичайно, не дуже оптимістично, але для багатьох операторів по економічних причинах значно дешевше поставити систему DPI для контролю утилізації каналів, чим розширювати up-link. Причому, зробити це без особливих втрат абонентської бази, тому що давно відомо, що більша частина трафіку генерується приблизно 5% найбільш активних абонентів. І в цьому випадку операторові економічно є сенс знизити абонентську базу, але платити менше грошей за up-link, тому що підуть самі активні ті, хто закачує, через які оператор змушений щомісяця платити немаленьку суму за up-link. Це нічний кошмар будь-якого маркетолога, але в деяких випадках втратити клієнтів – вигідно. Делікатність ситуації полягає в тому, що рано або пізно наступить такий момент, коли всі оператори так чи інакше будуть що-небудь шейпити за допомогою DPI. Тобто якщо сьогодні один оператор почне рубати торренти, самі активні ті, хто закачує разом підуть до іншого. Після цього в того сильно скакне завантаження його каналів і клієнти почнуть скаржитися на те, що погано працює веб-браузинг. Оператор подумає, підрахує, і в підсумку купить DPI. І так доти, поки всі гравці на ринку не обзаведуться подібною системою. Зрозуміло, установка DPI не знімає з оператора завдання по періодичному розширенню up-link і збільшенню швидкості доступу для передплатників. Просто тепер ці розширення не будуть безконтрольними. Тобто оператор завжди буде знати трафік якого типу й у якій кількості піде через його канали, це буде прогнозовано. Зрозуміло, коли мова йде про коробки вартістю $1M, справа не тільки в up-link, необхідно це розуміти. Моя особиста думка в першому наближенні, як користувача послуги широкополосного доступу в інтернет, полягає в тому, що що-небудь різати й блокувати, звичайно ж, погано й зовсім

неправильно. Але, дивлячись очами інженера на те, якими темпами ростуть обсяги трафіку, використання DPI стає порятунком для багатьох операторів, тому що торренти сьогодні здатні забити намертво практично будь-який up-link.

Нова модель послуг

Ми плавно перейшли до завдання розвитку мережі і її послуг. Дивлячись на те, як передплатники користуються купленою ними смугою, які додатки використовують, оператор може вивчати потреби кожної категорії передплатників і пропонувати їм більше гнучкі й зроблені тарифні плани. Приміром, ґрунтуючись на тому, що передплатники тарифу Silver активно користуються послугами сторонньої SIP-телефонії, можна запропонувати їм додатковий пакет, що дозволяє використовувати аналогічний сервіс, надаваний оператором, але й з знижкою. Інші передплатники при бажанні скористатися більше дешевою телефонією будуть мотивовані переходити на більше дорогий тариф, здобуваючи додаткові бонуси у вигляді підвищення швидкості. Можна придумати багато кейсів, це лише один з них. Своє бачення персоналізованих сервісів представила компанія Allot у своїй презентації. Підхід дуже цікавий, і вигідний як для користувача, так і для оператора. Тенденції розвитку телекомунікаційного ринку такі, що для операторів продавати трубу, як вони роблять зараз, незабаром буде просто невигідно, є маса досліджень, що підтверджують це. ARPU не збільшується, конкуренція висока, устаткування необхідно апгрейдити все частіше й частіше, витрати операторів ростуть, а бажання діставати прибуток нікуди не подінеться. Завдання DPI у даному розрізі – реалізувати нові моделі надання послуг кінцевому користувачеві. Деякі світові оператори маленькими кроками вже рухаються до даної ідеї.

DPI відмінно вміє працювати у зв'язуванні з різними VAS (Value Added Services) системами, такими як антиспам, антивірус, відеооптимізатори й т.п. Суть функціонала полягає у відводі частини трафіку за заданим адміністратором критеріями, на сторонні пристрої, для здійснення більше глибокого аналізу й обробки.

Досить легко можна організувати надання користувачам послуг по батьківському контролі, які стають усе більше й більше актуальними.

Наприкінці хотілося б сказати пари слів про те, для чого також закупається DPI, крім як для знущань над абонентами. Устаткування DPI, у зв'язку зі своїм умінням бачити геть усе, що відбувається на мережі, є досить цікавим пристроєм для спецслужб, без яких зараз нікуди. За допомогою DPI спецслужби можуть вести спостереження за мережною активністю того або іншого користувача. Можна перекрити йому VPN, HTTPS та інші принадності, що роблять неможливим аналіз контенту. Зрозуміло, можна закривати доступ користувачів до неугодної влади сайтам, що дуже актуально у зв'язку з останніми подіями в законотворчій діяльності в Україні.

Мережний нейтралітет

І, нарешті, хотілося б сказати пару слів про багатостраждальний мережний нейтралітет, що існує в деяких країнах. Якщо коротко, то операторам під час відсутності перевантажень на up-link нині заборонено блокувати трафік законних/легальних додатків. Тобто почати вибіркове блокування будь-якого трафіку тепер дозволяється тільки у випадку виникнення перевантаження. Але, у той же час, ще немає чітких формулювань на тему того, які саме додатки є законними, а які – немає. По логіці, незаконним може бути тільки контент, а не додаток. Існує незаконний контент, але протоколи HTTP і Bittorrent, за допомогою яких можна здійснювати його передачу – цілком собі легальні. Так що отут є ще досить великий простір для суперечок, а тема, на мій погляд, досить цікава. Поки що в нас мережним нейтралітетом не пахне, тому в операторів на руках – всі карти для керування трафіком за допомогою DPI.

3.2 Розробка структурної схеми

Система призначена для перевірки та фільтрації пакетів у мережі на основі технології DPI. У наш час кібертероризм усе більше знаходить реальні риси.

Ефективно протистояти віртуальному ворогові можна, тільки знаючи його в особу. Тому пропонується класифікація мережних аномалій, що відбиває сучасний стан проблеми.

Структурна схема розробленої системи зображена на рисунку 3.1. На ній показано структуру системи, яка складається з наступних блоків:

1. Система виявлення аномалій, яка включає в себе:

– Модуль взаємодії з базою даних перевірки та фільтрації пакетів.

– Модуль DPI.

– Модуль обробки результатів.

– Інтерфейс адміністратора.

– Базу даних виборок.

– Базу даних шаблонів нормального поводження трафіку.

2. Система перевірки та фільтрації пакетів у мережі, яка включає в себе Базу даних результатів перевірки та фільтрації пакетів у мережі.

Поява аномалій у різних точках спостереження протягом невеликого проміжку часу говорить про можливий взаємозв'язок подій. Об'єкт, що має більш ранній за часом момент виникнення аномалії може бути джерелом нестандартного поводження ряду даних. Досліджуючи виникнення аномалій на різних пристроях, можливим є побудова дерева аномалій, що веде від джерела через посередників різних рівнів до приймачів і навпаки: від приймача до джерела.

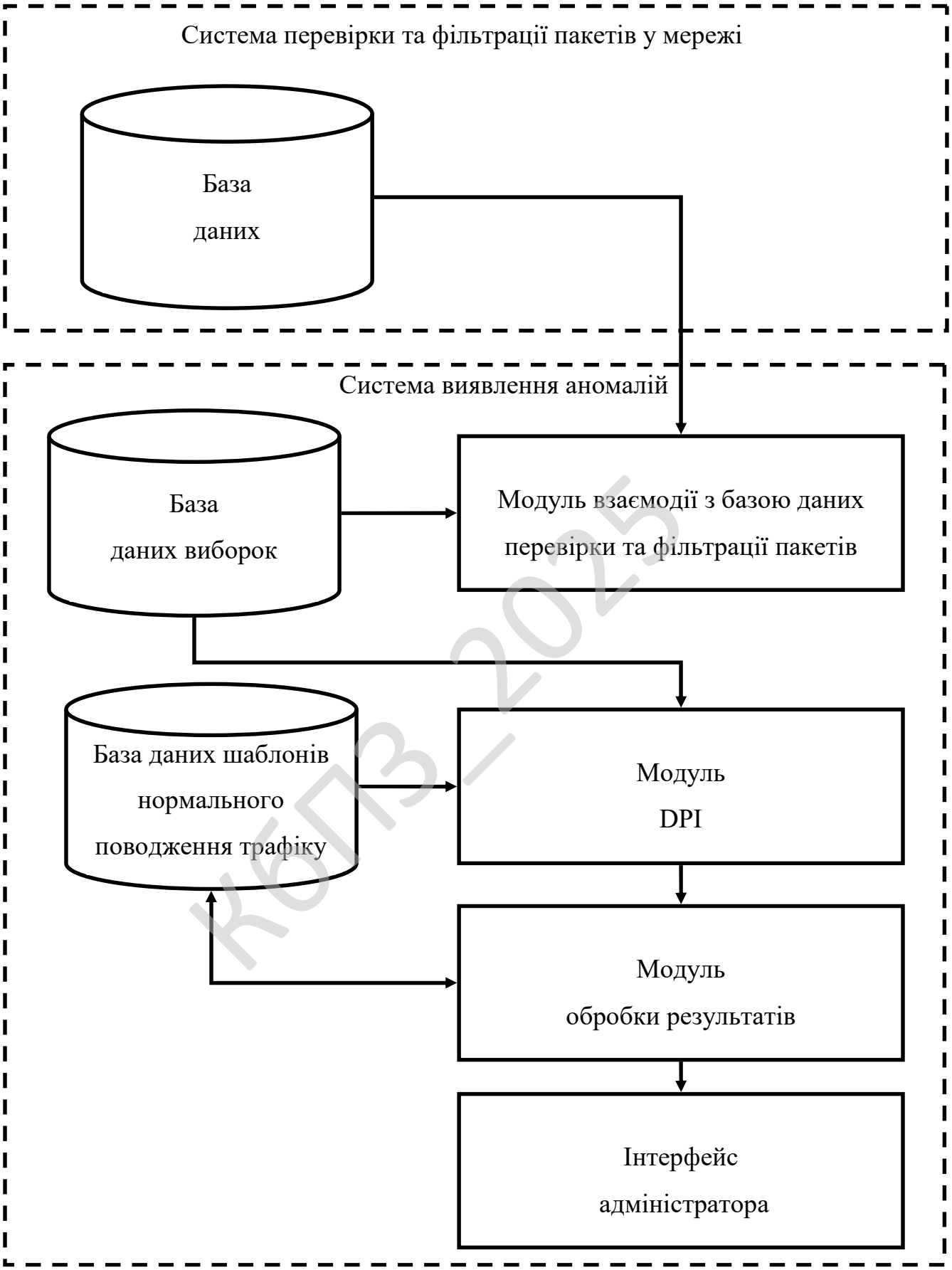


Рисунок 3.1 – Структурна схема системи

Більшість механізмів збору даних на основі пакетів не залежать від вмісту пакета. Зосереджуючись лише на форматах пакетів, збирач вирішує, які пакети слід підтримувати. На відміну від цих підходів, технологія Deep Packet Inspection (DPI) є технологією з урахуванням вмісту для ідентифікації трафіку на рівні мережевих програм. Ця технологія може бути розгорнута як механізм ідентифікації шлюзу на рівні програми, застосовний у ситуаціях, коли потік керування та потік трафіку розділені. У цій ситуації аналіз транспортного потоку не має значення. Натомість необхідно провести аналіз потоку керування. У традиційному підході пристрої моніторингу зазвичай збирають і виявляють трафік через порти. Багато програм часто змінюють свої порти в автоматичному або ручному режимі, тому використання інформації про порт не є ефективним для збору очікуваних даних.

Механізм DPI має дрібнозернистий розділ для збору даних і вимірювання витрати. Загалом архітектуру DPI можна розділити на попереднє зіставлення шаблону та перевірку функцій. Перший розрізняє різні шаблони за попередньо визначеними правилами. Найбільш поширеними технологіями є регулярні вирази (RE) і безконтекстні граматики (CFG). І доступні шаблони зазвичай є типами протоколів. Функції протоколів різноманітні. Ними можуть бути номери портів, символьні слова та шлюзи на прикладному рівні. Як ми обговорювали вище, виявлення портів може бути марним, оскільки порти програм часто змінюються. Фільтрація символьних слів забезпечує надійну альтернативну політику, коли механізм виявлення портів вимкнено. Таким чином можна розгорнути багато сучасних мережевих бізнес-додатків і програм керування, наприклад мережевий виставлення рахунків, розміщення реклами та ідентифікацію типу додатків. Однак, коли корисне навантаження пакета криптографічно зашифровано або захищено, цей підхід не працює.

Було проведено опитування щодо РЕ технології DPI. Вони перерахували відповідні вимоги, яким мають задовольняти механізми DPI. RE необхідно вичерпати формати даних, інакше деякий вміст корисного навантаження буде

пропущено для розпізнавання. Однак обмеження апаратних ресурсів, постійно зростаюча швидкість з’єднання та масштаб шаблону все ще є складними проблемами, які слід вирішити.

Механізми RE та CFG не можуть ефективно підходити до вмісту різних текстових конструкцій. Подібно до підходів RE, механізми CFG є складними через обмежені можливості вираження та неоднозначність деяких слів. Крім того, обидва ці механізми засновані на ручних операціях. Це передбачає низьку можливість повторного використання та легку схильність до помилок через зростаючу різноманітність і складність корисних навантажень. Відповідно, вони розробили архітектуру рахункових автоматів під назвою FlowSifter, щоб реалізувати вилучення та збір даних поля 7 рівня. Їхня граматика вилучення генерується оптимізатором граматики, який об’єднує інформацію бібліотеки протоколу та зручну специфікацію вилучення. Цей механізм збору польових даних рівня 7 долає спільну дилему виснажливої та неефективної обробки вручну та досягає мети автоматичного та адаптивного збору даних, навіть попередніх знань недостатньо. Експериментальні тести враховували три фактори: швидкість, пам'ять і складність визначення екстрактора. Згідно з результатами тестування, цей механізм має хороші показники за всіма трьома факторами. Таким чином, IN і EF можуть підтримуватися. Однак запропонований механізм все ще не може вирішити загальну проблему DPI, тобто криптографічно захищений трафік все ще нерозбірливий.

3.3 Розробка функціональної схеми

Функціональна схема розробленої системи зображена на рисунку 3.2. Так, як функціональна схема є більш подобним описом функціональних можливостей структурної схеми, то вона буде представляти собою, більш детальний варіант структурної схеми.

З рисунку видно, що розроблена система складається з наступних частин:

– Блок визначення топології мережі.

– Блок виявлення аномального поводження трафіку.

– Блок зберігання результатів.

– Блок визначення виду атаки.

– Блок перевірки та фільтрації пакетів у мережі.

– Блок аналізу мережної статистики.

Блок визначення топології мережі

Блок визначення топології мережі:

– Блок використання відомостей із загальної системи моніторингу мережі, а не опитування пристрою додатково.

– Блок складання списку пристроїв у мережі, автоматично, ґрунтуючись на дані системи моніторингу.

– Блок побудови топології мережі, за станом на задану дату й відстеження змін у топології протягом часу.

– Блок автоматичного визначення рівнів ієрархії пристроїв у мережі, з виділенням периферійних, проміжних і центральних вузлів;

– Блок побудови топології мережі, незалежно від використовуваної системи моніторингу й програмно-апаратних платформ;

– Блок комбінувати показників, на основі яких визначаються зв'язки між пристроями, і при їхньому обчисленні виконувати перевірку на значимість із використанням статистичних критеріїв.

Блок виявлення аномального поводження трафіку

Блок виявлення аномального поводження трафіку:

– Блок визначення профілю поводження нормального трафіку.

– Блок заміни направлення трафіку.

– Блок усунення аномального трафіку.

При первісному розгортанні рішення по DDoS адміністратор створює профіль поводження нормального трафіку. Цей процес іменується навчанням.

Компанія використовує додатки звичайним образом протягом 24 годин протягом одного тижня, і трафік додатка проходить через Детектор аномалій трафіку. У період навчання Детектор аномалій трафіку збирає базову інформацію для розуміння нормальної роботи мережі, куди входять:

– Інтенсивність пакетів для кожного типу пакетів, обмірювана як кількість пакетів у секунду (pps).

– Співвідношення пакетів, наприклад, співвідношення пакетів SYN і пакетів FIN.

– Кількість одночасних TCP-з'єднань, відкритих одним джерелом.

Базова інформація збирається по кожній цільовій адресі хост-ПК, цільовій підмережі, вихідній адресі хост-ПК і вихідній підмережі.

Після закінчення періоду навчання Детектор аномалій трафіку переводиться в режим моніторингу, а Блок усунення аномального трафіку – у резервний режим готовності. Доти, поки немає атаки, що активно розвивається, вхідний трафік з мережі Інтернет проходить через комутатор без якого-небудь втручання з боку Блоку усунення аномального трафіку. Копія вхідного трафіку посилає для аналізу на Детектор аномалій трафіку через зовнішній аналізатор протоколів (SPAN) або віртуальні списки ACL. Якщо Детектор аномалій трафіку виявляє аномальне в порівнянні з базовою інформацією поводження трафіку, починається процес усунення:

– Детектор аномалій трафіку направляє в Блок усунення аномального трафіку команду почати процес зміни напрямку.

– Блок усунення аномального трафіку відхиляє (“захоплює”) трафік, адресований на атакуєму IP-адресу, переадресуючи його на самого себе.

– Блок усунення аномального трафіку піддає трафік багатоступінчастому аналізу й застосовує контрзаходи для відділення благонадійних джерел від джерел атаки. Цей процес іменується очищенням або вичищенням.

– Блок усунення аномального трафіку скидає трафік атаки й пересилає благонадійний трафік назад на нормальний маршрут проходження трафіку до мети. Цей процес іменується ін'єкцією.

Детектор аномалій трафіку

Детектор аномалій трафіку – це пасивний пристрій моніторингу, що постійно виявляє ознаки, що вказують на присутність атаки DDoS, спрямованої проти захищеного місця призначення, також іменованого зоною. Це може бути сервер, інтерфейс міжмережного екрана або інтерфейс маршрутизатора. Детектор аномалій трафіку аналізує копії всього вхідного трафіку, адресуємого в захищені зони, через SPAN або відгалуження пасивної мережі. Цей аналіз включає зіставлення поточного поводження трафіку з базовими граничними параметрами, які також іменуються зональною політикою, для виявлення аномального поводження трафіку. Якщо аномальне поводження виявлене й виглядає як можлива атака, Детектор аномалій трафіку через позаполосну управлінську мережу Ethernet посилає в Блок усунення аномального трафіку сигнал про початок аналізу й усунення атаки.

Блок усунення аномального трафіку

Блок усунення аномального трафіку – це автономний пристрій аналізу й фільтрації трафіку. Починаючи прийом трафіку, адресованого в конкретну зону, що, очевидно, піддається атаці, Блок усунення аномального трафіку проводить точний аналіз цього трафіку. Якщо результати аналізу підтверджують, що трафік злочинний, Блок усунення аномального трафіку застосовує контрзаходи, наприклад, механізми анти-спуфінга й фільтрацію різного рівня (таблиця 3.1). Кінцевий результат полягає в тому, що трафік зі злочинних джерел скидається, а трафік із благонадійних джерел пересилається в передбачений пункт призначення.

Атаки DDoS – виявлення й усунення

Перерахуємо типи атак DDoS, які може виявляти й усувати Блок усунення аномального трафіку.

1. Атаки із заповненням смуги пропущення.

Лавинні атаки зі спуфінгом або без спуфінга:

– Прапор TCP (SYN, SYN-ACK, ACK, FIN).

– Протокол керування повідомленнями в Інтернет (ICMP).

– Протокол користувальницьких датаграмм (UDP).

Приклади: лавинна атака SYN, smurf, LAND і UDP – лавинні атаки.

Атаки зомбі-комп'ютерів/мереж зомбі-комп'ютерів, у яких кожний вихідний зомбі-ПК або мережа відкриває множинні TCP-з'єднання й, у деяких випадках, видає багаторазові запити HTTP.

Атаки DNS, наприклад, лавинна атака із запитами DNS.

2. Атаки з дефіцитом ресурсів:

– Атаки пакетного розміру, характерна риса яких – фрагментированні або великі пакети. Приклади: teardrop і ping-of-death.

– Атаки зомбі-комп'ютерів/мереж зомбі-комп'ютерів з низькою інтенсивністю схожі на атаки із заповненням смуги пропущення за тим виключенням, що кожне джерело атаки посилає множинні запити з невеликим обсягом в одиницю часу.

– Атаки DNS з рекурсивним переглядом DNS.

Можливі варіанти зміни напрямку трафіку

Фахівці з IT можуть використовувати описані нижче варіанти зміни напрямку трафіку з його пересиланням з мережі, розташованого вище лежачого оператора зв'язку, на Блок усунення аномального трафіку. Цей процес також іменується “захватом” трафіку:

– Повідомлення прикордонного шлюзового протоколу (Border Gateway Protocol, BGP) із Блок усунення аномального трафіку на маршрутизатори, розташовані у вище лежачого оператора зв'язку, з інформацією про те, що трафік, адресований на захищену адресу призначення, буде переспрямований на Блок усунення аномального трафіку.

– Використання зовнішніх механізмів зміни напрямку трафіку, наприклад, маршрутизаторів віддаленого відновлення BGP.

– Повідомлення про ін'єкцію очищеного трафіку на маршруті (Route Health Injection, RHI) від Блок усунення аномального трафіку для процесу маршрутизації в Catalyst серії 6500 або в систему нагляду серії 7600. Ці повідомлення поміщають статичний маршрут у глобальну таблицю маршрутизації, у якій модуль Блок усунення аномального трафіку позначений як наступний вузол.

Можливі варіанти ін'єкції трафіку

Ін'єкція трафіку – це процес, застосовуваний у Блок усунення аномального трафіку для пересилання очищеного благонадійного трафіку в точку призначення, що піддається атаці. Рішення підтримує різні варіанти ін'єкції трафіку. У варіанті 2-ого рівня топології, очищений трафік пересилається із Блок усунення аномального трафіку на статично-конфігуруєму наступну адресу заходу. Ця адреса перебуває на маршрутизаторі, розташованому нижче й з'єднаним з тої ж VLAN або підмережею, що й інтерфейс/VLAN ін'єкції трафіку. Ін'єкцію трафіку на 2-му рівні найпростіше конфігурувати, оскільки тут не потрібно вносити які-небудь істотні зміни в конфігурацію маршрутизатора, розташованого нижче.

Варіанти ін'єкції трафіку 3-го рівня:

– Маршрутизація й пересилання по VPN (VPN Routing and Forwarding, VRF).

– Маршрутизація на основі політики (Policy-Based Routing, PBR).

– Транкінг VLAN (VLAN Trunking).

– Інкапсуляція по загальній маршрутизації (GRE) або інкапсуляція IP у тунелі IP (IPIP).

Блок визначення виду атаки

Блок визначення виду атаки:

– Атака ARP-spoofing на таблицю mac-адрес комутаторів.

– Широкомовний шторм.

– Додатки, що роблять інтенсивне широкомовне розсилання, наприклад: широкомовні чати й мережні ігри.

ARP-spoofing

ARP-spoofing – техніка атаки в Ethernet мережах, що дозволяє перехоплювати трафік між хостами. Заснована на використанні протоколу ARP.

При використанні в розподіленій обчислювальній системи (РВМ) алгоритмів віддаленого пошуку існує можливість здійснення в такій мережі типової віддаленої атаки «помилковий об'єкт РВМ». Аналіз безпеки протоколу ARP показує, що, перехопивши на атакуючому хості усередині даного сегмента мережі широкомовний ARP-запит, можна послати помилкову ARP-відповідь, у якій оголосити себе шуканим хостом (наприклад, маршрутизатором), і надалі активно контролювати мережний трафік дезінформованного хосту, впливаючи на нього за схемою «помилковий об'єкт РВМ».

Протокол ARP призначений для перетворення IP-адрес в MAC-адреси. Найчастіше мова йде перетворенні в адреси Ethernet, але ARP використовується й у мережах інших технологій: Token Ring, FDDI і інших.

Алгоритм роботи ARP

Протокол може використовуватися в наступних випадках:

1. Хост А хоче передати IP-пакет вузлу В, що перебуває з ним в одній мережі.

2. Хост А хоче передати IP-пакет вузлу В, що перебуває з ним у різних мережах, і користується для цього послугами маршрутизатора R.

У кожному із цих випадку вузлом А буде використовуватися протокол ARP, тільки в першому випадку для визначення MAC-адреси вузла В, а в другому – для визначення MAC-адреси маршрутизатора R. В останньому випадку пакет буде переданий маршрутизатору для подальшої ретрансляції.

Далі для простоти розглядається перший випадок, коли інформацією обмінюються вузли, що перебувають безпосередньо в одній мережі. (Випадок коли пакет адресований вузлу, який знаходиться за маршрутизатором, відрізняється тільки тим, що в пакетах переданих після того як ARP-перетворення завершене, використовується IP-адреса одержувача, але MAC-адреса маршрутизатора, а не одержувача.).

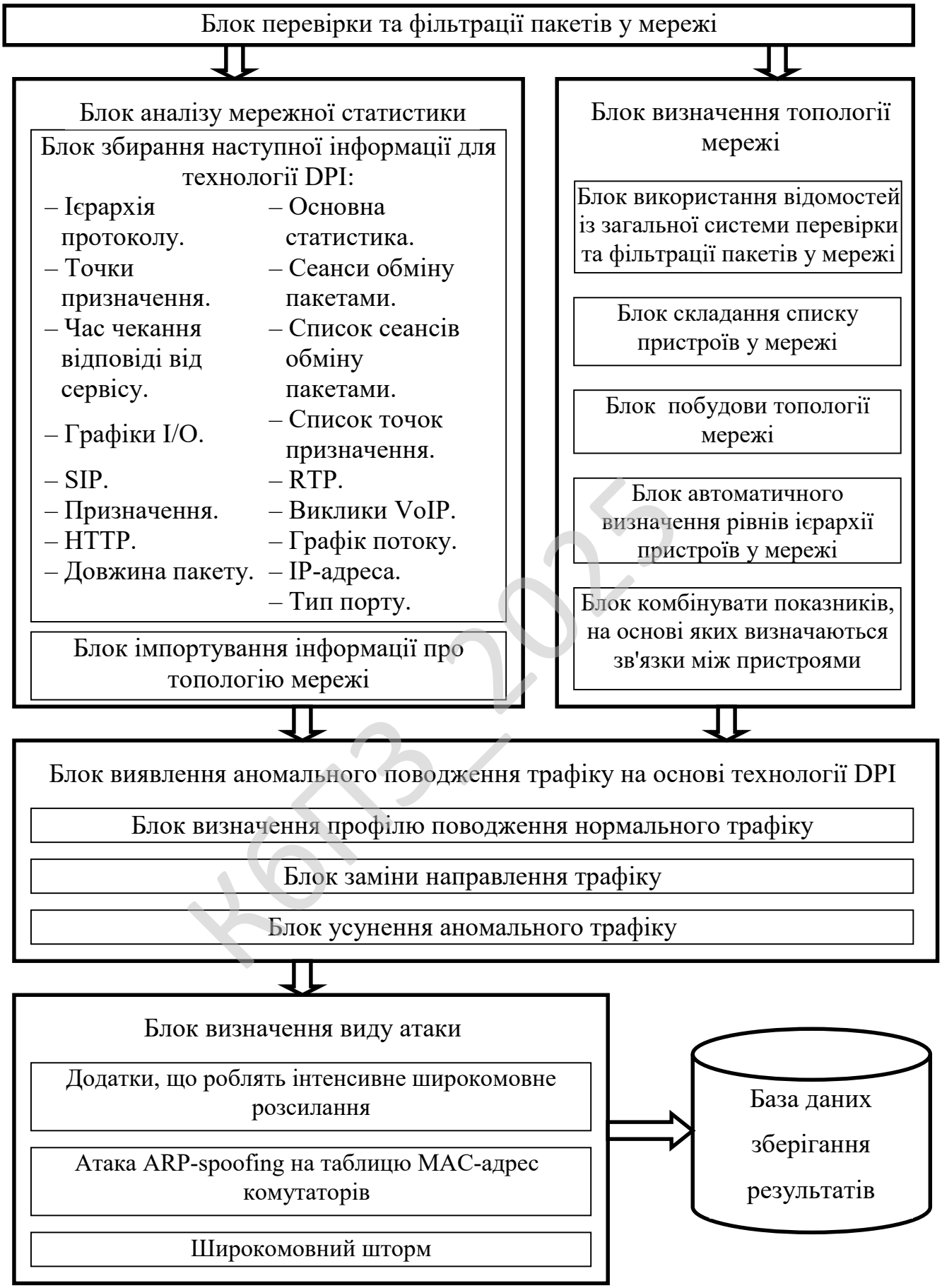


Рисунок 3.2 – Функціональна схема системи

Проблеми ARP

Протокол ARP є абсолютно незахищеним. Він не має ніякого способу перевірки дійсності пакетів: як запитів, так і відповідей. Ситуація стає ще більш складною, коли може використовуватися мимовільний ARP (gratuitous ARP).

Мимовільний ARP – таке поводження ARP, коли ARP-відповідь надсилається, коли в цьому (з погляду одержувача) немає особою необхідності. Мимовільна ARP-відповідь це пакет-відповідь ARP, присланий без запиту. Він застосовується для визначення конфліктів IP-адрес у мережі: як тільки станція одержує адресу по DHCP або адреса привласнюється вручну, розсилається ARP-відповідь gratuitous ARP.

Мимовільний ARP може бути корисний у наступних випадках:

– Відновлення ARP-таблиць, зокрема, у кластерних системах.

– Інформування комутаторів.

– Повідомлення про включення мережного інтерфейсу.

Незважаючи на ефективність мимовільного ARP, він є особливо небезпечним, оскільки з його допомогою можна запевнити віддалений вузол у тому, що MAC-адреса якої-небудь системи, що перебуває з нею в одній мережі, змінилася й указати, яка адреса використовується тепер.

До виконання ARP-spoofing'a в ARP-таблиці вузлів A і B існують записи з IP- і MAC-адресами один одного. Обмін інформацією виробляється безпосередньо між вузлами A і B.

У ході виконання ARP-spoofing'a комп'ютер **C**, що виконує атаку, відправляє ARP-відповіді (без одержання запитів):

– вузлу A: з IP-адресою вузла B і MAC-адресою вузла C;

– вузлу B: з IP-адресою вузла A і MAC-адресою вузла C.

У силу того що комп'ютери підтримують мимовільний ARP (gratuitous ARP), вони модифікують власні ARP-таблиці й поміщають туди записи, де замість справжніх MAC-адрес комп'ютерів A і B знаходиться MAC-адреса комп'ютера C.

Після того як атака виконана, коли комп'ютер А хоче передати пакет комп'ютеру В, він знаходить в ARP-таблиці запис (він відповідає комп'ютеру С) і визначає з її MAC-адресу одержувача. Відправлений по цьому MAC-адресу пакет приходить комп'ютеру С замість одержувача. Комп'ютер С потім ретранслює пакет тому, кому він дійсно адресований – тобто комп'ютеру В.

Широкомовний шторм

Широкомовний шторм – лавина (сплеск) широкомовних акетів (на другому рівні моделі OSI – кадрів). Розмноження некоректно сформованих широкомовних повідомлень у кожному вузлі приводить до експонентного росту їхнього числа й паралізує роботу мережі. Звичайно такі пакети використовуються мережними сервісами для оповіщення станцій про свою присутність. Вважається нормальним, якщо широкомовні пакети становлять не більше 10% від загального числа пакетів у мережі.

Також досить часто до шторму приводять кільця в мережі при некоректному налаштуванні протоколу Spanning Tree, оскільки в заголовку пакетів Ethernet немає інформації про час життя кадру, як, наприклад, у пакетів IP. Крім цього широкомовний шторм застосовується (навмисно) зломщиками.

Відповідно до галузевого стандарту де-факто число широкомовних і багатоадресних кадрів у мережі не повинне перевищувати 8-10% від загального числа кадрів.

Широкомовний кадр – це кадр, адресований всім станціям у домені мережі. Багатоадресний кадр – це кадр, адресований групі станцій у домені мережі. Оскільки широкомовний кадр адресований всім станціям, то, одержавши його, станції повинні перервати свою роботу й обробити такий кадр. Це сповільнює роботу всієї мережі.

Якщо відношення числа широкомовних кадрів до загального числа кадрів більше 10%, то такий ефект називається "широкомовним штормом".

Широкомовний шторм може бути наслідком дефектів устаткування або неправильного налаштування параметрів активного встаткування. Найчастіше це явище спостерігається в розподілених мережах NetWare, побудованих на основі комутаторів, або коли дані між сегментами або доменами мережі можуть

передаватися більш ніж по одному потенційному шляху. Якщо один з комутаторів такої мережі не підтримує протокол Spanning Tree (звичайно IEEE 802.1d) або останній неправильно настроєний або збоїть, то в мережі починається некерована циркуляція широкомовних кадрів.

Виявлення "широкомовного шторму" є не настільки тривіальним завданням, як це може здатися на перший погляд. Для його виявлення недостатньо взяти загальне число широкомовних кадрів і поділити його на загальне число кадрів, що пройшли по мережі.

Для цього ви повинні визначити: яку частку становлять широкомовні кадри в кожний інтервал часу (наприклад, за одну мінуту) і яка при цьому утилізація каналу зв'язку. Якщо, наприклад, за одну мінуту по мережі пройшло 4 кадри, а 2 з них були широкомовними, то це ще не виходить, що ви спостерігаєте "широкомовний шторм".

Захист від широкомовних штормів (broadcast storm)

Одна з характерних несправностей мережного програмного забезпечення – мимовільна генерація з високою інтенсивністю широкомовних пакетів. Широкомовним штормом вважається ситуація, у якій відсоток широкомовних пакетів перевищує 20% від загальної кількості пакетів у мережі. Звичайний комутатор або міст сліпо передає такі пакети на всі свої порти, як того вимагає його логіка роботи, засмічуючи, таким чином, мережу. Боротьба із широкомовним штормом у мережі, з'єднаної комутаторами, жадає від адміністратора відключення портів, що генерують широкомовні пакети. Маршрутизатор не поширює такі ушкоджені пакети, оскільки в коло його завдань не входить копіювання широкомовних пакетів в усі поєднувані їм мережі. Тому маршрутизатор є прекрасним засобом боротьби із широкомовним штормом, щоправда, якщо мережа розділена на достатню кількість підмереж.

Блок аналізу мережної статистики

Блок збирання наступної інформації:

– Основна статистика (Summary).

– Ієрархія протоколу (Protocol Hierachy).

– Сеанси обміну пакетами (Conversations).

– Точки призначення (Endpoints).

– Графіки I/O (IO Graphs).

– Список сеансів обміну пакетами (Conversation List).

– Список точок призначення (Endpoint List).

– Час чекання відповіді від сервісу (Service Response Time).

– RTP.

– SIP.

– Виклики VoIP (VoIP Calls).

– Призначення (Destination).

– Графік потоку (Flow Graph).

– HTTP.

– IP-адреса (IP address).

– Довжина пакету (Packet Length).

– Тип порту (Port Type).

Розпишемо їх більш детально.

1. Основна статистика. Доступні такі елементи основної статистики, як:

– Властивості захоплених файлів.

– Час захвату.

– Інформація про фільтр захвату.

– Інформація про фільтр відображення.

2. Ієрархія протоколу. Статистика ієрархії протоколу допомагає аналізувати пакети, розбиваючи відображені дані, які належать чинному рівню OSI.

3. Сеанси обміну пакетами. Якщо ви використовуєте протокол TCP/IP або програму, яка працює із цим протоколом, ви маєте побачити чотири активних вкладок для обміну пакетами за допомогою Ethernet, IP, TCP та UDP. «Діалог» між комп'ютерами відображає трафік між двома активними хостами. Номер, зазначений на вкладці після назви протоколу, означає кількість «діалогів» між

хостами. Номер, зазначений на вкладці після назви протоколу, означає кількість «діалогів» між хостами, наприклад, «Ethernet:6».

4. Точки призначення. Точки призначення забезпечують статистику даними про відправку та прийом пакетів. Номер, зазначений на вкладці після назви протоколу, вказує на кількість точок призначення. Наприклад, «Ethernet:6».

5. Графіки I/O. Основний графік може бути отриманий за допомогою команди «IO graphs» (Графіки I/O). Ще декілька графіків можуть бути додані у тому ж вікні на основі фільтрів відображення.

6. Час чекання відповіді від сервісу. 13 протоколів доступні для глибокого аналізу.

7. RTP. RTP (Real-time Transport Protocol, протокол передачі у реальному часі, RFC 3550) – це протокол для передачі звука та відео через IP-мережу. Він працює у початку протоколу дейтаграм користувача (User Datagram Protocol, UDP). Він часто використовується у сукупності з протоколами SIP або H.233, забезпечуючи виконання сигнальних завдань.

8. SIP. SIP (Session Initiation Protocol, протокол встановлення сесії, RFC 3261) – це сигнальний протокол, який оголошує відео– або VoIP-сесії. Він працює разом із протоколом RTP, який використовується для передачі мультимедійних даних.

9. Виклики VoIP.

VoIP (Voice over IP, голосовий зв’язок за допомогою Інтернету) взагалі використовує два типи протоколів:

– сигнальні протоколи, такі, як SIP або H.323

– переносні протоколи, наприклад, RTP

10 Призначення. Відображення усіх IP-адрес призначення мережевих пакетів.

11. Графік потоків. Графіки потоків забезпечує послідовний аналіз TCP-з’єднань. Перші три строки містять оголошення TCP– з’єднання з послідовностями «SYN», «SYN ACK» та «ACK».

12 HTTP. HTTP (Hypertext Transfer Protocol, протокол передачі гіпертексту) – це протокол типу «клієнт-сервер», який використовується для передачі HTML-файлів. HTTP-клієнт (у більшості випадків це web-браузер) відсилає HTTP-запит до web-серверу із полем «URL», який допомагає знайти потрібний файл. Web-сервер відповідає HTTP-пакетом та забезпечує клієнт необхідною web-сторінкою.

Меню «HTTP» містить три підменю:

– «Load Distribution» (Розподіл пакетів).

– «Packet Counter» (Лічильник пакетів).

– «Requests» (Запити).

14 IP-адреса. Відображення IP-адреси джерела або призначення мережевих пакетів.

15. Довжина пакету.

16. Тип порту. Відображення статистики портів TCP або UDP.

Розглянувши усі блоки функціональної схеми перейдемо до розгляду діаграми взаємодії процесів, які відбуваються у системі.

3.4 Розробка діаграми процесів

Діаграма процесів розробленої системи зображена на рисунку 3.3. Після початку роботи розробленого ПЗ ми потрапляємо до головного блоку системи звідки через ланку дій відбувається наступне:

– Інтерфейс ПЗ.

– Налаштування системи.

– Налаштування підсистеми моніторингу трафіку мережі.

– Налаштування підсистеми Deep Packet Inspection.

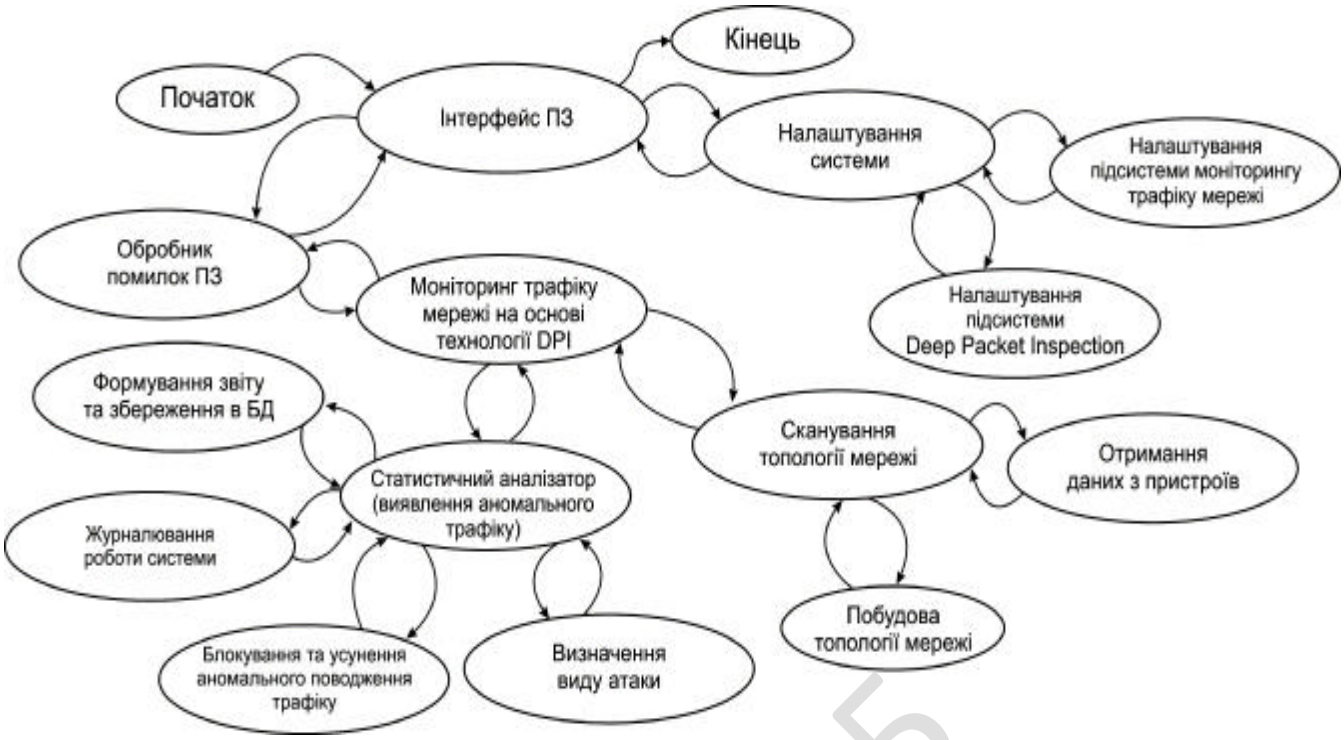


Рисунок 3.3 – Діаграма взаємодії процесів

– Обробник помилок ПЗ.

– Моніторинг трафіку мережі на основі технології DPI.

– Сканування топології мережі.

– Отримання даних з пристроїв.

– Побудова топології мережі.

– Статистичний аналізатор (виявлення аномального трафіку).

– Формування звіту та збереження в БД.

– Журналювання роботи системи.

– Блокування та усунення аномального поводження трафіку.

– Визначення виду атаки.

Таким чином, розглянувши опис системи, структурну, функціональну схеми системи, та діаграму взаємодії процесів перейдемо до опису блок-схем основної програми, та підпрограм, які використовуються, для реалізації системи.

4 РЕАЛІЗАЦІЯ ПРОЕКТУ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ПРАВИЛЬНІСТЬ ПРОЕКТНИХ РІШЕНЬ

4.1 Блок-схеми та опис алгоритмів функціонування системи

Первинною стадією без якої не відбувається розробка програмного забезпечення це звичайно розробка блок-схем. На рисунку 4.1 зображена основна блок-схема програми, на рисунку 4.2 зображено роботу підпрограми. З якої видно що робота основної програми складається з початкових етапів ініціалізації ПЗ, перевірки наявності ресурсів системи, блоку початку основного циклу з чеканням запиту від користувача в якому відбувається виклик підпрограми та останньої стадії – перевірка поточного стану з завершенням роботи розробленого ПЗ. При роботі підпрограми виконується основний функціонал системи з циклічними послідовностями, перевіркою поточного стану та поверненням в основну програму прапорів стану виконання.

Було використано підходи з використанням UML, це уніфікована мова моделювання, використовується у парадигмі об'єктно-орієнтованого програмування. Є невід'ємною частиною уніфікованого процесу розробки програмного забезпечення. UML є мовою широкого профілю, це відкритий стандарт, що використовує графічні позначення для створення абстрактної моделі системи, називаної UML-моделлю.

UML був створений для визначення, візуалізації, проектування й документування в основному програмних систем. UML не є мовою програмування, але в засобах виконання UML-моделей як інтерпретованого коду можлива кодогенерація. UML може бути застосовано на всіх етапах життєвого циклу аналізу бізнес-систем і розробки прикладних програм. Різні види діаграм які підтримуються UML, і найбагатший набір можливостей представлення певних

аспектів системи робить UML універсальним засобом опису як програмних, так і ділових систем. Діаграми дають можливість представити систему (як ділову, так і програмну) у такому вигляді, щоб її можна було легко перевести в програмний код.

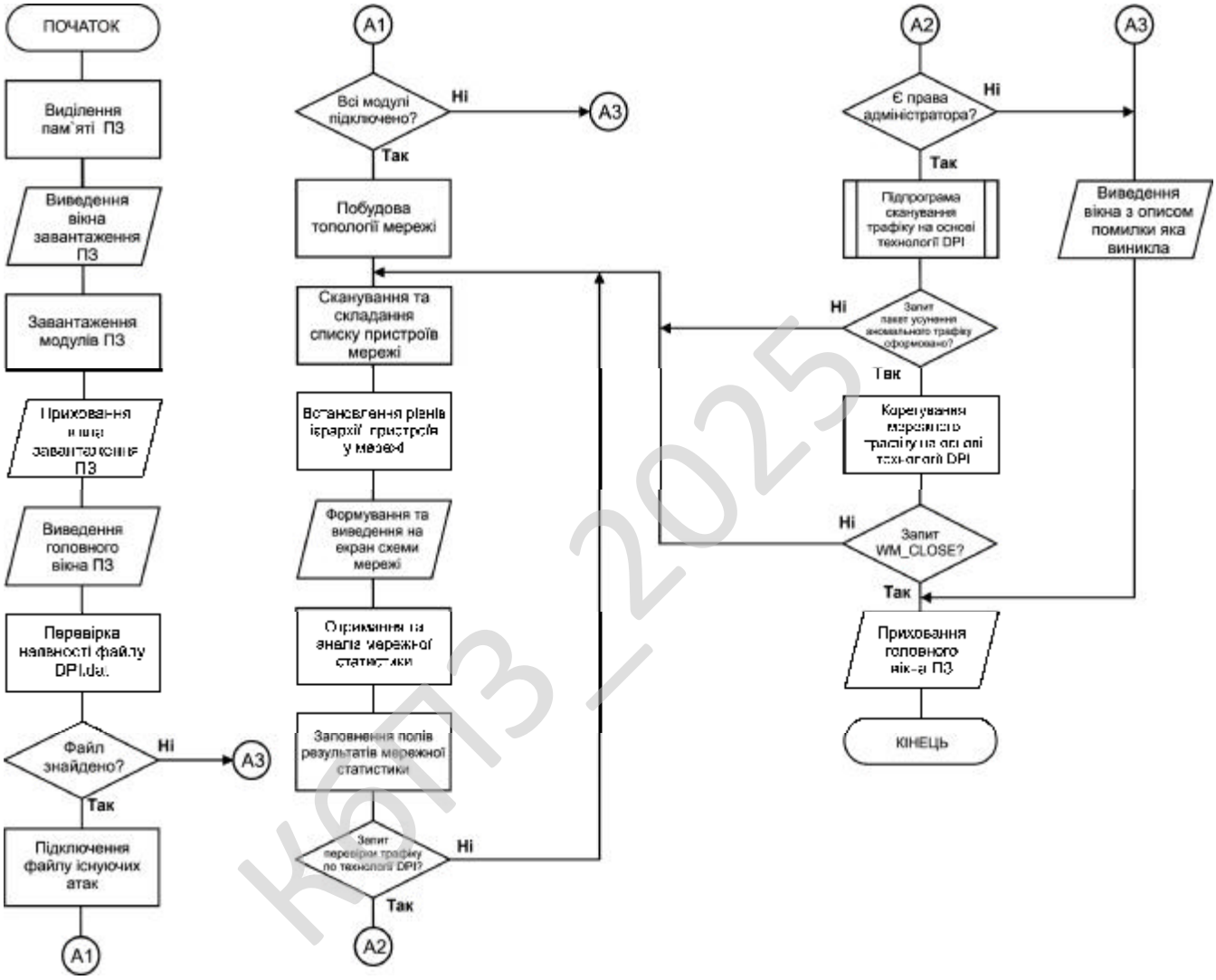


Рисунок 4.1 – Блок схема основної програми

Основною причиною використання мови UML є спілкування розробників між собою. Крім того, UML спеціально створювалася для оптимізації процесу розробки програмних систем, що дозволяє збільшити ефективність їх реалізації у кілька разів і помітно поліпшити якість кінцевого продукту.

UML прекрасно зарекомендувала себе в багатьох успішних програмних проектах. Засоби автоматичної генерації кодів дозволяють перетворювати моделі мовою UML у вихідний код об’єктно-орієнтованих мов програмування, що ще більш прискорює процес розробки.

Практично усі CASE-засоби (програми автоматизації процесу аналізу і проектування) мають підтримку UML. Моделі розроблені в UML, дозволяють значно спростити процес кодування і направити зусилля програмістів безпосередньо на реалізацію системи.

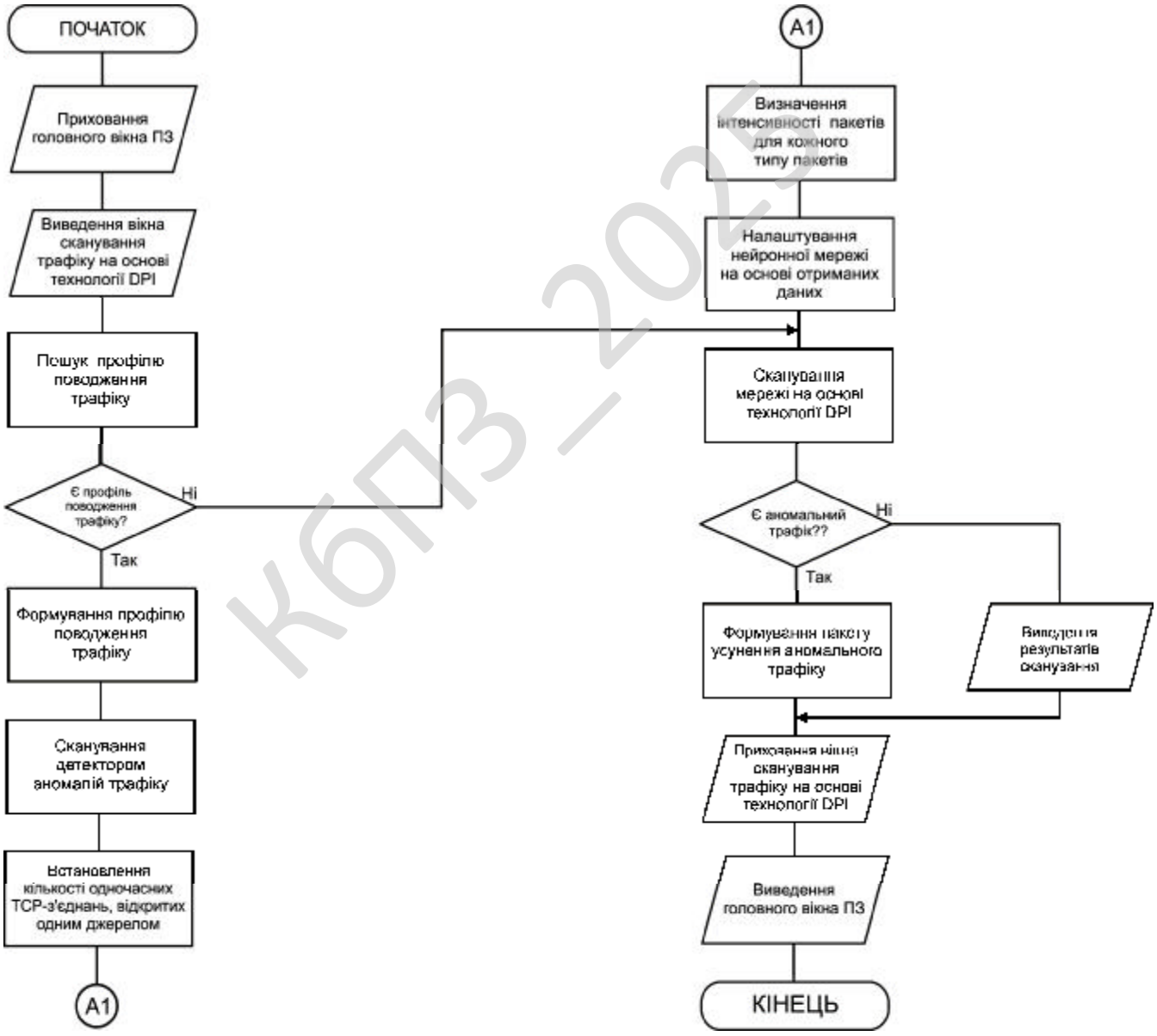


Рисунок 4.2 – Блок схема підпрограми

Діаграми підвищують супроводжуваність проекту і полегшують розробку документації.

UML необхідний:

– Керівникам проектів, які керують розподілом завдань і контролем за проектом.

– Проектувальникам інформаційних систем які розробляють технічні завдання для програмістів.

– Бізнес-аналітикам, які досліджують реальну систему і здійснюють інжиніринг і реінжиніринг бізнесу компанії.

– Програмістам які реалізовують модулі інформаційної системи.

При модифікації системи об'єктний підхід дозволяє легко включати в систему нові об'єкти і виключати застарілі без істотної зміни її життєздатності.

Використання побудованої моделі при модифікаціях системи дає можливість усунути небажані наслідки змін, оскільки вони не ламають структури системи, а тільки змінюють поведінку об'єктів.

Опис алгоритмів функціонування системи

Система перевірки та фільтрації пакетів у мережі на основі технології DPI працює з використанням мови програмування Python вона здійснює захоплення пакетів у реальному часі що дозволяє отримувати дані з мережевого трафіку система отримує інформацію про джерело призначення і вміст пакетів після цього відбувається детальний аналіз заголовків і корисного навантаження пакету.

З метою виявлення підозрілих шаблонів які характерні для шкідливих даних система приймає рішення про блокування або пропуск пакету залежно від результатів аналізу що забезпечує безпеку мережі додатковий модуль логування зберігає дані про прийняті рішення що дозволяє проводити статистичний аналіз ефективності роботи системи архітектура розроблена за принципами модульності де кожен компонент виконує свою чітку функцію що сприяє розширенню функціоналу та інтеграції з іншими мережевими сервісами приклад вихідного коду.

Це демонструє створення окремих функцій для захоплення аналізу та фільтрації пакетів а також містить розрахунки середнього часу обробки пакету що підтверджує можливість обробки до 1500 пакетів за секунду при високій навантаженості тестові дані отримані шляхом симуляції потоку ста пакетів свідчать про відповідність системи заданим параметрам та правильність обраних проектних рішень.

Система перевірки та фільтрації пакетів у мережі використовує технологію DPI для глибокого аналізу вмісту кожного пакету вона працює в реальному часі.

ЦЕ забезпечує оперативне виявлення загроз та швидку реакцію на підозрілий мережевий трафік архітектура системи побудована на модульному принципі де кожен компонент відповідає за свою функцію модуль захоплення даних працює безпосередньо з мережевими інтерфейсами для отримання інформації про всі мережеві з'єднання.

Модуль аналізу проводить детальну перевірку отриманих даних використовуючи алгоритми порівняння вмісту з відомими шаблонами загроз що дозволяє виявляти аномалії в мережевому трафіку модуль фільтрації приймає рішення про блокування підозрілих пакетів або їх пропускання що допомагає запобігати проникненню шкідливих даних у мережу.

Логуючий модуль зберігає інформацію про кожну операцію що дозволяє проводити аналіз ефективності роботи системи та вдосконалювати алгоритми обробки даних результати розрахунків засновані на вимірюванні часу обробки одного пакету та загальної кількості пакетів при симуляції високого навантаження.

Це демонструює стабільну продуктивність системи що є критично важливим для забезпечення безпеки мережі використання мови Python дозволяє інтегрувати систему з іншими мережевими рішеннями та легко розширювати її функціональність завдяки модульній структурі система успішно працює в реальному часі що є важливою перевагою в умовах сучасних кіберзагроз

Функція capture_packet реалізує захоплення пакетів шляхом симуляції отримання даних з мережі

```
def capture_packet():
```

Симуляція формування пакету з інформацією про джерело призначення і вміст.

```
import random
packet = {"src_ip": "192.168.1." + str(random.randint(1,254)),
          "dst_ip": "10.0.0." + str(random.randint(1,254)),
          "payload": "data" + str(random.randint(1000,9999))}
return packet
```

Функція analyze_packet проводить аналіз вмісту пакету з метою виявлення підозрілих шаблонів.

```
def analyze_packet(packet):
```

Використання списку підозрілих ключових слів для перевірки вмісту пакету.

```
suspicious_keywords = ["malware", "attack", "virus"]
for keyword in suspicious_keywords:
    if keyword in packet["payload"]:
        return True
return False
```

Функція filter_packet приймає рішення про блокування або пропуск пакету на основі результатів аналізу.

```
def filter_packet(packet):
```

Виклик функції аналізу пакету для визначення підозрілої активності.

```
is_suspicious = analyze_packet(packet)
if is_suspicious:
    return "block"
return "allow"
```

Функція main запускає систему перевірки пакетів проводячи симуляцію потоку пакетів та розрахунок часу обробки.

Ініціалізація змінних для обрахунку загальної кількості пакетів заблокованих пакетів пропущених пакетів та сумарного часу обробки

```
import time
total_packets = 0
blocked_packets = 0
allowed_packets = 0
```

```
total_time = 0
```

Симуляція обробки потоку з ста пакетів.

```
for i in range(100):
    start_time = time.time()
    packet = capture_packet()
    decision = filter_packet(packet)
    elapsed_time = time.time() - start_time
    total_time += elapsed_time
    total_packets += 1
    if decision == "block":
        blocked_packets += 1
    else:
        allowed_packets += 1
average_time = total_time / total_packets
```

Виведення результатів розрахунків щодо загальної кількості пакетів заблокованих пакетів пропущених пакетів та середнього часу обробки одного пакету.

```
print("Загальна кількість пакетів", total_packets)
print("Заблоковано пакетів", blocked_packets)
print("Пропущено пакетів", allowed_packets)
print("Середній час обробки пакету", average_time)
```

Підсистема керування логами системи перевірки пакетів у мережі забезпечує збереження інформації про кожну операцію аналізу та фільтрації вона працює в режимі реального часу що дозволяє одразу фіксувати дані про час обробки кожного пакету.

Отримані рішення щодо блокування або пропускання даних а також деталі мережевого трафіку система записує інформацію у спеціально створені лог файли з використанням стандартних бібліотек Python що дозволяє зберігати дані у зручному форматі для подальшої обробки.

Накопичені записи використовуються для аналізу продуктивності системи а також для виявлення можливих вразливостей завдяки постійному моніторингу роботи система може виявити аномалічні затримки у обробці пакетів що свідчить про потенційні проблеми у мережевій інфраструктурі накопичлена інформація використовується для аналізу тенденцій у роботі системи.

Вона допомагає визначити закономірності появи підозрілих операцій що дозволяє своєчасно реагувати на збільшення загроз можливість інтеграції з зовнішніми сервісами моніторингу дозволяє проводити аналіз подій у режимі реального часу.

Система формує повідомлення для адміністратора у випадках підвищеної активності що свідчить про можливе порушення безпеки архітектура підсистеми розроблена з урахуванням розділення функціональних задач що забезпечує високу продуктивність основних модулів системи та надійне збереження логів для подальшого аналізу роботи системи та вдосконалення алгоритмів безпеки.

Система оптимізації роботи системи застосовує модуль паралелізації завдань який використовує асинхронне виконання операцій для одночасного аналізу даних що дозволяє ефективно розподіляти навантаження між різними процесами.

Система використовує можливості багатопоточності для розподілу завдань між ядрами процесора що сприяє зниженню часу обробки мережевих даних оптимізація здійснюється за рахунок впровадження асинхронного програмування яке дозволяє виконувати операції незалежно одна від одної без очікування завершення попередніх завдань.

Система аналізує поточне навантаження в режимі реального часу вона динамічно розподіляє ресурси між процесами що сприяє підвищенню ефективності роботи навіть при різкому збільшенні кількості оброблюваних пакетів алгоритми розподілу завдань дозволяють визначити оптимальний спосіб виконання операцій.

Що забезпечує стабільну роботу системи при високій інтенсивності мережевого трафіку розрахунки середнього часу обробки підтверджують можливість системи.

Адаптуватися до умов високого навантаження завдяки застосуванню паралельних потоків система зменшує загальний час обробки кожного пакету що дає змогу досягти високої продуктивності в режимі реального часу.

Система оптимізує використання апаратних ресурсів завдяки моніторингу поточної завантаженості вона в режимі реального часу коригує розподіл завдань що гарантує оперативну реакцію на зміни в інтенсивності потоку даних

4.2 Захист розробленого програмного забезпечення

Для захисту розробленого програмного забезпечення запропоновано використовувати алгоритм Khufu. Khufu – це 64-бітовий блоковий шифр. 64-бітовий відкритий тест спочатку розщеплюється на дві 32-бітові половини, *L* і *R.* Над обома половинами й певними частинами ключа виконується операція XOR. Потім, аналогічно DES, результати проходять деяку послідовність раундів. У кожному раунді молодший значущий байт *L* використовується як вхід S-блоку. У кожного S-блоку 8 вхідних біт і 32 вихідних біта. Далі обраний в S-блоці 32-бітовий елемент піддається операції XOR з *R.* Потім *L* циклічно зрушується на число, кратним восьми биткам, *L* і *R* міняються місцями, і раунд завершується. Сам S-блок не статичний, він міняється кожні вісім раундів. Нарешті, по закінченні останнього раунду, над *L* і *R* виконується операція XOR з іншими частинами ключа, і половини поєднуються, утворюючи блок шифртексту.

Хоча частини ключа використовуються для операції XOR із блоком шифрування на початку й кінці виконання алгоритму, головне призначення ключа – генерація S-блоків. Ці S-блоки секретні, по суті, це частина ключа. Повний розмір ключа алгоритму Khufu дорівнює 512 біт (64 байт), алгоритм надає спосіб генерації S-блоків по ключу.

5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

Розглянемо розроблене ПЗ яке зображено на рисунку 5.1. З рисунку можна побачити що інтерфейс головного вікна розподілено на наступні розділи:

– Меню користувача. В меню користувача відображаються всі дії які можна виконати з програмою.

– Підключення. Обирає мережеву карту яка використовується.

– Функціонал програми: Сканування мережі; Графічна побудова топології мережі; Блокування портів; Задати фільтр IP-адрес; Налаштування фільтру; Виявлення аномалій телекомунікаційного трафіку.

– Статус бар.

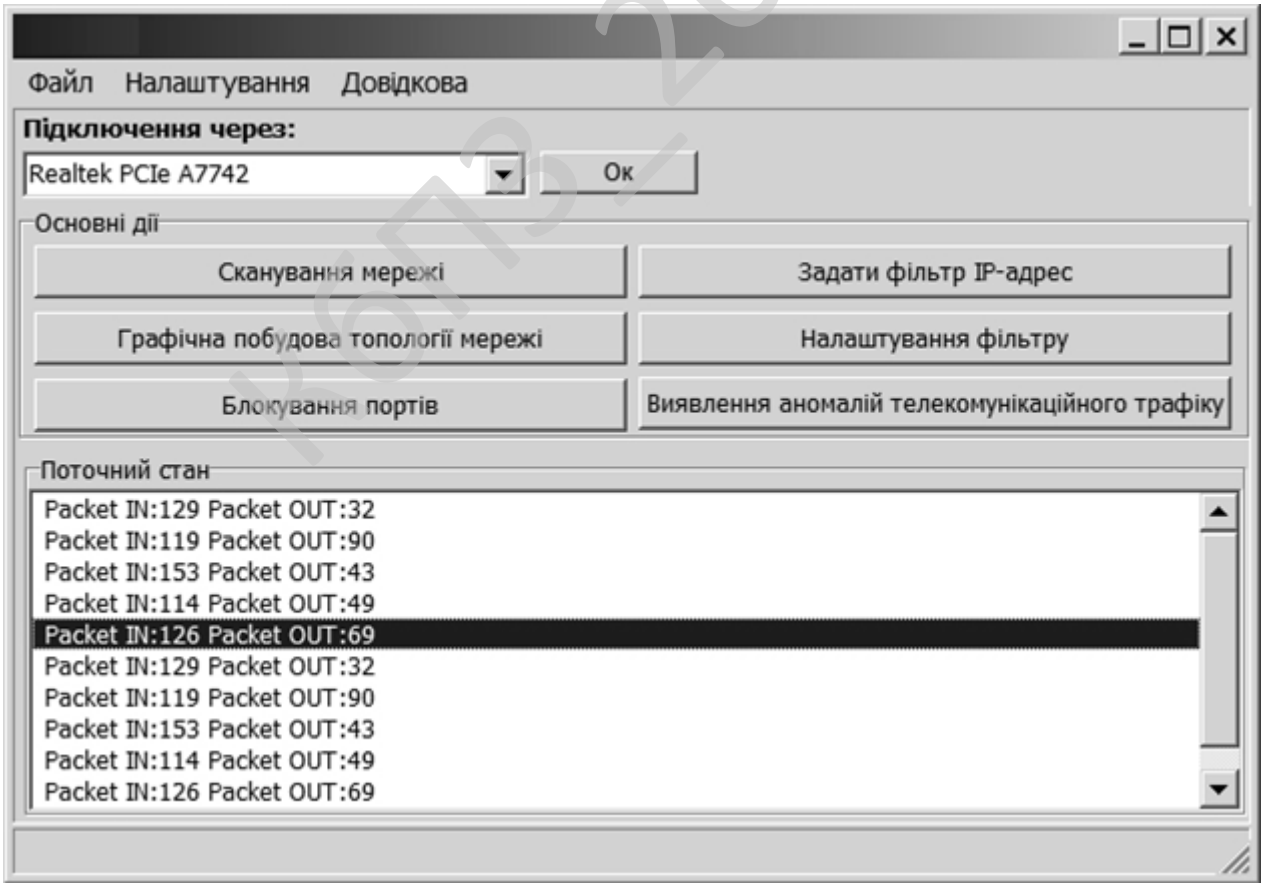


Рисунок 5.1 – Головне вікно програми

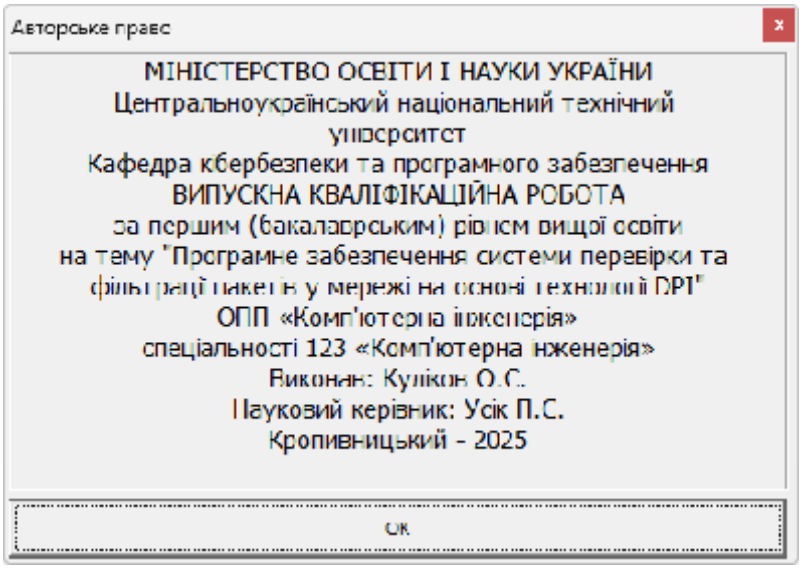


Рисунок 5.2 – Авторське право

На рисунку 5.2 зображено авторські дані розробленого програмного забезпечення.

Обрано умови розповсюдження – proprietary software.

Програмне забезпечення, на яке зберігаються як немайнові, так і майнові авторські права. Отримавши або придбавши таке програмне забезпечення, користувач отримує обмежені права користування ним: може бути заборонено або закрито доступ до коду (вивчення), внесення змін, тиражування, розповсюдження та перепродаж. Програмне забезпечення вважається власницьким, якщо наявне хоча б одне з перелічених обмежень. Найчастіше основним методом захисту майнових прав на власницьке ПЗ, поза ліцензійною угодою, власник обирає закриття сирцевого коду, захищаючи свій продукт від модифікації і вбудовуючи системи обмеження користування через авторизацію. Таке програмне забезпечення називається закритим. Проте, код власницького продукту може бути і відкритим, але власник може обмежити права користувача умовами користувацької ліцензії. Власницьке програмне забезпечення та комерційне програмне забезпечення не є синонімами – власницьким може бути і безплатне (тобто, некомерційне) програмне забезпечення.

6 ОСНОВНІ ВИСНОВКИ

Програмне забезпечення, створене в результаті виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти, призначено для системи перевірки та фільтрації пакетів у мережі на основі технології DPI.

В межах України в недостатній мірі представлені вітчизняні розробки в цій області.

Рішення завдання полягало у вирішенні наступних задач:

– Був проведений огляд існуючих систем перевірки та фільтрації пакетів у мережі на основі технології DPI.

– Досліджена система перевірки та фільтрації пакетів у мережі на основі технології DPI.

– На основі отриманих результатів досліджень створена програмна реалізація системи перевірки та фільтрації пакетів у мережі на основі технології DPI.

Розроблені під час виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти алгоритми дозволяють успішно вирішувати завдання перевірки та фільтрації пакетів у мережі на основі технології DPI.

Розроблене програмне забезпечення має простий, дружній та зручний інтерфейс користувача, що забезпечує легкість у освоєнні роботи програмного продукту, зручність у використанні, і не потребує особливих спеціальних знань.

При створені програмного забезпечення було використано об’єктно-орієнтований підхід, що відповідає сучасним тенденціям у галузі розробки комерційних програмних систем.

Програма реалізована на мові високого рівня Python. Дана мова програмування дозволяє найбільш ефективно обробляти дані призначені для системи перевірки та фільтрації пакетів у мережі на основі технології DPI. Це

дозволило мінімізувати строк розробки програмного забезпечення, і, як слід, зменшити витрати на його розробку. Запропоноване програмне забезпечення ділиться на загальне програмне забезпечення, що поставляється із засобами обчислювальної техніки й спеціальне програмне забезпечення, що спеціально розроблене для даної конкретної системи й включає програми, що реалізують її функції.

Програма призначена для виконання під управлінням багатозадачної операційної системи Windows 10/11.

Даються необхідні рекомендації з установки розробленого програмного забезпечення.

Для підвищення рівня безпеки запропоновано застосовувати алгоритм Khufu.

В цілому створене програмне забезпечення підтверджує правильність використаних проектних рішень та повністю відповідає вимогам технічного завдання. Створене програмне забезпечення має потенційну можливість для подальшого вдосконалення і застосування у різних галузях.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Awais Rashid, Howard Chivers, George Danezis, Emil Lupu, Andrew Martin. CyBOK The Cyber Security Body of Knowledge. The National Cyber Security Centre. 2019. 854 p.

2. Loren Kohnfelder. Designing Secure Software. No Starch Press. 2022. 332 p.

3. Samir Kumar Rakshit. Ethical Hacker's Penetration Testing Guide. BPB Online. 2022. 509 p.

4. Corey J. Ball. Hacking APIs. No Starch Press. 2022. 353 p.

5. Kevin Beaver. Hacking for Dummies. John Wiley & Sons. 2022. 419 p.

6. Mark S. Merkow. Practical Security for Agile and DevOps. CRC Press. 2022. 236 p

7. Derek Fisher. Application Security Program Handbook. Manning Publications. 2021. 155 p.

8. Cameron Wyatt PH.D. Kali Linux Tutorial. Independently published. 2021. 60 p.

9. Alex Matrosov, Eugene Rodionov, Sergey Bratus. Rootkits and Bootkits. No Starch Press. 2019. 450 p.

10. Kuznetsov O., Frontoni E., Kuznetsova Y., Smirnov O., Moskovchenko I. «Trust-Based Security Architecture for Edge Computing: A Simulation Study of Dynamic Trust Evolution and Attack Detection». *CEUR Workshop Proceedings,* 2024, 3909, pp. 227–241.

11. Lakhno, V., Malyukov, V., Smirnov, O., Bebeshko, B., Chubaievskiy, V., Zhumadilova, M., Malyukova, I., Smirnov, S. «Multifactorial Model for Targeted Attacks Counteracting Within the Framework of a Multi-Step Quality Game with Fuzzy Information». *8th International Symposium on Intelligent Informatics, ISI 2023*, 2025. vol 389. pp 377-389. Springer, Singapore.

12. Kuznetsov, O., Frontoni, E., Kryvinska, N., Chevardin, V., Smirnov, O. «Wireless Network Encryption Stream Ciphers, Computational Modeling, and Security Analysis». *Computational Modeling and Simulation of Advanced Wireless Communication Systems*, 2024, pp. 379–402.

13. Kuznetsov, O., Frontoni, E., Kryvinska, N., Smirnov, O., Imoize, G.L. «Computational Modeling of Enhanced Spread Spectrum Codes for Asynchronous Wireless Communication». *Computational Modeling and Simulation of Advanced Wireless Communication Systems*, 2024, pp. 403–447.

14. Kuznetsov, O., Kryvinska, N., Ilchenko, O., Smirnova, T., Ulianovska, Y. «Comparative Analysis of Cryptocurrency Trading Platforms Using the Analytic Hierarchy Process». *CEUR Workshop Proceedings*, 2023, 3628, pp. 106-115.

15. Akhalaia, G., Iavich, M., Iashvili, G., Prysiazhnyy, D., Smirnova, T. «Secure Encrypted Connection on Georgian Website». *CEUR Workshop Proceedings,* 2023, 3550, pp. 313-320.

16. Al-Mudhafar Aqeel, A.M., Smirnova, T., Buravchenko, K., Smirnov, O. «The method of assessing and improving the user experience of subscribers in software-configured networks based on the use of machine learning». *Advanced Information Systems*, 2023, 7(2), pp. 49-56

17. Smirnov, O., Sydorenko, V., Aleksander, M., Zhyharevych, O., Yenchev, S. «Simulation of the cloud IoT-based monitoring system for critical infrastructures». *CEUR Workshop Proceedings,* Volume 3530, 2023, pp. 256-265.

18. Kuznetsov, O., Kandiy, S., Frontoni, E., Smirnov, O. «Trade-offs in Post-Quantum Cryptography: A Comparative Assessment of BIKE, HQC, and Classic McEliece». *CEUR Workshop Proceedings,* Volume 3504, 2023, pp. 1-11.

19. Smirnov, O., Neskorodieva, T., Fedorov, E., Rudakov, K., Neskorodieva, A. «Method Detection Audit Data Anomalies on Basis Restricted Cauchy Machine» *CEUR Workshop Proceedings,* Volume 3187, 2022,

20. Smirnov, O., Lakhno, V., Akhmetov, B., Chubaievskyi, V., Khorolska, K., Bebeshko, B. «Selection of a Rational Composition of İnformation Protection

Means Using a Genetic Algorithm». *In: Rajakumar, G., Du, KL., Vuppalapati, C., Beligiannis, G.N. (eds) Intelligent Communication Technologies and Virtual Mobile Networks. Lecture Notes on Data Engineering and Communications Technologies*, vol 131. 2023. Springer, Singapore. pp. 21-34.

21. Smirnov O.A., Al-Oraiqat A.M., Ulichev O.S., Meleshko Ye.V., Al-Rawashdeh H.S., Polishchuk L.I. «Modeling strategies for information influence dissemination in social networks». *Journal of Ambient Intelligence and Humanized Computing* Volume 13, Issue 5. Springer, Cham. 2022, pp. 2463-2477.

22. Smirnov O., Kuznetsov A., Zhora V., Onikiychuk A., Pieshkova O. «Hiding Messages in Audio Files Using Direct Spread Spectrum». *11th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2021,* Cracow, Poland, 22-25 September 2021. P. 414-418

23. Smirnov O., Kuznetsov A., Lokotkova I., Kuznetsova T., Florov S., Lebid O. «Using Orthogonal Signals to Hide Information in Images». *4 IEEE International Conference on Advanced Information and Communication Technologies (AICT) - 2021*, Lviv, Ukraine, September 21-25, 2021. P. 255-260.

24. Smirnov O., Kuznetsov A., Girzheva O., Kiian A., Nakisko O., Kuznetsova T. «Advanced Code-Based Electronic Digital Signature Scheme». *2020 IEEE International Conference on Problems of Infocommunications Science and Technology, PIC S and T 2020,* Kharkiv, 6 October 2020-9 October 2020, P. 358-362.

25. Smirnov O., Kuznetsov A., Kiian A., Kuznetsova K. «Data hiding scheme based on spread sequence addressing». *CEUR Workshop Proceedings* Volume 2805, 2020, Pages *44-58.*

26. Smirnov, O., Kuznetsov, A., Potii, O., Poluyanenko, N., Stelnyk, I., Mialkovsky, D. «Combining and filtering functions in the framework of nonlinear-feedback shift register». *International Journal of Computing*; 2020, Volume 19, Issue 2 – Research Institute for Intelligent Computer Systems – 2020. – P. 247-256.

27. Smirnov O., Kuznetsov A., Kiian A., Kuznetsova T. «Non-binary constant weight coding technique». *CEUR Workshop Proceedings*. Volume 2740, 2020, Pages 102-114.

28. Smirnov O., Alimseitova Zh., Adranova A., Akhmetov B., Lakhno V., Zhilkishbayeva G. «Models and algorithms for ensuring functional stability and cybersecurity of virtual cloud resources». *Journal of theoretical and applied information technology* Vol.98. No 21, 2020, P. 3334-3346.

29. Smirnov O., Kuznetsov A., Arischenko A., Chepurko I., Onikiychuk A., Kuznetsova T. «Pseudorandom sequences for spread spectrum image steganography». *CEUR Workshop Proceedings* Volume 2654, 2020, Pages 122-131.

30. Smirnov O., Kuznetsov A., Kovalchuk D., Kuznetsova T. «New technique for data hiding in cover images using adaptively generated pseudorandom sequences». *CEUR Workshop Proceedings* Volume 2654, 2020, Pages 1-14.

31. Smirnov O., Lutsenko M., Kuznetsov A., Kiian A., Kuznetsova T., «Biometric cryptosystems: overview, state-of-the-art and perspective directions». *Lecture Notes in Networks and Systems*, vol 152. Springer, Cham. 2021, pp 66-84.

32. Smirnov O., Kuznetsov A., Onikiychuk A., Makushenko T., Anisimova O., Arischenko A. «Adaptive pseudo-random sequence generation for spread spectrum image steganography». *2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT)*, Ukraine, Kyiv, May 14-18. 2020. P. 161-165.

33. Smirnov O., Kuznetsov A., Kiian A., Babenko V., Perevozova I.,Chepurko I. «New Approach to the Implementation of Post-Quantum Digital Signature Scheme». *2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT),* Ukraine, Kyiv, May 14-18. 2020. P. 166-171.

34. Smirnov O., Kuznetsov A., Kiian A., Cherep A., Kanabekova M., Chepurko I. «Testing of code-based pseudorandom number generators for post-quantum application». *2020 IEEE 11th International Conference on Dependable*

Systems, Services and Technologies (DESSERT), Ukraine, Kyiv, May 14-18. 2020. P. 172-177.

35. Smirnov O., Kuznetsov A., Pushkar'ov A., Serhiienko R., Babenko V., Kuznetsova T., «Representation of Cascade Codes in the Frequency Domain». In: Radivilova T., Ageyev D., Kryvinska N. (eds) Data-Centric Business and Applications. Lecture Notes on Data Engineering and Communications Technologies, vol 48. Springer, Cham. 2021. pp 557-587.

36. Smirnov, O., Markovets, O. Vovk, N., Turchyn, Y., «Model of informational support for social network administrators' content creation». *CEUR Workshop Proceedings* Volume 2616, 2020, Pages 125-136.

37. Smirnov, O., Shekhanin, K., Kuznetsov, A., Krasnobayev, V. «Detecting Hidden Information in FAT». *International Journal of Computer Network and Information Security (IJCNIS)*. Vol. 12, No. 3, 2020. PP.33-43.

38. Smirnov, O., Kuznetsov, A., Gorbacheva, L., Babenko, V., «Hiding data in images using a pseudo-random sequence», *CEUR Workshop Proceedings* Volume 2608, 2020, Pages 646-660.

39. Smirnov, O., Kuznetsov, A., Kolovanova, I., Kuznetsova, T., «Noise immunity of the algebraic geometric codes». International Journal of Computing; 2019, Volume 18, Issue 4 – Research Institute for Intelligent Computer Systems – 2019. – P. 393-407.

40. Smirnov, O., Kuznetsov, A., Reshetniak, O., Ivko, N., Katkova, T., Kuznetsova, T., «Generators of Pseudorandom Sequence with Multilevel Function of Correlation». *2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T),* Kyiv, Ukraine, 8 – 11 October 2019 . P.517-522.

41. Smirnov, O., Ulichev, O., Meleshko, Y., Khokh, V., Goncharenko, I. «Method of Choosing Objects for Informational Influence in Social Networks during Information Campaign Based on the Analytic Hierarchy Process». *CEUR Workshop Proceedings*, Vol 2588, P. 215-227, 2019.

42. Smirnov, O., Krasnobayev, V., Yanko, A., Kuznetsova, T. «Methods of nulling numbers in the system of residual classes». *CEUR Workshop Proceedings*, Vol 2588, P. 90-106, 2019.

43. Smirnov, O., Kuznetsov, A., Kiian, A., Gorbenko, Y., Cherep, O., Bexhter L. «Code-based Pseudorandom Generator for the Post-Quantum Period», *2019 IEEE International Conference on Advanced Trends in Information Theory (IEEE ATIT 2019).* 18.12.19-20.12.19 Kyiv Ukraine. P. 204 – 209.

44. Smirnov, O., Kuznetsov, A., Nariezhnii, O., Stelnyk, S., Kokhanovska, T., Kuznetsova T., «Side Channel Attack on a Quantum Random Number Generator», *10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2019*; Metz; France; 18 - 21 September 2019. P.713-718.

45. Kuznetsova, T., «Code-Based Schemes for Post-Quantum Digital Signatures», *10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2019;* Metz; France; 18-21 September 2019. P. 707-712.

46. Smirnov, O., Kuznetsov, A., Stefanovych, O., Gorbenko, Y., Krasnobaev, V., Kuznetsova K. «Information Hiding Using 3D-Printing Technology», *10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2019*; Metz; France; 18-21 September 2019. P.701-706.

47. Smirnov, O., Hu, Z., Vasiliu, Y., Sydorenko, V., Polishchuk, Y., «Abstract Model of Eavesdropper and Overview on Attacks in Quantum Cryptography Systems», *10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2019*; Metz; France; 18-21 September 2019. P.399-405.

48. Smirnov, O., Kuznetsov, A., Kovalchuk, D., Averchev, A., Pastukhov, M., Kuznetsova, K., «Formation of Pseudorandom Sequences with Special Correlation

Properties», *2019 3rd International Conference on Advanced Information and Communications Technologies, AICT -2019*/ Lviv, Ukraine, 2-6 July, 2019, P. 395-399.

49. Smirnov, O., Kuznetsov, A., Kiian, A., Babenko, B., Zhosan, H., Prokopovych-Tkachenko, D., «Soft Decoding Method for Turbo-Productive Codes», *2019 3rd International Conference on Advanced Information and Communications Technologies, AICT 2019*, Lviv, Ukraine, 2-6 July, 2019, P. 129-134.

50. Smirnov, O., Kuznetsov, A., Kiian, A., Zamula, A., Rudenko, S., Hryhorenko, V., «Variance Analysis of Networks Traffic for Intrusion Detection in Smart Grids», *2019 IEEE 6th International Conference On Energy Smart Systems (2019 IEEE ESS)*, Kyiv, Ukraine April 17-19, 2019 P. 353-358.

51. Smirnov, O., Kuznetsov, A., Kavun, S., Babenko, B., Nakisko, O., Kuznetsova, K., «Malware Correlation Monitoring in Computer Networks of Promising Smart Grids», *2019 IEEE 6th International Conference On Energy Smart Systems (2019 IEEE ESS)*, Kyiv, Ukraine April 17-19, 2019 P. 347-352.

Додаток А
(обов’язковий)

Технічне завдання

Зміст

1 Найменування та область застосування

Це технічне завдання розповсюджується на розробку системи перевірки та фільтрації пакетів у мережі на основі технології DPI.

2 Підстава для розробки

Підставою для розробки служить завдання на випускну кваліфікаційну роботу за першим (бакалаврським) рівнем вищої освіти, видане на кафедрі кібербезпеки та програмного забезпечення (нак. № 46-02 від 17.01.2025 року).

3 Мета та призначення розробки

Метою випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти є розробка програмного забезпечення системи перевірки та фільтрації пакетів у мережі на основі технології DPI.

4 Джерела розробки

Джерелом цієї випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти є стосовна до теми література і існуючі аналоги.

5 Технічні вимоги

5.1 Склад продукції

Складниками розробки є:

– вибір і обґрунтування методів реалізації проекту;

– розробка програмної частин системи, а також розробка взаємодії системи з ОС та з користувачем;

– розробка програми, що реалізує спроектовані алгоритми роботи системи.

5.2 Показники призначення

Система повинна забезпечувати:

– системи перевірки та фільтрації пакетів у мережі на основі технології DPI;

– цілісність даних у процесі роботи та при зберіганні;

– простий, інтуїтивно зрозумілий інтерфейс.

5.3 Вимоги до функціональних характеристик

Розроблене програмне забезпечення не повинно мати обмежень на версію драйверів та операційної системи.

5.4 Вимоги до архітектури

Компонент, що розробляється повинен використовувати системні засоби та апаратні засоби, що на даному етапі розвитку обчислювальної техніки найбільше поширені.

5.5 Вимоги до надійності

Програмні модулі написані по всім правилам, які стосуються стандартних викликів процедур, функцій, методів і форм, визначених технічною документацією на середовище розробки.

5.6 Умови експлуатації

Робочі місця користувачів ПЗ повинні задовольняти наступним умовам експлуатації:

– температура повітря: 19-20 град. по Цельсію;

– відносна вологість повітря до 80%;

– атмосферний тиск 107 кПа.

5.7 Вимоги до складу та параметрів технічних засобів

Програмне забезпечення повинно бути реалізоване на ПЕОМ, працювати в ОС Windows 10/11 і з сумісними з цією платформою пристроями і прикладним програмним забезпеченням.

5.8 Вимоги до інформаційної і програмної сумісності

Переносність програмного забезпечення повинна бути забезпечена за рахунок його реалізації стандартного інтерфейсу взаємодії з ОС, що працюють під управлінням ОС Windows 10/11.

5.8.1 Обладнання

Комп’ютер Intel® Celeron/8 Mb/1.2 Gb/SVGA 14” 1Mb або сумісні з ним.

5.8.2 Мова програмування

Середовище Python.

5.8.3 Вхідні дані

Опис алгоритму роботи запропонованої системи.

5.8.4 Вихідні дані

Робоча програма.

6 Вимоги до програмної документації

Програмна продукція повинна бути представлена у виді опису структури даних, схем та опису алгоритму, а також текстів вихідних модулів програмного забезпечення згідно ЄСПД .

7 Перелік документів, що розробляються

- Структурна схема системи – 1 аркуш.
- Функціональна схема системи – 1 аркуш.
- Діаграма процесів – 1 аркуш.
- Блок-схема алгоритму роботи програми – 2 аркуша.
- Пояснювальна записка – 61 аркуш.

8 Етапи розробки

8.1 Збір і обробка інформації по темі випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти. Постановка задачі на виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти (складання ТЗ).

8.2 Проведення досліджень або експериментальних робіт для уточнення основних положень випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти.

8.3 Розробка функціональних схем, блок схем алгоритмів роботи програмного забезпечення.

8.4 Побудова схем взаємодії даних.

8.5 Створення прототипу ПЗ.

8.6 Віднаходження ПЗ, аналіз отриманих результатів.

8.7 Оформлення пояснювальної записки і виконання робіт по графічній частині.

9 Порядок контролю та приймання

9.1 Подання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти на попередній захист 23.05.2025 р.

9.2 Подання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти на захист 6.06.2025 р.