

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ НАЦІОНАЛЬНИЙ АГРАРНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ ЕКОНОМІКИ І МЕНЕДЖМЕНТУ**

Кафедра кібернетики та інформатики

КВАЛІФІКАЦІЙНА РОБОТА

освітній ступінь - «Бакалавр»

**на тему: “РОЗРОБКА ВЕБЗАСТОСУНКУ ДЛЯ
МОНІТОРИНГУ ТРАФІКУ ІНТЕРНЕТ-ПРОВАЙДЕРА”**

Виконав:

студент спеціальності

126 «Інформаційні системи та технології»

ОП «Інформаційні системи та технології»

Корецький Василь Андрійович

Керівник:

Долгіх Яна Володимирівна

доцент кафедри кібернетики та інформатики

Рецензент:

Коломієць Світлана Володимирівна

старший викладач кафедри економічної кібернетики
Сумського державного університету

Суми – 2024

СУМСЬКИЙ НАЦІОНАЛЬНИЙ АГРАРНИЙ УНІВЕРСИТЕТ

Факультет	Економіки і менеджменту
Кафедра	Кібернетики та інформатики
Освітній ступінь	«Бакалавр»
Спеціальність	126 «Інформаційні системи та технології» ОП «Інформаційні системи та технології»

Затверджую:

Завідувач кафедри _____

« _____ » _____ 2024 р.

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ *Корецькому Василю Андрійовичу*

1. Тема роботи: “Розробка вебзастосунку для моніторингу інтернет-трафіку провайдера”

2. База практичного дослідження: ФОП «Єрмоєнко Дмитро Сергійович», м. Суми.

керівник роботи Долгих Яна Володимирівна, к.е.н., доцент

затверджена наказом по університету від « _____ » _____ 2024р. № _____ -н

3. Строк подання студентом закінченого проекту (роботи) _____

4. Вихідні дані до проекту (роботи):

результати аналітичних досліджень зарубіжних та вітчизняних вчених, інтернет-джерела

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

- виконати постановку задачі;
- вибрати середовище для реалізації;
- розробити технічне завдання;
- розробити програмний продукт відповідно до поставлених задач;
- розробити інструкцію для користувача

6. Дата видачі завдання: « _____ » _____ 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту	Примітка
1	Визначення об'єкту, предмету дослідження, формулювання мети та задач кваліфікаційної роботи, складання плану	січень 2024 р.	
2	Підбір та вивчення літературних джерел, технічної документації	січень 2024 р.	
3	Розробка технічного завдання	січень 2024 р.	
4	Розробка вебсайту	січень-травень 2024 р.	
5	Оформлення пояснювальної записки	до 30 травня 2024 р.	
6	Оформлення презентації	до 1 червня 2024 р.	
7	Опрацювання зауважень керівника, перевірка на автентичність	2 червня 2024 р.	
8	Підготовка доповіді та її попередній захист	6 червня 2024 р.	
9	Перевірка з академічної доброчесності кваліфікаційної роботи	10 червня 2024 р.	
10	Представлення кваліфікаційної роботи на кафедрі, в деканат	23 червня 2024 р.	
11	Захист кваліфікаційної роботи	28 червня 2024р.	

Студент

_____ (підпис)

Василь КОРЕЦЬКИЙ

Керівник роботи

_____ (підпис)

Яна ДОЛГІХ

Перевірку на автентичність проведено

_____ (підпис)

Кваліфікаційна робота допущена до захисту

_____ (підпис)

АНОТАЦІЯ

Корецький В.А. Розробка вебзастосунку для моніторингу інтернет-трафіку провайдера

Кваліфікаційна робота зі спеціальності 126 «Інформаційні системи та технології» ОП Інформаційні системи та технології, СНАУ, Суми-2024р. – Рукопис.

Кваліфікаційна робота присвячена розробці вебзастосунку для моніторингу інтернет-трафіку провайдера.

Метою роботи є створення ефективного інструменту, що допоможе забезпечити моніторинг трафіку провайдера в реальному часі

У роботі проведений аналіз аналогів для моніторингу трафіку. Запропоновано концептуальну модель вебзастосунку для моніторингу статистики використання трафіку процесами на обраному інтерфейсі.

Для розробки вебзастосунку було використано сучасну мову програмування та бібліотеки Python, такі як Pandas та Scapy. Середовищем програмування було обрано PyCharm, забезпечуює швидке та надійне функціонування.

Проведено тестування вебзастосунку з використанням реальних даних. Результати тестування підтвердили правильність функціонування вебзастосунку та відповідність вимогам та очікуванням користувачів.

Отримані результати дозволяють зробити висновок про доцільність впровадження розробленого вебзастосунку для моніторингу трафіка. Вебзастосунок сприятиме покращенню ефективності використання трафіку, а також забезпечить зручний доступ до необхідної інформації.

Ключові слова: Python, Npcap, NDIS, API, NPF, вебзастосунок, моніторинг.

ABSTRACT

Koretskyi V.A. Development of a Web Application for Monitoring ISP Internet Traffic

Qualification Work for Specialty 126 "Information Systems and Technologies" under the Information Systems and Technologies Program, SNAU, Sumy, 2024 – Manuscript.

This work is dedicated to the development of a web application for monitoring the internet traffic of an ISP. The main goal of the work is to create an effective tool that provides real-time monitoring of ISP traffic.

The work includes an analysis of existing traffic monitoring solutions. A conceptual model of a web application is proposed for monitoring traffic usage statistics by processes on a selected interface.

The development of the web application utilized a modern programming language and Python libraries such as Pandas and Scapy. The development environment chosen was PyCharm, which ensures fast and reliable functionality.

Testing of the web application was conducted using real data. The testing results confirmed the correct operation of the web application and its compliance with user requirements and expectations.

The results obtained allow concluding that the developed web application for traffic monitoring is feasible for implementation. The web application will contribute to improving the efficiency of traffic usage and provide convenient access to necessary information.

Keywords: Python, Npcap, NDIS, API, NPF, web application, monitoring.

ЗМІСТ

ВСТУП

РОЗДІЛ 1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд існуючих рішень.....	10
1.2 Аналоги готових продуктів.....	11
1.2.1 Wireshark	11
1.2.2 Zabbix.....	13
1.2.3 Nagios.....	14
1.2.4 Cisco Stealthwatch.....	16
1.2.5 SolarWinds Network Performance Monitor.....	17
1.2.6 Cacti.....	19
1.3 Постановка задачі.....	21

РОЗДІЛ 2 ВИБІР ПРОГРАМНИХ ЗАСОБІВ ТА ПРОЕКТУВАННЯ ВЕБЗАСТОСУНКУ ДЛЯ МОНІТОРИНГУ ТРАФІКУ ІНТЕРНЕТ-ПРОВАЙДЕРА

2.1 Технологія захоплення трафіку прсар.....	22
2.2 Принцип роботи прсар.....	25
2.3 Scapy.....	29
2.4 Pandas.....	31
2.5 Psutil.....	32
2.6 Tkinter.....	33
2.7 Засоби розробки програмного забезпечення	34

РОЗДІЛ 3 ПРОЕКТУВАННЯ ТА РЕАЛІЗАЦІЯ ВЕБЗАСТОСУНКУ ДЛЯ МОНІТРОНИНГУ ТРАФІКУ ІНТЕРНЕТ-ПРОВАЙДЕРА

3.1 Проектування вебзастосунку.....	42
3.2 Опис функціоналу вебзастосунку.....	46
3.3 Опис програмного забезпечення.....	54

ВИСНОВКИ.....	60
---------------	----

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	61
---------------------------------	----

ДОДАТКИ

Додаток А ТЕХНІЧНЕ ЗАВДАННЯ	64
Додаток Б ПЛАНУВАННЯ РОБІТ	79
Додаток В КОПІЇ ПУБЛІКАЦІЙ	74
Додаток Г ЛІСТИНГ ПРОГРАМ.....	82

ВСТУП

Моніторинг та аналіз інтернет-трафіку є надзвичайно важливою частиною сучасного управління мережами та інформаційною безпекою, забезпечуючи безперебійну роботу, захист від загроз та підтримку стратегічного розвитку організацій.

Моніторинг трафіку допомагає виявляти підозрілу активність та кіберзагрози, такі як DDoS-атаки, зломи, шкідливе ПЗ та витoki даних, дозволяє визначити, які сервіси та додатки використовують найбільше ресурсів, що допомагає оптимізувати їх роботу.

Завдяки аналізу трафіку можна швидко реагувати на загрози та вжити заходів для їх нейтралізації, дозволяє виявляти та блокувати несанкціонований доступ до мережі та ресурсів компанії, допомагає виявити проблеми з пропускнуою здатністю та інші технічні проблеми, що можуть впливати на продуктивність мережі. Отже дослідження існуючих методів перехоплення пакетів та аналізу отриманих даних та розробка сучасних вебзастосунків для моніторингу трафіку інтернет-провайдера є актуальним.

Метою роботи є розробка вебзастосунку для моніторингу трафіку інтернет-провайдера локальної комп'ютерної мережі.

Об'єкт дослідження – процес розробки вебзастосунку для моніторингу трафіку інтернет-провайдера локальної комп'ютерної мережі.

Предметом дослідження є інформаційні технології розробки вебзастосунку для моніторингу трафіку інтернет-провайдера локальної комп'ютерної мережі.

Задачі дослідження:

- аналіз існуючих рішень на ринку та визначення основних принципів функціонування та розробки вебзастосунків для моніторингу трафіку інтернет-провайдера;
- розробка архітектури вебзастосунку для моніторингу трафіку інтернет-провайдера;

- реалізація створеної архітектури вебзастосунку для моніторингу трафіку інтернет-провайдера з використанням сучасних інформаційних технологій;
- тестування та оцінка ефективності розробленого рішення.

Для досягнення мети використані такі методи дослідження:

- 1) логіко-аналітичні – для визначення мети дослідження, засобів її досягнення, обґрунтування результатів, формулювання висновків;
- 2) теоретичні – для аналізу наукових джерел з теми дослідження;
- 3) принципи системного аналізу при дослідженні предметної області.

Дослідження наведено в трьох розділах кваліфікаційної роботи. У першому розділі проведено аналіз існуючих рішень та аналогів готових продуктів для моніторингу трафіку. У другому розділі було обрано програмні засоби для реалізації вебзастосунку для моніторингу трафіку. У третьому виконана розробка вебзастосунку для моніторингу трафіку інтернет-провайдера.

Апробація матеріалів дослідження здійснювалася на міжнародній науково-практичній конференції “Технології XXI сторіччя”, 20-22 листопада 2023 р., Суми: СНАУ [1] та щорічній науково-практичній конференції викладачів, аспірантів та студентів Сумського національного аграрного університету, 14-16 травня 2024 р., Суми: СНАУ [2].

РОЗДІЛ 1

АНАЛІТИЧНИЙ ОГЛЯД

1.1 Огляд існуючих рішень

Моніторинг та аналіз інтернет-трафіку включає захоплення мережових пакетів даних, їх аналіз та інтерпретацію. Необхідно зрозуміти поведінку мережі, виявити проблеми, оптимізувати продуктивність та забезпечити безпеку.

Збір даних включає захоплення мережевого трафіку для подальшого аналізу. Використовується пасивний та активний моніторинг. Системи пасивного моніторингу захоплюють копії мережевого трафіку без втручання в трафік. Це може бути здійснено за допомогою network taps або port mirroring, коли комутатор копіює трафік на спеціальний моніторинговий порт. Системи активного моніторингу генерують трафік для тестування продуктивності мережі та виявлення проблем. Це можуть бути пінг-запити, трасування маршруту або тестові завантаження [3].

Аналіз трафіку для виявлення шаблонів, аномалій або конкретних проблем. Аналіз трафіку може здійснюватися через фільтрацію та декодування для виділення корисної інформації, такої як IP-адреси, порти, типи протоколів, тощо. інспекція на рівні пакетів: Деякі системи виконують глибокий аналіз кожного пакета, перевіряючи його вміст і структуру. Інші системи можуть агрегувати дані для отримання загальної картини трафіку, наприклад, обчислення загального обсягу трафіку, визначення найбільш активних хостів або додатків [4].

Інтерпретація та звітність включає:

- візуалізація: дані представляються у вигляді графіків, таблиць, діаграм, що дозволяє легко інтерпретувати інформацію;
- сповіщення: системи можуть налаштовувати сповіщення для інформування адміністраторів про важливі події або аномалії в реальному часі;

- звіти: генерація періодичних звітів, які підсумовують стан мережі, виявлені проблеми та надані рекомендації.

Реагування та оптимізація включає:

- на основі отриманих даних адміністратори можуть виконувати необхідні дії для оптимізації роботи мережі або усунення виявлених проблем;
- налаштування обладнання: внесення змін до конфігурації мережевого обладнання для підвищення продуктивності або безпеки;
- політики безпеки: оновлення політик безпеки, наприклад, блокування підозрілих IP-адрес або обмеження доступу до певних ресурсів;
- планування ресурсів: оптимізація використання мережевих ресурсів, наприклад, розподіл пропускну здатності для критично важливих додатків.

Для реалізації цих функцій системи моніторингу та аналізу інтернет-трафіку використовують різні технології та методи [5]:

- Sniffers: програми, які захоплюють мережевий трафік для подальшого аналізу;
- NetFlow/SFlow/IPFIX: протоколи для збору інформації про трафік на рівні мережевих пристроїв;
- DPI (Deep Packet Inspection): технологія глибокого аналізу пакетів, яка дозволяє переглядати не тільки заголовки, але й вміст пакетів.

Системи моніторингу та аналізу інтернет-трафіку є важливим інструментом для забезпечення безпеки, оптимізації та надійної роботи мережі.

1.2 Аналоги готових продуктів

1.2.1 Wireshark

Wireshark це популярний інструмент для аналізу мережевого трафіку, який використовується для захоплення і детального аналізу пакетів даних. Хоча існують інші спеціалізовані інструменти для моніторингу та аналізу, Wireshark залишається

одним з найбільш використовуваних завдяки своїм широким можливостям і безкоштовній ліцензії.

Wireshark дозволяє захоплювати мережевий трафік в режимі реального часу з інтерфейсів на локальному комп'ютері або зберігати ці дані для подальшого аналізу та підтримує різні типи мережевих інтерфейсів, включаючи Ethernet, Wi-Fi, Bluetooth і багато інших. Вона підтримує тисячі різних протоколів, включаючи HTTP, FTP, TCP, UDP, DNS, SSL/TLS і багато інших, автоматично визначає протоколи і показує структуру та вміст пакетів у зручному для користувача вигляді [6].

Існує можливість роботи в графічному інтерфейсі (див. рисунок 1.1), що дозволяє використовувати весь інструментарій безпосередньо на робочому місці. Wireshark має інтуїтивно зрозумілий графічний інтерфейс, який дозволяє переглядати захоплені дані в різних форматах, включаючи таблиці, графіки та діаграми підтримує кольорове виділення для кращого візуального розрізнення різних типів трафіку [7].

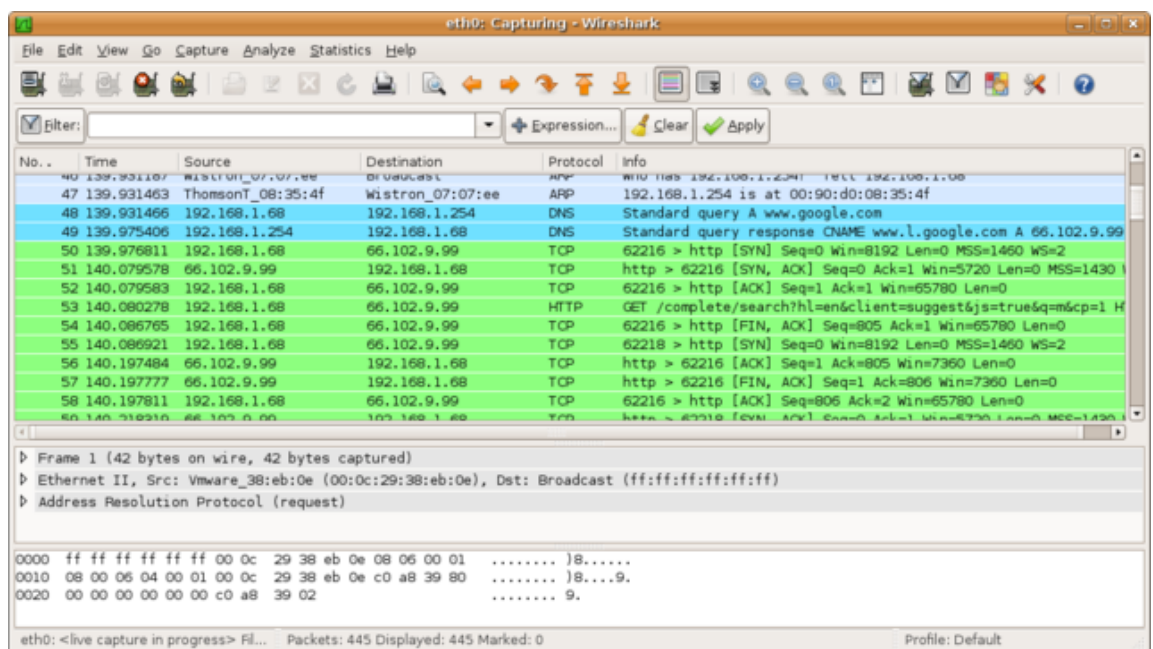


Рисунок 1.1 – Приклад графічного інтерфейсу програми Wireshark

«Wireshark»

Джерело [14]

1.2.2 Zabbix

Zabbix – це потужна система моніторингу мережі та додатків з відкритим вихідним кодом. Вона дозволяє відстежувати стан серверів, мережевих пристроїв, віртуальних машин, баз даних і багатьох інших систем у реальному часі, надаючи адміністратору детальну інформацію про їх стан та продуктивність.

Основні Особливості Zabbix [8]:

- Zabbix підтримує моніторинг серверів, мережевих пристроїв, додатків, баз даних та інших ІТ-ресурсів та відстежує продуктивність і доступність систем, забезпечуючи повний контроль над ІТ-інфраструктурою;
- ця система підтримує різні методи збору даних, включаючи SNMP, IPMI, JMX, агентів Zabbix, а також збір даних через HTTP, SSH і Telnet. Також є можливість створення користувацьких шаблонів і сценаріїв для спеціалізованого моніторингу;
- Zabbix може масштабуватися для моніторингу великих і складних ІТ-інфраструктур з тисячами пристроїв і параметрів, підтримує розподілений моніторинг з використанням проксі-серверів для зменшення навантаження на центральний сервер;
- Zabbix має можливість налаштування сповіщень через електронну пошту, SMS, Telegram, Slack та інші канали. Підтримка умов сповіщення для різних подій і тригерів, включаючи складні логічні вирази;
- має вбудовані засоби для створення графіків, діаграм, карт мережі та панелей моніторингу (див. рисунок 1.2), може генерувати детальні звіти про стан та продуктивність систем.

Переваги Zabbix:

- Zabbix є безкоштовним, і його код доступний для зміни та покращення;
- можливість адаптації до різних сценаріїв моніторингу;
- підходить для моніторингу як малих, так і великих ІТ-інфраструктур;
- підтримка інтеграції з іншими системами та сервісами.

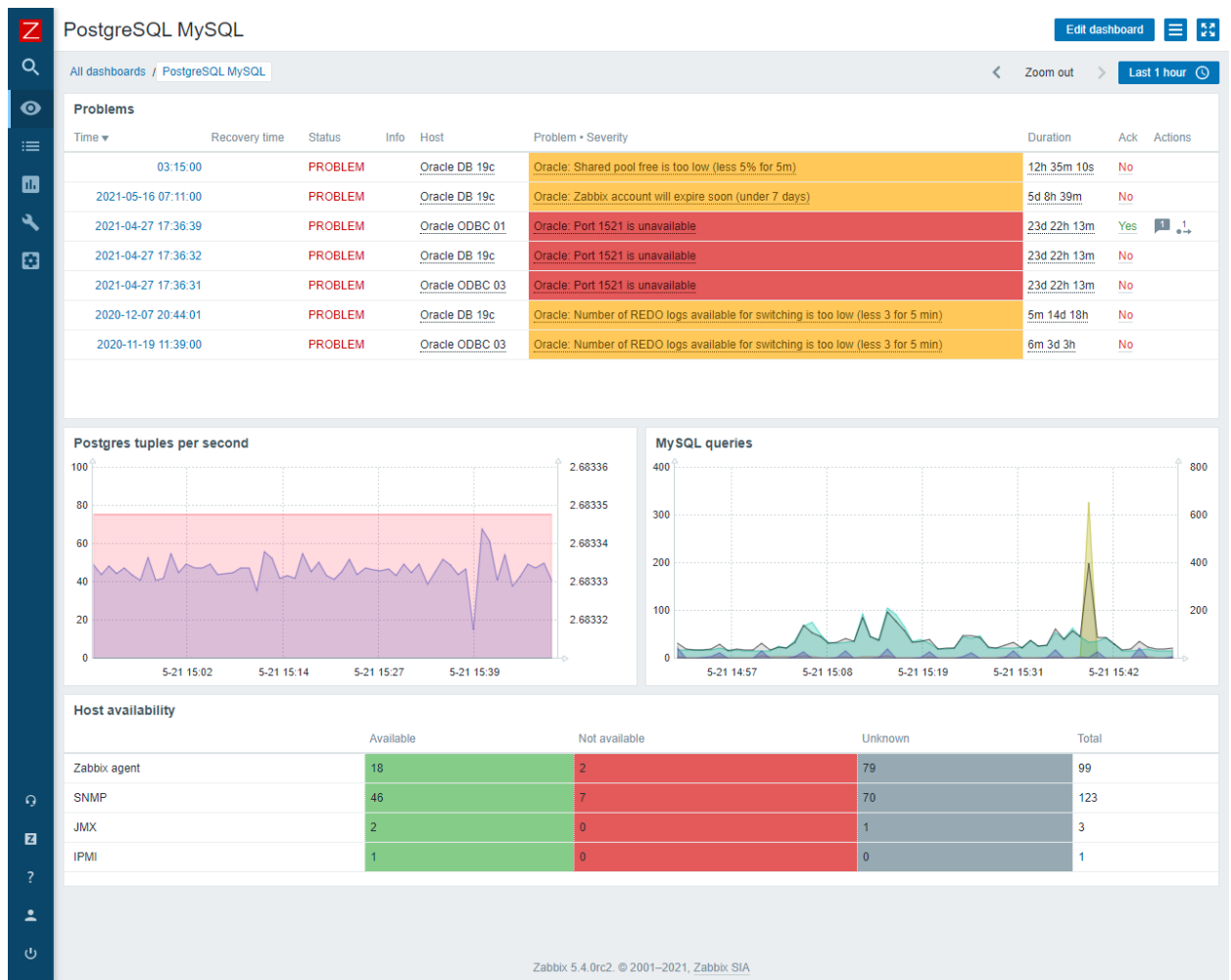


Рисунок 1.2 – Приклад графічного інтерфейсу програми Zabbix
«Zabbix»

Джерело [15]

1.2.3 Nagios

Nagios – це одна з найпопулярніших систем для моніторингу мережевих ресурсів та сервісів. Вона надає можливість відстежувати стан і продуктивність різних компонентів IT-інфраструктури, виявляти проблеми та повідомляти адміністратора про потенційні збої [9].

Основні особливості Nagios:

- ця програма відстежує доступність і продуктивність серверів, комутаторів, маршрутизаторів, баз даних, додатків і багато іншого, та підтримує різні протоколи моніторингу, включаючи HTTP, SMTP, POP3, FTP, SSH і SNMP;
- Nagios використовує плагіни для розширення функціоналу та додавання підтримки нових типів ресурсів і сервісів, має можливість створення користувацьких плагінів для спеціалізованого моніторингу;
- має налаштування сповіщень через електронну пошту, SMS, повідомлення у месенджерах і багато іншого, підтримку ескалації сповіщень для забезпечення своєчасного реагування на проблеми;
- Nagios має веб-інтерфейс для зручного доступу до інформації про стан системи, перегляду звітів і налаштування моніторингу (див рисунок 1.3). Він підтримує різні теми та налаштування для індивідуальних потреб користувача;
- Nagios має можливість моніторингу великих і складних інфраструктур з використанням розподілених серверів Nagios, може зменшення навантаження на центральний сервер за рахунок використання пасивного моніторингу.

Переваги Nagios:

- вільний доступ до коду, що дозволяє налаштовувати його під потреби організації;
- можливість моніторингу практично будь-яких систем та сервісів;
- велика база користувачів і розробників, яка сприяє швидкому розвитку та підтримці продукту.

Недоліки Nagios:

- для початківців може бути складно налаштувати і підтримувати;
- за замовчуванням веб-інтерфейс може здаватися застарілим і незручним без додаткових розширень.

Nagios є ефективним інструментом для моніторингу IT-інфраструктури, завдяки своїй гнучкості та багатій екосистемі плагінів, але його налаштування може вимагати значних технічних зусиль.

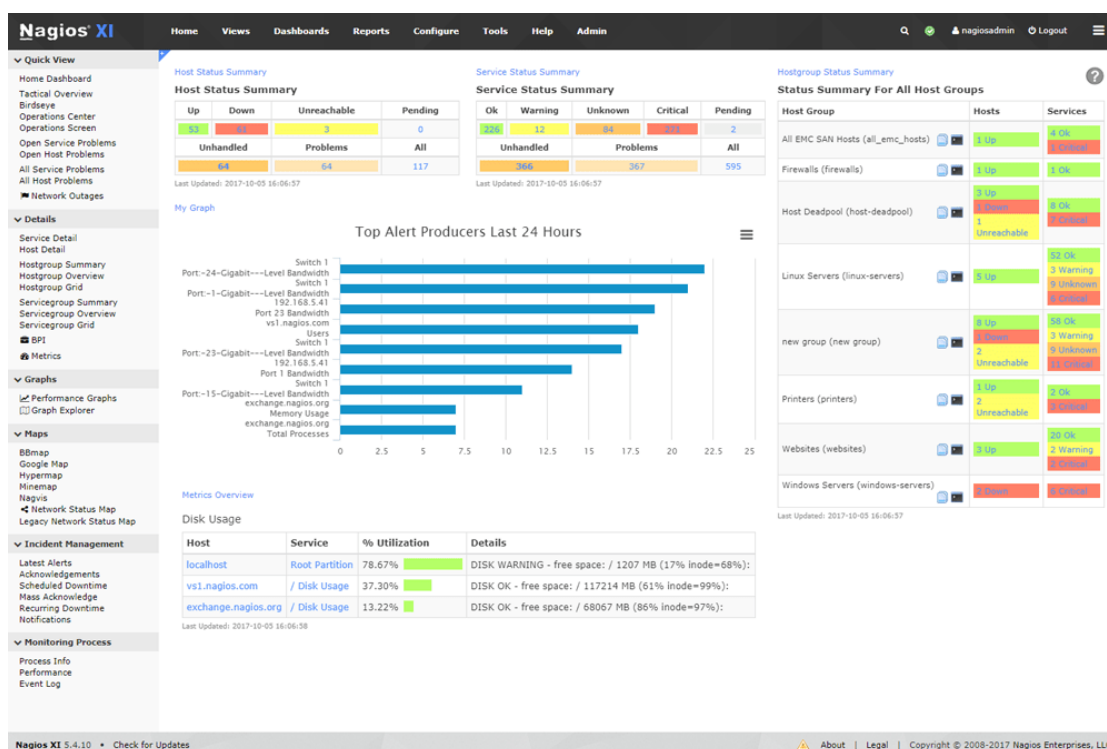


Рисунок 1.3 – Приклад графічного інтерфейсу програми Nagios

«Nagios»

Джерело: [16]

1.2.4 Cisco Stealthwatch

Cisco Stealthwatch – це комплексна система мережевої безпеки, яка використовує поведінковий аналіз для виявлення та реагування на загрози в мережі. Вона забезпечує видимість трафіку, автоматизує виявлення аномалій і надає інструменти для розслідування інцидентів, що дозволяє швидко виявляти та реагувати на кібератаки.

Основні функції та можливості Cisco Stealthwatch:

- збирання та аналіз мережевого трафіку з використанням NetFlow, IPFIX, та інших протоколів;
- підтримка аналізу на рівнях від мережевого трафіку до поведінки додатків;
- надання інтерфейсів для візуалізації потоку трафіку і виявлення аномалій (див. рисунок 1.4);

- використання алгоритмів машинного навчання для виявлення аномальної активності та потенційних загроз;
- включає можливості виявлення DDoS-атак, ексфільтрації даних, внутрішніх загроз і багато іншого;
- підтримує інтеграцію з іншими рішеннями безпеки для кореляції даних і виявлення складних загроз;
- налаштування автоматичних дій у відповідь на виявлені загрози;
- інструменти для детального розслідування інцидентів, включаючи можливості аналізу трафіку до та після інциденту.

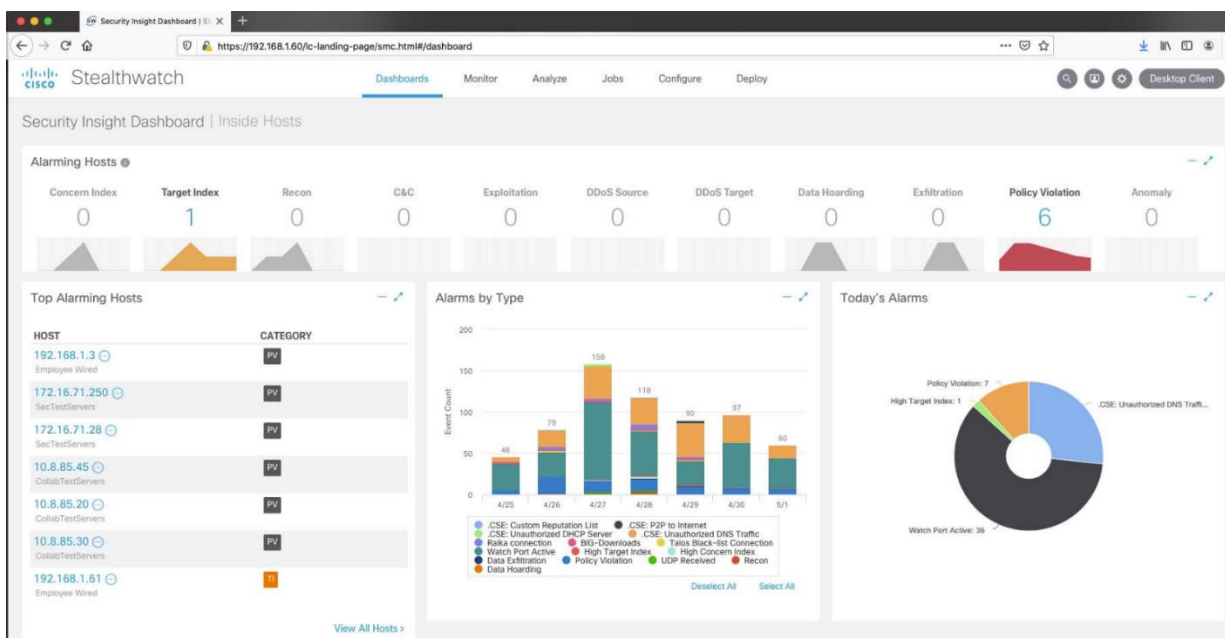


Рисунок 1.4 – Приклад графічного інтерфейсу Cisco Stealthwatch
«Cisco Stealthwatch»
Джерело: [17]

1.2.5 SolarWinds Network Performance Monitor

SolarWinds Network Performance Monitor (NPM) це програмне забезпечення для моніторингу мережі, яке забезпечує видимість, аналіз та управління мережевою

продуктивністю. NPM дозволяє виявляти, діагностувати та вирішувати проблеми мережі за допомогою інтуїтивно зрозумілих інструментів та інтеграцій.

Основні функції та можливості SolarWinds Network Performance Monitor [10]:

- відстеження використання пропускну здатності по інтерфейсах, щоб виявляти вузькі місця;
- вимірювання затримки між пристроями та відстеження метрики якості обслуговування;
- аналіз мережевого трафіка для визначення джерел високого навантаження і аномальної активності;
- надання інтерактивних карт мережі, що дозволяють бачити статус кожного пристрою та зв'язку (див. рисунок 1.5);
- автоматичне створення та оновлення топологічних карт мережі;
- виявлення пристроїв і застосування відповідних шаблонів моніторингу;
- проведення детального аналізу мережевих інцидентів і надання рекомендацій для усунення проблем;
- можливість налаштування сповіщень для виявлення проблем у реальному часі та швидкого реагування;
- підтримка різних форматів звітів для аналізу мережевої продуктивності та інцидентів;
- використання історичних даних для прогнозування майбутніх проблем і планування;
- підтримка інтеграцій з іншими продуктами SolarWinds, такими як NetFlow Traffic Analyzer, Server & Application Monitor, та багато інших;
- можливість розширення функціоналу через плагіни та API для інтеграції з сторонніми системами.

SolarWinds Network Performance Monitor це потужний інструмент для забезпечення високої продуктивності і надійності мережі, який підходить як для середніх, так і великих підприємств завдяки своїй функціональності та зручному інтерфейсу.

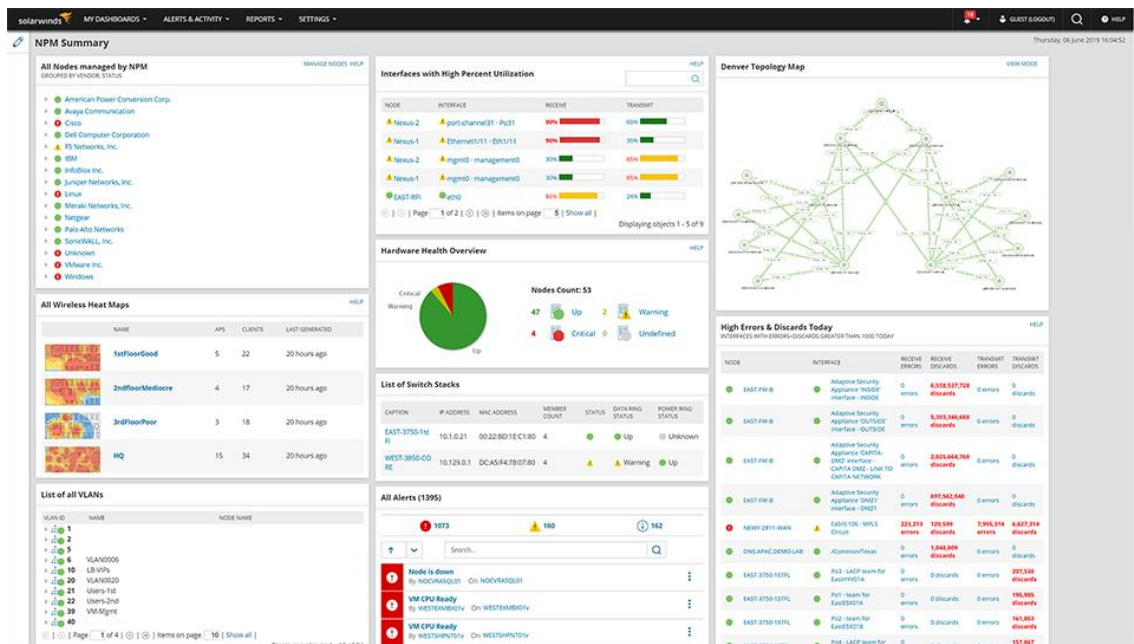


Рисунок 1.5 – Приклад графічного інтерфейсу SolarWinds Network Performance Monitor

«SolarWinds Network Performance Monitor»

Джерело: [18]

1.2.6 Cacti

Cacti це інструмент з відкритим кодом для моніторингу мережевих ресурсів та трафіку, який спеціалізується на створенні графіків із використанням системи RRDTool. Cacti дозволяє адмініструвати та візуалізувати дані про продуктивність мережі та системи, використовуючи інтерфейс, орієнтований на простоту та ефективність.

Основні можливості Cacti:

- збирання даних за допомогою SNMP, скриптів або команд для генерації графіків;
- автоматична генерація графіків на основі зібраних даних (див Рисунок 1.6);
- використання RRDTool для зберігання даних у форматі кругових баз даних, що дозволяє ефективно зберігати та візуалізувати великі обсяги даних;

- можливість створення та використання шаблонів для хостів, графіків, джерел даних;
- полегшує налаштування моніторингу схожих об'єктів у великих середовищах;
- підтримка простого мережевого управління протоколом SNMP для збору метрик з мережевих пристроїв;
- підтримка масштабування для моніторингу великої кількості пристроїв і джерел даних;
- веб-інтерфейс для налаштування та керування графіками, шаблонами та джерелами даних;
- можливість створення персоналізованих переглядів графіків.

Sacti є корисним інструментом для організацій, яким потрібна ефективна система для збору та візуалізації даних про мережеву і серверну продуктивність, надаючи простий у використанні інтерфейс та широкі можливості для налаштування і розширення.

The screenshot shows the Sacti web interface. The top navigation bar includes 'Graphs', 'Reporting', and 'Logs'. The left sidebar contains a menu with options like 'Main Console', 'Create', 'Management', 'Data Collection', 'Templates', 'Automation', 'Presets', 'Import/Export', 'Configuration', 'Utilities', and 'Troubleshooting'. The main content area displays a table of device metrics under the 'Reporting' tab. The table has columns for Device Description, Hostname, ID, Graphs, Data Sources, Status, In State, Uptime, Poll Time, Current (ms), Average (ms), Availability, and Created. The table lists various devices including Cacti Server, Central NAS, HP Printer, and several vhost entries. The bottom of the table shows pagination information: '1 to 18 of 56' and 'Next >>'.

Device Description	Hostname	ID	Graphs	Data Sources	Status	In State	Uptime	Poll Time	Current (ms)	Average (ms)	Availability	Created
Cacti Server	localhost	1	4	5	Up	N/A	N/A	0.1	0	0	100 %	2020-09-06 21:43:06
Central NAS	192.168.11.105	56	12	19	Up	120	42	0.26	0.35	1.15	99.36 %	2020-09-06 21:43:06
HP Printer	192.168.11.174	55	22	22	Up	137	54	0.65	1.04	1.8	99.81 %	2020-09-06 21:43:06
vhost01	192.168.11.201	46	12	19	Up	120	4	0.38	1.45	1.61	99.99 %	2020-09-06 21:43:06
vhost02	192.168.11.202	45	12	19	Up	120	4	0.34	0.56	0.94	99.99 %	2020-09-06 21:43:06
vhost03	192.168.11.203	44	12	19	Up	120	4	0.24	0.9	2.09	99.98 %	2020-09-06 21:43:06
vhost04	192.168.11.204	43	12	19	Up	120	4	0.26	1.01	0.76	100 %	2020-09-06 21:43:06
vhost05	192.168.11.205	42	12	19	Up	120	4	0.33	0.83	1.25	99.99 %	2020-09-06 21:43:06
vhost06	192.168.11.206	41	12	19	Up	120	4	0.39	0.74	0.79	100 %	2020-09-06 21:43:06
vhost07	192.168.11.207	40	12	19	Up	267	4	0.4	0.52	1.06	98.93 %	2020-09-06 21:43:06
vhost08	192.168.11.208	39	12	19	Up	120	4	0.19	0.89	1.24	99.99 %	2020-09-06 21:43:06
vhost09	192.168.11.209	38	12	19	Up	267	4	0.15	0.7	1.07	98.93 %	2020-09-06 21:43:06
vhost10	192.168.11.210	37	12	19	Up	120	4	0.22	0.77	0.77	100 %	2020-09-06 21:43:06
vhost11	192.168.11.211	36	12	19	Up	120	4	0.09	2.61	1.01	99.98 %	2020-09-06 21:43:06
vhost12	192.168.11.212	35	12	19	Up	120	4	0.32	1.14	1.09	99.99 %	2020-09-06 21:43:06
vhost13	192.168.11.213	34	12	19	Up	120	4	0.25	2.63	1.05	99.98 %	2020-09-06 21:43:06
vhost14	192.168.11.214	33	12	19	Up	267	4	0.26	3.99	1.02	98.93 %	2020-09-06 21:43:06
vhost15	192.168.11.215	32	12	19	Up	120	4	0.31	1.11	0.93	99.99 %	2020-09-06 21:43:06

Рисунок 1.6 – Приклад графічного інтерфейсу Sacti

«Sacti»

Джерело [19]

1.3 Постановка задачі

Нижче наведена загальна постановка задачі або технічна пропозиція для розробки вебзастосунку для моніторингу трафіку провайдера.

Функціональні вимоги:

- відображення поточних мережевих підключень на відповідні ідентифікатори процесів (PID);
- відображення інформації про ім'я процесу, завантажений і вивантажений обсяг трафіку, а також швидкість завантаження і завантаження в реальному часі;
- захоплення мережевого трафіку на вказаному інтерфейсі.

Технічні вимоги:

- меню вибору інтерфейсів;
- таблиця для запису статистики використання трафіку;
- поле для запису резюме пакетів.

РОЗДІЛ 2

ВИБІР ПРОГРАМНИХ ЗАСОБІВ ТА ПРОЕКТУВАННЯ ВЕБЗАСТОСУНКУ ДЛЯ МОНІТОРИНГУ ТРАФІКУ ІНТЕРНЕТ-ПРОВАЙДЕРА

2.1 Технологія захоплення трафіку nrcap

Nrcap це високопродуктивна бібліотека для захоплення та передачі мережових пакетів, яка є сучасною та потужною заміною для старішої бібліотеки WinPcap. Вона розроблена спеціально для Windows і має низку переваг та особливостей, які роблять її популярним вибором серед мережових адміністраторів, розробників і дослідників.

Захоплення мережового трафіку на Windows без використання спеціалізованого програмного забезпечення, такого як Nrcap або WinPcap, має кілька значних обмежень та проблем, які зводять нанівець можливість виконувати цей процес ефективно. Основні причини необхідності спеціалізованого програмного забезпечення:

- Windows розділяє виконання кодів на кілька рівнів (кілець) безпеки: режим користувача (User Mode) та режим ядра (Kernel Mode). Доступ до низькорівневих операцій, таких як безпосереднє захоплення мережових пакетів, обмежений режимом ядра;
- строгий контроль доступу до апаратних ресурсів і мережових стеків;
- Windows API не забезпечує прямого доступу до мережових пакетів на рівні низькорівневих стеків протоколів;
- платформа Windows Sockets (WinSock) забезпечує програмний інтерфейс для мережових комунікацій, але вона працює на вищому рівні і не дозволяє захоплювати або аналізувати трафік на рівні L2 або L3 моделі OSI.
- доступ до NDIS-пакетів без використання спеціальних драйверів захоплення у користувацькому просторі обмежений;

- підтримка RAW Sockets, яка дозволяє програмам захоплювати необроблені мережеві пакети, обмежена в Windows, особливо після Windows XP SP2, де вона стала ще більш обмеженою через проблеми безпеки;
- інші вбудовані утиліти Windows для діагностики мережі, такі як netsh trace, можуть захоплювати трафік, але вони не забезпечують гнучкість і потужність захоплення на рівні пакетів, які надає Npcap;
- системні засоби не дозволяють виконувати ефективну фільтрацію трафіку на рівні пакетів або застосовувати правила BPF, що є стандартом в аналізі мереж;
- без використання спеціалізованих драйверів, обробка мережевих пакетів не може бути настільки ефективною, тому що стандартні засоби не здатні обробляти великий обсяг даних з високою швидкістю.

Wireshark це один з найпоширеніших інструментів для аналізу мережевого трафіку. Він покладається на Npcap/WinPcap для захоплення трафіку (див. рисунок 2.1).

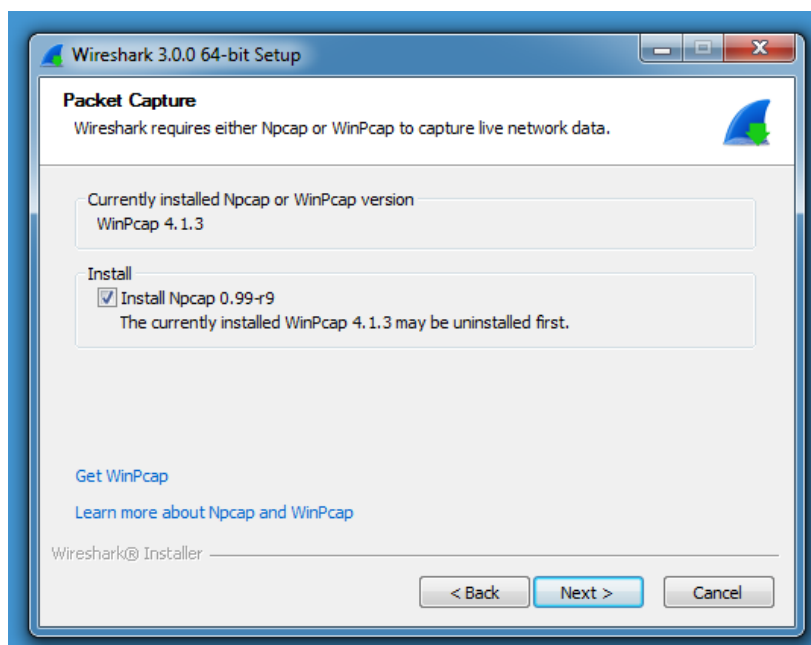


Рисунок 2.1 – Інсталяція Wireshark

Джерело: запропоновано автором

Без цього драйвера він не може взаємодіяти з мережевими інтерфейсами на низькому рівні. Npcap дозволяє обходити обмеження стандартної моделі безпеки Windows і стандартних API, забезпечуючи доступ до мережових пакетів на низькому рівні, що необхідно для ефективного аналізу мережевого трафіку. Використання таких драйверів, як Npcap, є критично важливим для завдань, що вимагають детального аналізу та захоплення мережевого трафіку на Windows.

Основні функції та переваги Npcap:

- Npcap оптимізована для кращої продуктивності, що дозволяє більш ефективно захоплювати та передавати мережеві пакети, особливо на високошвидкісних мережах.
- Npcap активно підтримується та оновлюється для роботи з останніми версіями Windows, включаючи Windows 10 та Windows Server 2019.
- Npcap може працювати як у привілейованому (admin) режимі, так і без привілеїв, що підвищує безпеку та зручність використання в різних сценаріях.
- Підтримка захоплення пакетів у режимі моніторингу (monitor mode). Це дозволяє захоплювати пакети, які не призначені для конкретного хоста, що корисно для аналізу бездротових мереж.
- Npcap дозволяє захоплювати та аналізувати пакети на каналному рівні, надаючи доступ до низькорівневих даних мережових пакетів.
- Npcap має поліпшене управління енергоспоживанням, що знижує навантаження на систему під час тривалого захоплення пакетів.

Програмний інтерфейс Npcap може використовуватися багатьма типами мережових інструментів для аналізу, усунення несправностей, забезпечення безпеки та моніторингу. Зокрема, класичні інструменти, які покладаються на Npcap, включають:

- аналізатори мереж і протоколів;
- мережеві монітори;
- реєстратори трафіку;
- генератори трафіку;

- користувачькі мости та маршрутизатори;
- системи виявлення вторгнень у мережу;
- мережеві сканери;
- інструменти безпеки.

Nrcar приймає та відправляє пакети незалежно від хост-протоколів, таких як TCP/IP. Це означає, що він не може блокувати, фільтрувати або маніпулювати трафіком, згенерованим іншими програмами на тому ж комп'ютері: він просто «перехоплює» пакети, що проходять через мережу. Таким чином, він не забезпечує належної підтримки для застосунків, таких як шейпери трафіку, планувальники QoS та персональні брандмауери.

2.2 Принцип роботи nrcar

NDIS це стандарт, який визначає зв'язок між мережевим адаптером і драйверами протоколів. Основна мета NDIS полягає в тому, щоб діяти як оболонка, яка дозволяє драйверам протоколів надсилати та отримувати пакети в мережі, не звертаючи уваги ні на конкретний адаптер, ні на конкретну операційну систему.

По-перше, система захоплення повинна обійти стек протоколів операційної системи, щоб отримати доступ до необроблених даних, що передаються мережею. Це вимагає наявності компонента, який працює всередині ядра ОС і взаємодіє безпосередньо з драйверами мережевого інтерфейсу. Цей компонент є дуже залежним від системи, і в нашому рішенні він реалізований у вигляді драйвера пристрою, що називається Netgroup Packet Filter. Цей драйвер забезпечує базові функції, такі як захоплення та ін'єкція пакетів, а також більш розширені функції, такі як програмований фільтрувальний механізм та моніторинговий двигун. Фільтрувальна система може використовуватися для обмеження сесії захоплення до підмножини мережевого трафіку; моніторинговий двигун надає потужний, але простий у використанні механізм для отримання статистики по трафіку.

По-друге, система захоплення повинна експортувати інтерфейс, який користувачські програми можуть використовувати для доступу до функцій, наданих драйвером ядра. Npcap надає дві різні бібліотеки: `packet.dll` та `wpcap.dll`.

`Packet.dll` пропонує низькорівневий API, який можна використовувати для прямого доступу до функцій драйвера з програмним інтерфейсом, незалежним від операційної системи Microsoft.

`Wpcap.dll` експортує потужніший набір високорівневих примітивів захоплення, які сумісні з `libpcap`, відомою бібліотекою захоплення Unix. Ці функції дозволяють здійснювати захоплення пакетів незалежно від підлеглого мережевого обладнання та операційної системи.

NDIS підтримує три типи мережевих драйверів: NIC, проміжні драйвери, транспортні драйвери або драйвери протоколів.

Драйвери NIC безпосередньо керують платами мережевого інтерфейсу, які називаються NIC. Драйвери NIC підключаються безпосередньо до апаратного забезпечення на їх нижньому краю та на їхньому верхньому краю представляють інтерфейс, який дозволяє верхнім рівням надсилати пакети в мережі, обробляти переривання, скидати NIC, зупиняти NIC, а також запитувати та встановлювати експлуатаційні характеристики драйвера. Драйвери NIC можуть бути або мініпортами, або застарілими повними драйверами NIC.

Драйвери мініпорту реалізують лише апаратно-спеціальні операції, необхідні для керування NIC, включаючи надсилення та отримання даних на NIC. Звичайні операції для всіх драйверів NIC найнижчого рівня, такі як синхронізація, забезпечуються NDIS. Мініпорт не відстежує прив'язки. Він просто передає пакети до NDIS, а NDIS гарантує, що ці пакети передаються до правильних протоколів.

Повні драйвери NIC були написані для виконання операцій, пов'язаних із апаратним забезпеченням, і всіх операцій синхронізації та постановки в чергу, які зазвичай виконує NDIS. Повноцінні драйвери NIC, наприклад, зберігають власну прив'язувальну інформацію для вказівки отриманих даних.

Проміжні драйвери інтерфейси поміж драйверів верхнього рівня, таким як драйвер протоколу, і мініпорт. Для драйвера верхнього рівня проміжний драйвер

виглядає як мініпорт. Для мініпорту проміжний драйвер виглядає як драйвер протоколу. Проміжний драйвер протоколу може накладатися на інший проміжний драйвер, хоча таке накладання може мати негативний вплив на продуктивність системи. Типовою причиною розробки проміжного драйвера є виконання транслявання медіа між існуючим застарілим драйвером протоколу та мініпортом, який керує NIC для нового типу медіа, невідомого драйверу протоколу. Наприклад, проміжний драйвер може транслювати з протоколу LAN на протокол ATM. Проміжний драйвер не може зв'язуватися з програмами режиму користувача, а лише з іншими драйверами NDIS.

Драйвер протоколу реалізує стек мережеских протоколів, як IPX/SPX або TCP/IP, пропонуючи свої послуги через одну або декілька карт мережевого інтерфейсу. Драйвер протоколу обслуговує клієнтів прикладного рівня на своєму верхньому краю та підключається до одного або кількох драйверів NIC або проміжних драйверів NDIS на своєму нижньому краю.

NPF реалізований як драйвер протоколу. Це не найкращий вибір з точки зору продуктивності, але забезпечує розумну незалежність від рівня MAC, а також повний доступ до необробленого трафіку.

На рисунку 2.2 наведено вигляд структури NDIS з двома стеками захоплення пакетів, прив'язаних до одного мережевого адаптера.

Взаємодія з ОС зазвичай асинхронна. Це означає, що драйвер надає набір функцій зворотного виклику, які викликаються системою, коли для NPF потрібна якась операція. NPF експортує функції зворотного виклику для всіх операцій вводу/виводу програм: відкриття, закриття, читання, запис, тощо.

Взаємодія з NDIS також є асинхронною: такі події, як надходження нового пакету, повідомляються в NPF через функцію зворотного виклику. Крім того, взаємодія з NDIS і драйвером NIC завжди відбувається за допомогою неблокуючих функцій: коли NPF викликає функцію NDIS, виклик негайно повертається; коли обробка завершується, NDIS викликає певний зворотний виклик NPF, щоб повідомити, що функція завершена. Драйвер експортує зворотний виклик для будь-

якої низькорівневої операції, як-от надсилання пакетів, встановлення або запит параметрів на NIC, тощо.

На рисунку 2.3 наведено структуру стеку захоплення пакетів, звертаючись до драйвера NPF.

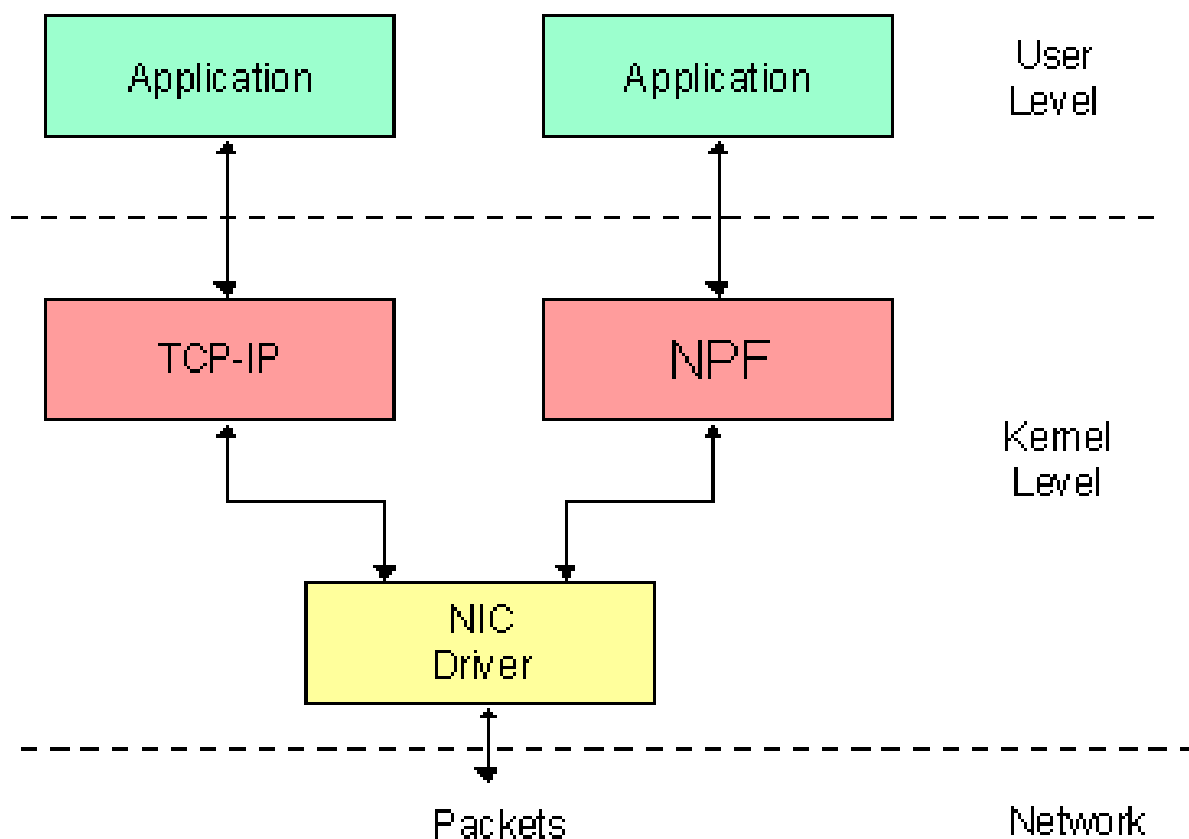


Рисунок 2.2 – Структури NDIS з двома стеками захоплення пакетів
«NPF driver internals manual»

Джерело: [20]

NPF здатний виконувати низку різних операцій: захоплення, моніторинг, скидання на диск, впровадження пакетів.

Найважливішою операцією NPF є захоплення пакетів. Під час захоплення драйвер «нюхає» пакети за допомогою мережевого інтерфейсу та доставляє їх у недоторканому вигляді до програм на рівні користувача.

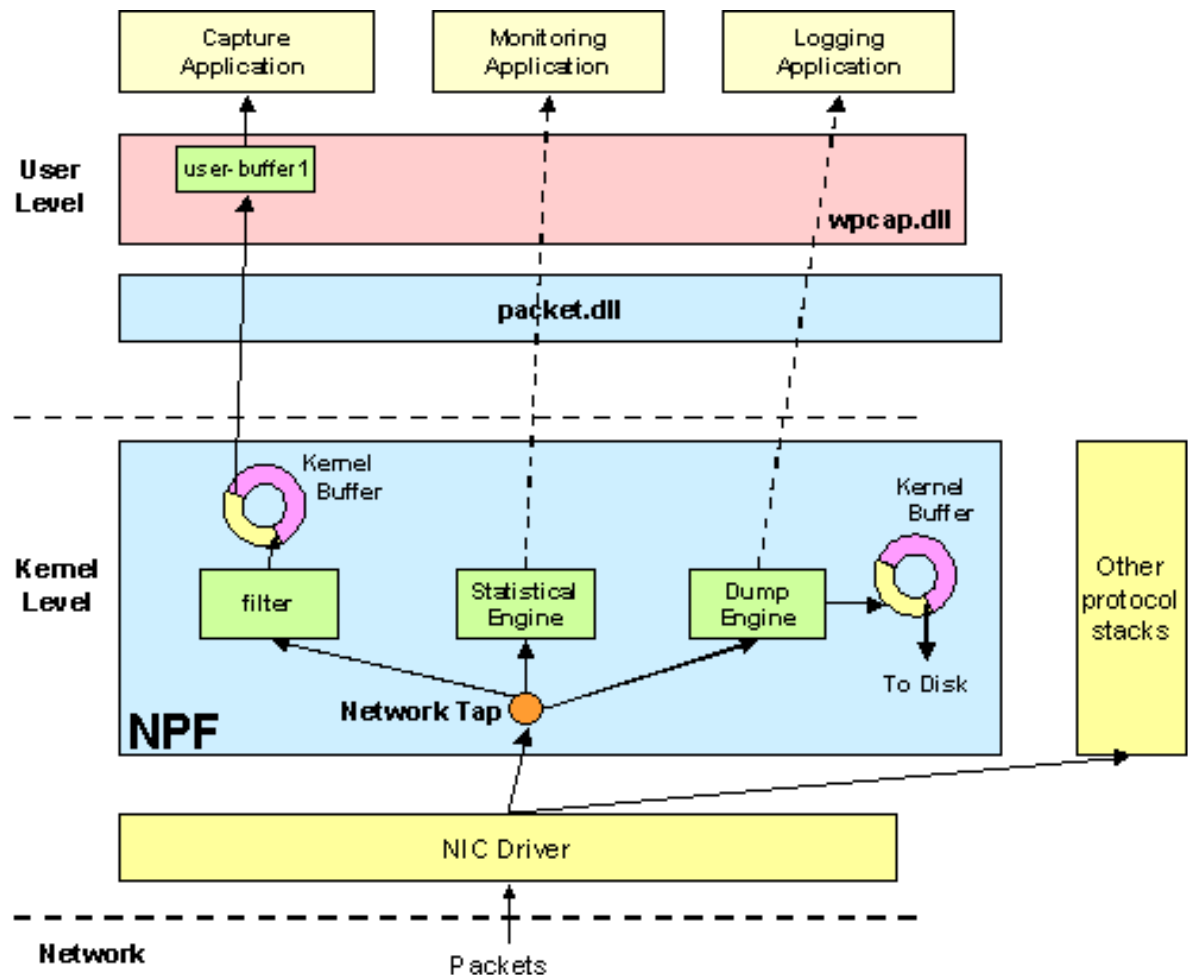


Рисунок 2.3 – Структура стеку захоплення пакетів

«NPF driver internals manual»

Джерело: [20]

2.3 Scapy

Scapy це потужна бібліотека для аналізу та генерації мережових пакетів на Python. Вона широко використовується для мережових аналізів, тестування безпеки, написання прототипів нових мережових протоколів, а також для мережевого моніторингу.

Основні можливості Scapy [11]:

- можливість створювати пакети будь-якого типу (Ethernet, IP, TCP, UDP і т.д.), змінювати їх поля і відправляти їх через мережу;

- Scapy може захоплювати пакети з мережі і розбирати їх для подальшого аналізу;
- Scapy підтримує різні типи мережеских сканувань, такі як SYN, ACK, UDP сканування і т.д.;
- можливість інтеграції Scapy з іншими мережевими інструментами для більш глибокого аналізу або автоматизації завдань;
- можливість створення нових протоколів або розширення існуючих, шляхом додавання своїх власних обробників.

Інші інструменти дотримуються парадигми "програма, яка запускається з оболонки". Це призводить до жахливого синтаксису для опису пакета. У цих інструментах використовується вища, але менш потужна форма опису у вигляді сценаріїв, придуманих автором інструменту. Наприклад, для сканера портів потрібно вказати лише IP-адресу, щоб запустити сценарій сканування портів. Навіть якщо сценарій трохи змінюється, ви все одно прив'язані до сканування портів.

Парадигма Scapy полягає в тому, щоб запропонувати спеціалізовану мову предметної області, яка дозволяє потужний і швидкий опис будь-якого типу пакета. Використання синтаксису Python і інтерпретатора Python як синтаксису і інтерпретатора DSL має багато переваг: немає необхідності писати окремий інтерпретатор, користувачам не потрібно вивчати ще одну мову, і вони отримують у користування повну, лаконічну та дуже потужну мову.

Scapy дозволяє користувачеві описувати пакет або набір пакетів як шари, що накладаються один на одного. Поля кожного шару мають корисні значення за замовчуванням, які можуть бути перевантажені. Scapy не змушує користувача використовувати попередньо визначені методи або шаблони. Це позбавляє від необхідності писати новий інструмент щоразу, коли потрібен інший сценарій. У C для опису пакета може знадобитися в середньому 60 рядків коду. З Scapy пакети можна описати лише одним рядком, а ще одним – вивести результат. 90% інструментів для зондування мереж можна переписати в два рядки на Scapy.

2.4 Pandas

Бібліотека Pandas це потужний інструмент для обробки та аналізу даних у Python. Вона надає структури даних та функції для ефективної роботи з табличними даними, що робить її відмінним вибором для роботи зі статистикою, аналізом даних та маніпулюванням даними в загальному.

Основні концепції та можливості бібліотеки Pandas включають:

- DataFrame є основною структурою даних в Pandas. Це таблична структура, що містить рядки та стовбці з даними;
- кожен стовбець у DataFrame представляється як одновірний масив даних;
- Pandas може зчитувати дані з різних джерел, таких як CSV, Excel, бази даних SQL, JSON і т.д., і конвертувати їх у DataFrame;
- Pandas дозволяє зберігати дані у різних форматах файлів;
- Pandas надає зручний механізм для вибірки та доступу до даних у DataFrame за допомогою різних методів і операторів індексації;
- Pandas підтримує багато вбудованих операцій над даними, таких як об'єднання, злиття, групування, фільтрація, сортування тощо;
- завдяки можливостям Pandas можна легко виконувати агрегацію даних та обчислювати статистичні показники, такі як середнє значення, медіана, дисперсія і т.д.;
- Pandas може працювати з іншими бібліотеками візуалізації, такими як Matplotlib та Seaborn, щоб надати зручні інструменти для візуалізації даних у вигляді графіків, діаграм тощо;

Загалом, бібліотека Pandas є невід'ємною частиною екосистеми аналізу даних Python і дозволяє легко та ефективно працювати зі структурованими даними, що робить її незамінною для багатьох завдань, пов'язаних з обробкою та аналізом даних.

2.5 Psutil

Psutil це бібліотека Python, яка надає інтерфейс для збору інформації про системні ресурси і їх моніторинг. Вона дозволяє отримувати дані про використання процесора, пам'яті, дискових ресурсів, мережевих інтерфейсів і процесів. psutil є потужним інструментом для розробників, системних адміністраторів і аналітиків, які потребують інформації про стан системи для моніторингу, оптимізації та діагностики.

Основні можливості psutil:

- отримання загальної і пер-процесорної завантаженості;
- збір інформації про частоту процесора;
- використання фізичної (оперативної) пам'яті;
- інформація про swap (віртуальну) пам'ять;
- інформація про дискові розділи і файлові системи;
- вхідний/вихідний дисковий трафік;
- дані про використання дискового простору;
- збір статистики по мережевих інтерфейсах;
- дані про вхідний/вихідний мережевий трафік;
- інформація про мережеві з'єднання (TCP/UDP);
- інформація про всі активні процеси;
- можливість зупинки, відновлення, завершення процесів;
- збір метрик про процеси: використання CPU, пам'яті, відкриті файли і сокети;
- інформація про температуру, батарею (де підтримується).

Особливості psutil:

- працює на різних операційних системах (Windows, macOS, Linux, FreeBSD, SunOS);
- простий інтерфейс для доступу до системних метрик;
- підтримка безлічі функцій для різних аспектів системного моніторингу.

Обмеження psutil:

- деякі функції можуть вимагати прав адміністратора (root) на системі;
- надмірне використання psutil може вплинути на продуктивність системи, особливо при частому виклику функцій.

Psutil це універсальна бібліотека для моніторингу і збору інформації про системні ресурси в Python. Вона забезпечує доступ до важливої інформації про стан системи, що робить її незамінною для розробки скриптів для управління системами, моніторингу ресурсів і діагностики проблем.

2.6 Tkinter

Tkinter це стандартна бібліотека Python для створення графічних інтерфейсів користувача (GUI). Вона надає доступ до бібліотеки Tk GUI Toolkit і є універсальним інструментом для створення віконних додатків, що працюють на різних операційних системах (Windows, macOS, Linux).

Основні можливості tkinter:

- створення основних вікон програм;
- організація елементів інтерфейсу всередині фреймів для кращої структури і управління макетом;
- інтерактивні елементи для виконання дій;
- відображення тексту або зображень;
- текстові поля для введення даних користувачем;
- відображення списків елементів;
- можливість використання перемикачів та прапорців;
- багаторядкові поля для введення тексту;
- інтерактивні полози для вибору значення з діапазону;
- системи меню для додатків;
- простий спосіб розміщення виджетів вертикально або горизонтально;
- гнучке розміщення виджетів у вигляді таблиці;
- абсолютне позиціонування виджетів з вказівкою точних координат;

- обробка різних подій, таких як натискання кнопок, рух миші, натискання клавіш;
- зв'язування подій з функціями зворотнього виклику для динамічного реагування на дії користувача;
- стандартні діалогові вікна для вибору файлів, відображення повідомлень, підтвердження дій і вибору кольору;
- підтримка тем для стилізації виджетів з використанням бібліотеки `ttk` (`tkinter.ttk`).

Переваги `tkinter`:

- не потребує зовнішніх залежностей для роботи;
- легкий для вивчення та використання;
- працює на багатьох операційних системах без змін у коді;
- включає всі основні елементи інтерфейсу;

Недоліки `tkinter`:

- може виглядати застарілим у порівнянні з новішими фреймворками;
- нелегко налаштовувати або створювати складні користувацькі віджети;
- може бути недостатньо гнучким для дуже великих або складних програм.

Tkinter це потужний і простий інструмент для створення графічних інтерфейсів на Python, який підходить для різноманітних додатків від простих утиліт до більш складних програм. Він забезпечує широкий спектр можливостей для роботи з інтерфейсом користувача і є гарним вибором для тих, хто шукає простий та ефективний спосіб створення GUI на Python.

2.7 Засоби розробки програмного забезпечення

Для реалізації програмного забезпечення було обрано мову програмування python. Python є одним з найпопулярніших мов програмування для роботи з інтернет-трафіком завдяки своїй простоті, великій кількості бібліотек і активній спільноті. Python надає багатий набір інструментів для роботи з інтернет-трафіком,

що дозволяє ефективно захоплювати, аналізувати і маніпулювати мережевими пакетами.

Python це високорівнева мова програмування загального призначення, яка була створена Гвідо ван Россумом і вперше випущена в 1991 році. Вона відома своєю простотою, зрозумілим синтаксисом і великою кількістю бібліотек, що робить її популярною серед програмістів різного рівня.

Основні характеристики Python:

- Python має простий і зрозумілий синтаксис, що сприяє читабельності коду. Наприклад, використовується відступ (індентифікація) для визначення блоків коду;
- Python є інтерпретованою мовою, тобто код виконується рядок за рядком за допомогою інтерпретатора, що спрощує налагодження;
- Python підтримує динамічну типізацію, де типи змінних визначаються під час виконання;
- Мова пропонує високий рівень абстракції над апаратними та системними ресурсами, що дозволяє зосередитися на логіці вирішення завдань;
- Python підтримує кілька парадигм програмування, включаючи об'єктно-орієнтоване, процедурне та функціональне програмування.

Переваги Python:

- простота і зрозумілість;
- широка стандартна бібліотека;
- підтримка різних парадигм програмування;
- велика і активна спільнота.

Недоліки Python:

- порівняно низька швидкодія для завдань, що вимагають високої продуктивності;
- глобальна блокування інтерпретатора, що обмежує продуктивність багатопоточних додатків.

В якості середовища розробки було вирішено використовувати середовище PyCharm. PyCharm – це інтегроване середовище розробки для мови програмування

Python, розроблене компанією JetBrains. Воно забезпечує потужний набір інструментів для розробки, налагодження, тестування та управління проектами Python (див. рисунок 2.4). PyCharm відомий своїми розширеними функціями, що роблять розробку на Python ефективнішою і зручнішою [13].

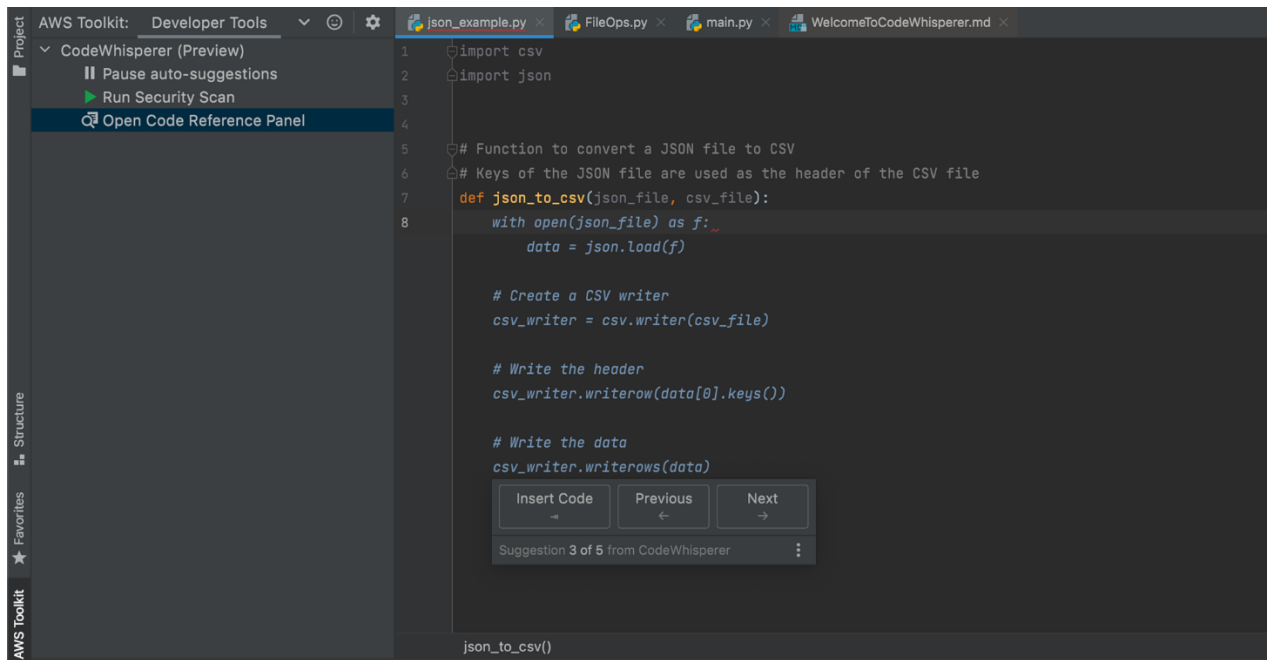


Рисунок 2.4 – Середовище розробки PyCharm

Джерело: запропоновано автором

Основні особливості PyCharm:

- підтримка підсвічування коду Python і інших мов;
- PyCharm автоматично перевіряє код на наявність помилок і дає рекомендації (див. рисунок 2.5);
- пропозиції автодоповнення коду на основі контексту (див. рисунок 2.6);
- зручна навігація по проекту, включаючи перехід до визначення, пошук використань і швидке переміщення між файлами (див. рисунок 2.7);
- інструменти для рефакторингу коду, включаючи перейменування, зміну підпису функцій і розділення класів (див. рисунок 2.8);
- інструменти для підвищення продуктивності;

- PyCharm виконує аналіз коду і дає рекомендації для покращення;
- підтримка шаблонів і фрагментів коду для швидкого написання часто використовуваних конструкцій;
- можливість налаштування середовища розробки, включаючи підтримку різних стилів кодування;
- вбудована система підтримки Git, SVN, Mercurial та інших систем контролю версій;
- візуальні інструменти для порівняння файлів і вирішення конфліктів;
- підтримка повноцінного налагодження з точками зупинки, покроковим виконанням та переглядом значень змінних (див. рисунок 2.9);
- інструменти для профілювання продуктивності додатків;
- інтеграція з популярними базами даних (PostgreSQL, MySQL, Oracle тощо), включаючи можливості перегляду, редагування і виконання SQL-запитів;
- підтримка роботи з Jupyter Notebook, що дозволяє виконувати інтерактивні обчислення і візуалізувати результати прямо в PyCharm;
- підтримка створення і управління віртуальними середовищами (venv, Conda) для ізоляції залежностей проектів;
- інтеграція з популярними веб-фреймворками, такими як Django, Flask, FastAPI.
- підсвічування синтаксису, автодоповнення і налагодження коду для веб-технологій;
- інтеграція з науковими бібліотеками, такими як NumPy, Pandas, Matplotlib (див. рисунок 2.10);
- інтерактивний режим консолі з підтримкою Python (див. рисунок 2.11).

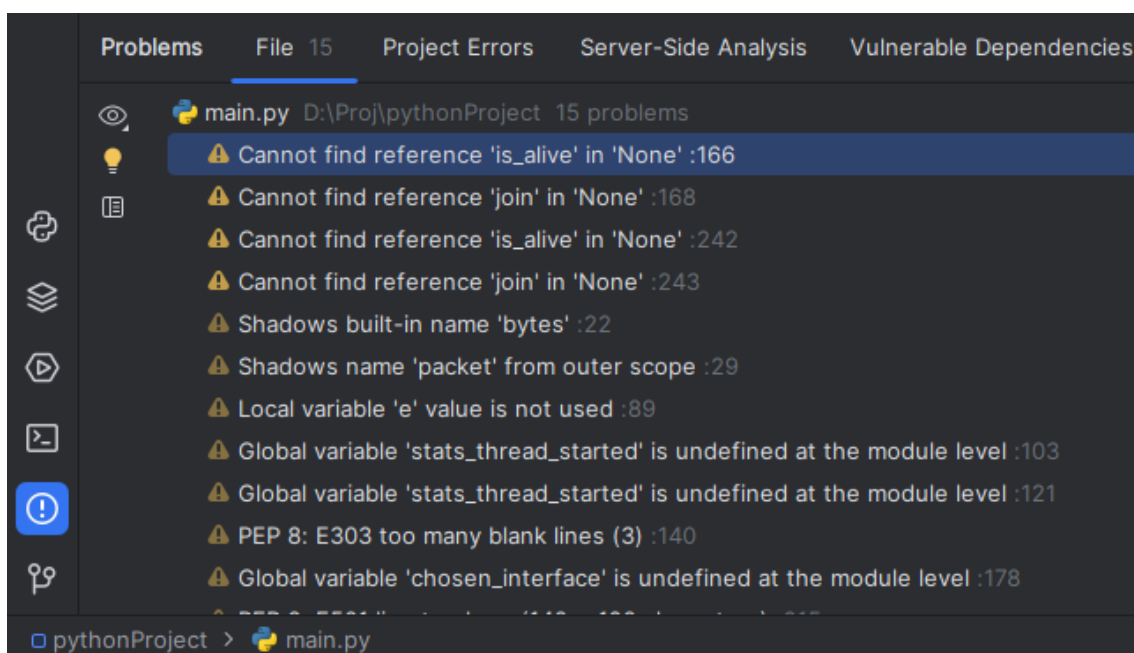


Рисунок 2.5 – Автоматична перевірка коду на наявність помилок і надання рекомендацій

Джерело: запропоновано автором

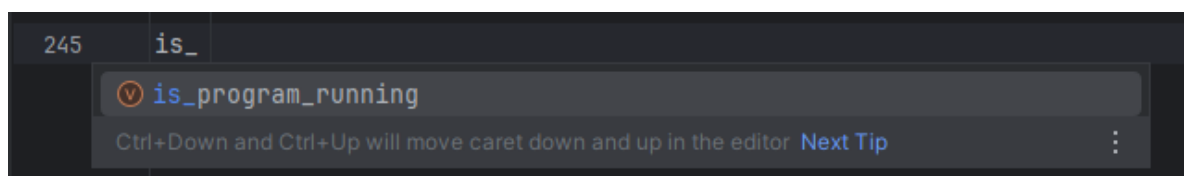


Рисунок 2.6 – Рекомендація на основі контексту

Джерело: запропоновано автором

Основною причиною для вибору PyCharm – розширений редактор коду. PyCharm надає інтелектуальне автодоповнення коду, підсвічування синтаксису, перевірку помилок у реальному часі та швидке виправлення, можливість швидко переходити до визначень, використань, класів, методів та файлів. Також підтримка розширеного рефакторингу, що включає перейменування, витяг змінних, методів, класів та інших структур.

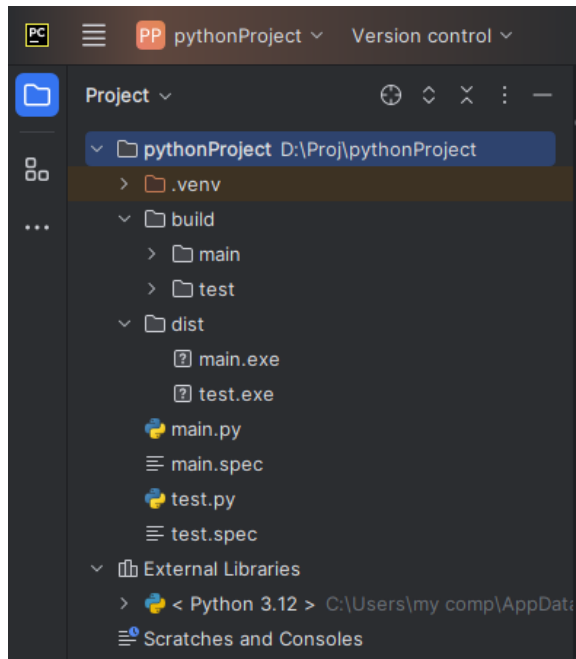


Рисунок 2.7 – Навігація по проєкту
Джерело: запропоновано автором

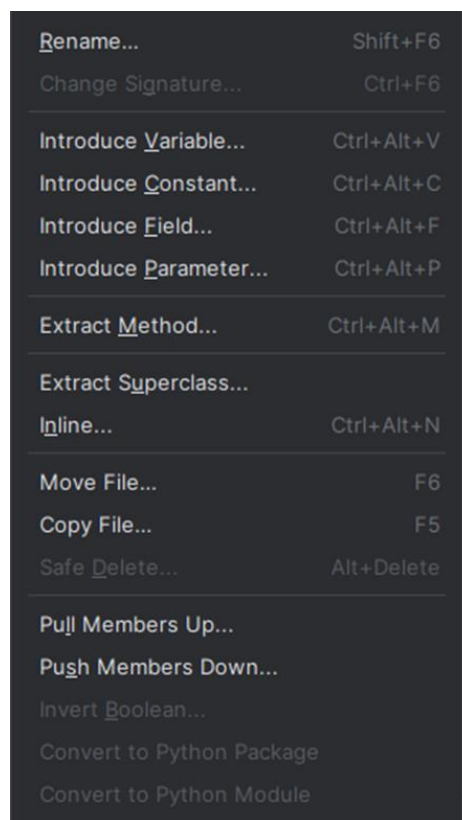


Рисунок 2.8 – Інструменти для рефакторингу
Джерело: запропоновано автором

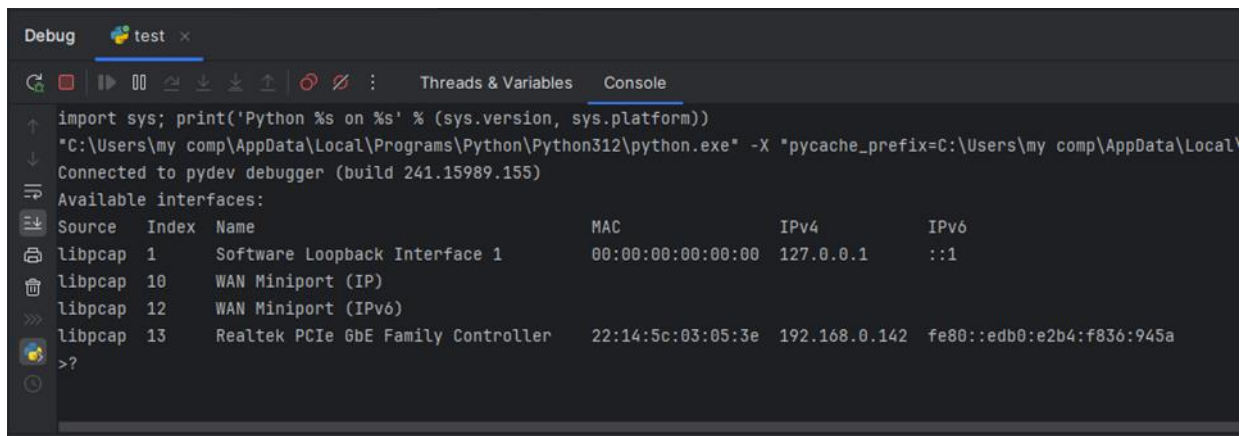


Рисунок 2.9 – Налаштування з точками зупинки, покроковим виконанням та переглядом значень змінних
Джерело: запропоновано автором

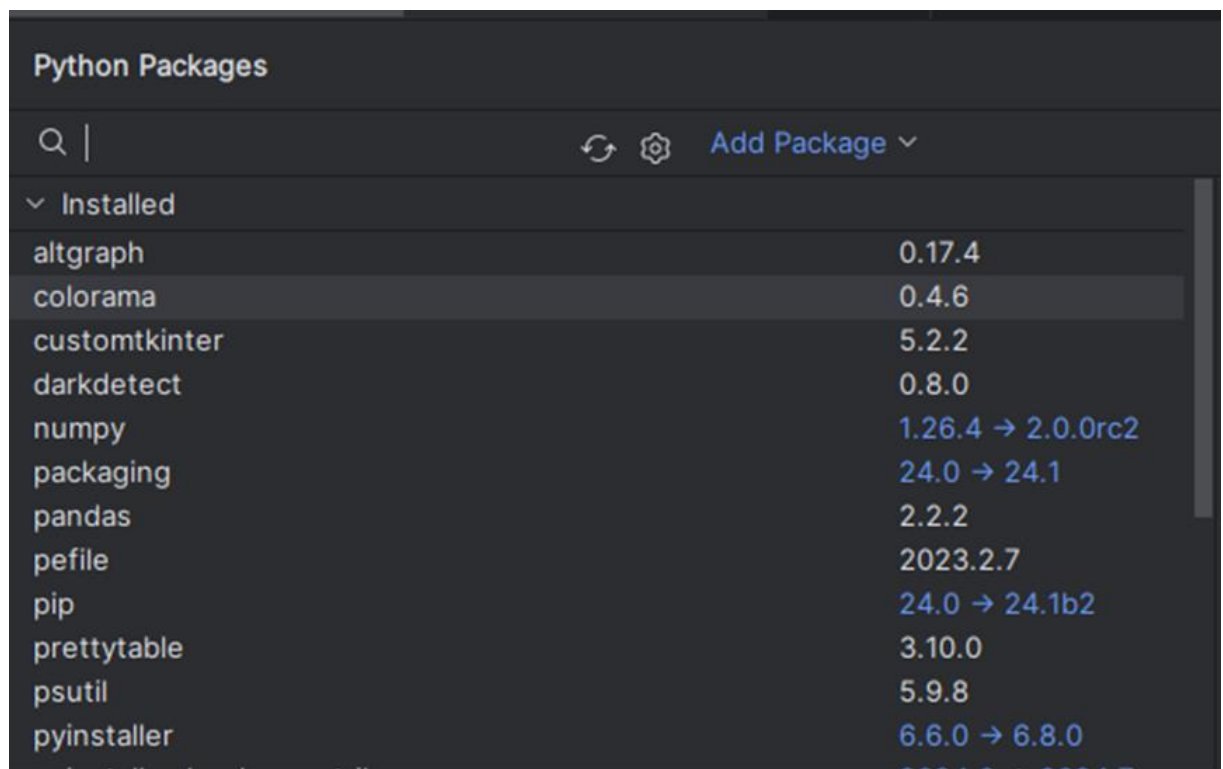


Рисунок 2.10 – Бібліотеки PyCharm
Джерело: запропоновано автором

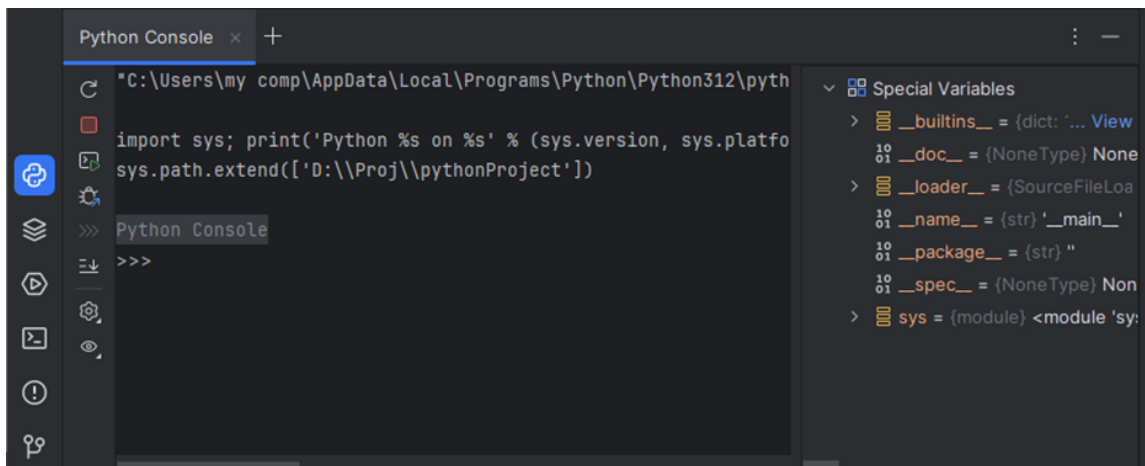


Рисунок 2.11 – Консоль Python

Джерело: запропоновано автором

РОЗДІЛ 3

ПРОЕКТУВАННЯ ТА РЕАЛІЗАЦІЯ ВЕБЗАСТОСУНКУ ДЛЯ МОНІТРОНИГУ ТРАФІКУ ІНТЕРНЕТ-ПРОВАЙДЕРА

3.1 Проектування вебзастосунку

Вебзастосунок призначений для моніторингу мережевого трафіку, асоціювання трафіку з процесами, що працюють у системі, та надання актуальних даних через графічний інтерфейс користувача (GUI). Користувач може обрати мережевий інтерфейс, запускати моніторинг, переглядати зведення пакетів та статистику трафіку у режимі реального часу.

Моніторинг мережевого трафіку:

- використання `scapy` для захоплення пакетів;
- використання `psutil` для асоціювання пакетів з процесами на основі з'єднань;
- зберігання даних трафіку для кожного процесу.

Графічний інтерфейс користувача (GUI):

- для елементів GUI, таких як кнопки, поля, текстові області та перегляд дерев;
- відображення актуальних зведень пакетів та статистики.

Потоки:

- потік захоплення пакетів захоплює та обробляє мережеві пакети;
- потік асоціації з'єднань відображає мережеві з'єднання на ідентифікатори процесів;
- потік друку статистики періодично оновлює GUI зі статистикою трафіку.

Структура вебзастосунку:

- для захоплення пакетів використовується функція `process_packet(packet)`, яка викликається для кожного пакета, що захоплюється. Вона аналізує пакет і оновлює статистику трафіку для відповідного процесу;
- для асоціації процесів використовується функція `get_connections()`, яка створює словник `connection2pid`, що зберігає відповідність між мережевими

з'єднаннями і процесами. Цей словник оновлюється у фоновому режимі;

- для виведення статистики використовується функція `print_pid2traffic()`, яка збирає дані про трафік для кожного процесу і оновлює дерево статистики в GUI;
- побудова GUI за допомогою Tkinter, що включає вибір інтерфейсу, кнопку запуску моніторингу, а також відображення зведення пакетів і статистики.

Основні функції:

- функція `get_size(bytes)` перетворює розмір у байтах на зручний для читання формат з відповідним суфіксом (B, KB, MB і т. д.);
- функція `process_packet(packet)` обробляє кожен захоплений пакет, оновлює статистику трафіку для процесів і додає інформацію про пакет у текстове поле GUI, якщо увімкнено відображення зведення пакетів;
- функція `get_connections()` це фоновий процес, що постійно оновлює словник `connection2pid` з поточними мережевими з'єднаннями і процесами, які їх використовують;
- функція `print_pid2traffic()` збирає дані про трафік для кожного процесу, формує DataFrame з цими даними і оновлює дерево статистики в GUI;
- функція `print_stats()` викликає `print_pid2traffic()` з певним інтервалом, якщо увімкнено відображення статистики, і зупиняє роботу, коли завершено моніторинг;
- функція `toggle_packet_summary()` перемикає режим відображення зведення пакетів;
- функція `toggle_print_stats()` перемикає режим відображення статистики;
- функція `start_sniffing(interface)` запускає процес захоплення пакетів на обраному інтерфейсі;
- функція `start_monitoring()` запускає фонові потоки для отримання з'єднань і захоплення пакетів, активує кнопки керування в GUI.

Для забезпечення асинхронної роботи програми використовується бібліотека `threading`. Програма запускає окремі потоки для:

- захоплення пакетів (`start_sniffing`);

- оновлення з'єднань (`get_connections`);
- виведення статистики (`print_stats`).

Синхронізація між потоками здійснюється за допомогою об'єкта `Event` (`command_event`), що дозволяє безпечно припиняти потоки.

Графічний інтерфейс:

- рамка `interface_frame`, що містить випадаючий список для вибору мережевого інтерфейсу та кнопку для запуску моніторингу;
- рамка `control_frame`, що містить кнопки для перемикання відображення зведення пакетів і статистики;
- рамка `display_frame`, що містить текстове поле для зведення пакетів та дерево для відображення статистики трафіку.

Взаємодія з користувачем:

- після запуску, користувач вибирає мережевий інтерфейс зі списку і натискає кнопку "Start Monitoring";
- під час роботи користувач може переключати відображення зведення пакетів і статистики за допомогою відповідних кнопок;
- після закриття GUI усі фонові процеси зупиняються.

На рисунках 3.1 – 3.2 наведений високорівневий псевдокод, що підсумовує проектування.

Проектування програми мережевого моніторингу охоплює аспекти обробки мережевого трафіку, асоціації даних з процесами, створення інтуїтивно зрозумілого GUI та забезпечення ефективної роботи через багатопоточність. Правильний підхід до кожного з цих аспектів дозволяє створити надійну та функціональну програму, що задовольняє вимоги користувачів. Цей дизайн охоплює основні аспекти програми, забезпечуючи баланс між функціональністю, продуктивністю та зручністю.

Ініціалізувати глобальні змінні та структури даних
Створити елементи інтерфейсу Tkinter

Функція `start_monitoring`:

- Перевірити вибраний інтерфейс
- Запустити потік захоплення пакетів
- Запустити потік асоціації з'єднань
- Увімкнути кнопки управління

Функція `start_sniffing(interface)`:

- Захоплювати пакети за допомогою `scapy`
- Для кожного пакета, викликати `process_packet`

Функція `get_connections`:

- Поки працює:
 - Отримати активні з'єднання за допомогою `psutil`
 - Відобразити з'єднання на PID
- Спати

Функція `process_packet(packet)`:

- Якщо відображення зведень увімкнено:
 - Оновити текстову область зведення пакетів
- Визначити з'єднання
- Оновити дані трафіку для відповідного PID

Функція `print_pid2traffic`:

- Створити `DataFrame` процесів та їх даних трафіку
- Оновити перегляд дерев у GUI

Рисунок 3.1 – Перша частина високорівневого псевдокоду вебзастосунку

Джерело: запропоновано автором

```

Функція print_stats:
    Поки працює:
        Якщо увімкнено:
            Викликати print_pid2traffic
        Чекати

Функція toggle_packet_summary:
    Перемкнути стан відображення зведення

Функція toggle_print_stats:
    Перемкнути стан відображення статистики

Обробити головний цикл GUI
Закрити потоки при виході

```

Рисунок 3.2 – Друга частина високорівневого псевдокоду

Джерело: запропоновано автором

3.2 Опис функціоналу везастосунку

Вебзастосунок моніторить мережевий трафік на вибраному інтерфейсі, асоціюючи його з процесами на комп'ютері. Вона використовує бібліотеку Scapy для перехоплення мережевих пакетів, бібліотеку psutil для отримання інформації про процеси та мережеві з'єднання, а також Pandas для обробки та представлення даних.

Цей вебзастосунок проводить моніторинг мережевого трафіку, який використовує бібліотеки tkinter, psutil, scapy та pandas для створення графічного інтерфейсу користувача (GUI) та моніторингу мережевого трафіку.

Основні компоненти програми:

- бібліотека tkinter для створення вікна програми та різних елементів інтерфейсу, таких як кнопки, поля вводу та текстові поля;
- бібліотеки psutil для отримання інформації про мережеві з'єднання та

процеси, що їх використовують;

- бібліотека `scapy` для аналізу та обробки мережевих пакетів;
- моніторинг трафіку відбувається в окремому потоці, щоб не блокувати основний потік виконання програми;
- бібліотека `threading` для запуску окремих потоків для моніторингу мережі та виводу статистики;
- взаємодія між потоками відбувається через події (Event) та спільні змінні;
- бібліотека `pandas` для створення та обробки даних, що стосуються процесів, мережевого трафіку та його швидкості;
- статистика відображається в таблиці в графічному інтерфейсі користувача.
- користувач може обирати інтерфейс для моніторингу;
- користувач може управляти відображенням підсумкових даних про мережевий трафік та статистику через відповідні кнопки у GUI;
- програма отримує інформацію про мережеві з'єднання та процеси з операційної системи через `psutil`;
- для аналізу та обробки мережевих пакетів використовується бібліотека `scapy`;
- можливість зупинити виконання програми та закрити всі потоки через взаємодію з графічним інтерфейсом користувача.

Функція `get_size(bytes)` конвертує байти в зручний для читання формат (див. рисунок 3.3).

```

4 usages
21  def get_size(bytes):
22      for unit in ['', 'K', 'M', 'G', 'T', 'P']:
23          if bytes < 1024:
24              return f"{bytes:.2f}{unit}B"
25          bytes /= 1024

```

Рисунок 3.3 – Функція `get_size(bytes)`

Джерело: запропоновано автором

Функція `get_connections()` постійно оновлює словник `connection2pid`, перевіряючи поточні мережеві з'єднання та їх відповідні PID за допомогою `psutil` (див. рисунок 3.4).

```

1 usage
47 def get_connections():
48     global connection2pid
49     while is_program_running:
50         if command_event.is_set():
51             return
52         for c in psutil.net_connections():
53             if c.laddr and c.raddr and c.pid:
54                 connection2pid[(c.laddr.port, c.raddr.port)] = c.pid
55                 connection2pid[(c.raddr.port, c.laddr.port)] = c.pid
56         time.sleep(1)

```

Рисунок 3.4 – Функція `get_connections()`

Джерело: запропоновано автором

Функція `print_stats()` постійно викликає `print_pid2traffic()`, якщо прапорець `print_stats_enabled` встановлено (див. рисунок 3.5).

```

1 usage
102 def print_stats():
103     global stats_thread_started
104     stats_thread_started.set()
105     while is_program_running:
106         if print_stats_enabled:
107             print_pid2traffic()
108             if command_event.wait(timeout=1):
109                 return

```

Рисунок 3.5 – Функція `print_stats()`

Джерело: запропоновано автором

Функція `print_pid2traffic()` створює `DataFrame`, що містить статистику трафіку для кожного процесу, та оновлює `Treeview` в GUI останніми даними (див. рисунок 3.6).

```

1 usage
59 def print_pid2traffic():
60     global global_df
61     processes = []
62     for pid, traffic in pid2traffic.items():
63         if command_event.is_set():
64             return
65         try:
66             p = psutil.Process(pid)
67         except psutil.NoSuchProcess:
68             continue
69         name = p.name()
70         try:
71             create_time = datetime.fromtimestamp(p.create_time())
72         except OSError:
73             create_time = datetime.fromtimestamp(psutil.boot_time())
74         process = {
75             "pid": pid, "name": name, "create_time": create_time, "Upload": traffic[0],
76             "Download": traffic[1],
77         }
78         try:
79             process["Upload Speed"] = traffic[0] - global_df.at[pid, "Upload"]
80             process["Download Speed"] = traffic[1] - global_df.at[pid, "Download"]
81         except (KeyError, AttributeError):
82             process["Upload Speed"] = traffic[0]
83             process["Download Speed"] = traffic[1]
84         processes.append(process)
85     df = pd.DataFrame(processes)
86     try:
87         df = df.set_index("pid")
88         df.sort_values(by="Download", inplace=True, ascending=False)
89     except KeyError as e:
90         pass
91     global_df = df
92
93     for row in stats_tree.get_children():
94         stats_tree.delete(row)
95     for index, row in df.iterrows():
96         stats_tree.insert( parent="", index=index, values=(index, row["name"], get_size(row["Upload"]),
97                                                             get_size(row["Download"]),
98                                                             get_size(row["Upload Speed"]) + "/s",
99                                                             get_size(row["Download Speed"]) + "/s"))

```

Рисунок 3.6 – Функція print_pid2traffic()

Джерело: запропоновано автором

Функція toggle_packet_summary() вмикає або вимикає відображення підсумків пакетів (див. рисунок 3.7).

```

2 usages
112 def toggle_packet_summary():
113     global display_packet_summary
114     display_packet_summary = not display_packet_summary
115     state = "enabled" if display_packet_summary else "disabled"
116     packet_summary_text.insert(tk.END, chars=f"Packet summary display {state}.\n")
117     packet_summary_text.see(tk.END)

```

Рисунок 3.7 – Функція toggle_packet_summary()

Джерело: запропоновано автором

Функція `toggle_print_stats()` вмикає або вимикає відображення статистики трафіку та керує відповідним потоком (див. рисунок 3.8).

```

2 usages
120 def toggle_print_stats():
121     global print_stats_enabled, print_stats_thread, stats_thread_started
122     if print_stats_enabled:
123         print_stats_enabled = False
124         if print_stats_thread is not None and print_stats_thread.is_alive():
125             command_event.set()
126             print_stats_thread.join()
127             command_event.clear()
128     else:
129         print_stats_enabled = True
130         if print_stats_thread is None or not print_stats_thread.is_alive():
131             command_event.clear()
132             stats_thread_started = Event()
133             print_stats_thread = Thread(target=print_stats)
134             print_stats_thread.start()
135             stats_thread_started.wait()
136             command_event.clear()

```

Рисунок 3.8 – Функція `print_stats()`

Джерело: запропоновано автором

Функція `start_sniffing(interface)` запускає захоплення пакетів на вибраному мережевому інтерфейсі з використанням Scapy (див. рисунок 3.9).

```

1 usage
140 def start_sniffing(interface):
141     try:
142         sniff(prn=process_packet, store=False, iface=interface, stop_filter=lambda x: command_event.is_set())
143     except Exception as e:
144         packet_summary_text.insert(tk.END, chars=f"Exception in sniffing thread: {e}\n")
145         packet_summary_text.see(tk.END)

```

Рисунок 3.9 – Функція `start_sniffing(interface)`

Джерело: запропоновано автором

Функція `handle_user_input()` обробляє команди з командного рядка для керування відображенням підсумків пакетів та статистики (див. рисунок 3.10). У GUI не використовується.

```

148 def handle_user_input():
149     global is_program_running, print_stats_enabled, display_packet_summary
150     while is_program_running:
151         cmd = input("Enter command (psum to display packet summary, stats to display stats printing,
152                     " p to stop packet summary display, s to stop stats printing): \n")
153         print("You entered:", cmd, end='\n')
154         if cmd == "psum":
155             toggle_packet_summary()
156         elif cmd == "stats":
157             toggle_print_stats()
158         elif cmd == "p":
159             if display_packet_summary:
160                 display_packet_summary = False
161                 print("Packet summary display stopped.")
162             else:
163                 print("Packet summary display is not currently enabled.")
164         elif cmd == "s":
165             print_stats_enabled = False
166             if print_stats_thread is not None and print_stats_thread.is_alive():
167                 command_event.set()
168                 print_stats_thread.join()
169                 command_event.clear()
170                 print("Stats printing stopped.")
171             else:
172                 print("Stats printing is not currently running.")
173         else:
174             print("Unknown command.")

```

Рисунок 3.10 – Функція handle_user_input()

Джерело: запропоновано автором

Функція start_monitoring() перевіряє вибраний інтерфейс, запускає потоки для відстеження з'єднань та захоплення пакетів (див. рисунок 3.11).

```

1 usage
177 def start_monitoring():
178     global chosen_interface
179     chosen_interface = interface_var.get()
180     if chosen_interface not in [iface_info.name for iface_info in ifaces.values()]:
181         packet_summary_text.insert(tk.END, chars: "This interface does not exist. Please choose a valid interface.\n")
182         packet_summary_text.see(tk.END)
183         return
184
185     connections_thread = Thread(target=get_connections)
186     connections_thread.start()
187
188     sniffing_thread = Thread(target=start_sniffing, args=(chosen_interface,))
189     sniffing_thread.start()
190
191     toggle_packet_summary_button.config(state=tk.NORMAL)
192     toggle_print_stats_button.config(state=tk.NORMAL)

```

Рисунок 3.11 – Функція start_monitoring()

Джерело: запропоновано автором

Для створення головне вікна вебзастосунку з назвою "Network Traffic Monitor" були використані команди `root`, `root.title` та `root.geometry` (див. рисунок 3.12).

```

196
197     root = tk.Tk()
198     root.title("Network Traffic Monitor")
199     root.geometry("800x600")
200

```

Рисунок 3.12 – Створення основного вікна програми

Джерело: запропоновано автором

Для реалізації вибору інтерфейсу був використаний компонент `interface`, що дозволяє вибрати мережевий інтерфейс (див. рисунок 3.13).

```

201     interface_frame = ttk.LabelFrame(root, text="Select Interface")
202     interface_frame.pack(padx=10, pady=10, fill="x")
203
204     interface_var = tk.StringVar(value="Select an Interface")
205     interfaces = [iface_info.name for iface_info in ifaces.values()]
206     interface_menu = ttk.Combobox(interface_frame, textvariable=interface_var, values=interfaces, state="readonly")
207     interface_menu.pack(padx=10, pady=10, fill="x")
208
209     start_button = ttk.Button(interface_frame, text="Start Monitoring", command=start_monitoring)
210     start_button.pack(padx=10, pady=10, fill="x")

```

Рисунок 3.13 – Реалізації вибору інтерфейсу

Джерело: запропоновано автором

```

209     start_button = ttk.Button(interface_frame, text="Start Monitoring", command=start_monitoring)
210     start_button.pack(padx=10, pady=10, fill="x")
211
212     control_frame = ttk.LabelFrame(root, text="Controls")
213     control_frame.pack(padx=10, pady=10, fill="x")
214
215     toggle_packet_summary_button = ttk.Button(control_frame, text="Toggle Packet Summary", command=toggle_packet_summary, state=tk.DISABLED)
216     toggle_packet_summary_button.pack(padx=10, pady=5, fill="x")
217
218     toggle_print_stats_button = ttk.Button(control_frame, text="Toggle Stats Display", command=toggle_print_stats, state=tk.DISABLED)
219     toggle_print_stats_button.pack(padx=10, pady=5, fill="x")

```

Рисунок 3.14 – Реалізації кнопок управління

Джерело: запропоновано автором

Для реалізації кнопок управління для керування підсумками пакетів і відображенням статистики було використані компоненти `toggle_packet_summary_button` та `toggle_print_stats_button` (див. рисунок 3.14).

Для реалізації відображення виводу було використано текстове поле для підсумків пакетів і Treeview для відображення статистики трафіку (див. рисунок 3.15).

Принцип роботи вебзастосунку:

- користувач вибирає мережевий інтерфейс з випадаючого меню та натискає кнопку "Start Monitoring";
- функція `start_monitoring()` перевіряє вибраний інтерфейс і запускає потоки для відстеження з'єднань та захоплення пакетів;
- функція `process_packet()` обробляє кожен захоплений пакет. Вона оновлює словник `pid2traffic` довжиною пакету (або як завантаження, або як вивантаження), залежно від MAC-адреси джерела пакета;
- функція `print_pid2traffic()` створює DataFrame з інформацією про трафік для кожного процесу та оновлює Treeview в GUI новими даними;
- функція `print_stats()` періодично викликає `print_pid2traffic()`, якщо відображення статистики включено;

Користувач може вмикати або вимикати відображення підсумків пакетів і статистики трафіку через кнопки GUI.

```

220
221     display_frame = ttk.LabelFrame(root, text="Output")
222     display_frame.pack(padx=10, pady=10, fill="both", expand=True)
223
224     packet_summary_scroll = ttk.Scrollbar(display_frame)
225     packet_summary_scroll.pack(side=tk.RIGHT, fill=tk.Y)
226
227     packet_summary_text = tk.Text(display_frame, wrap=tk.WORD, yscrollcommand=packet_summary_scroll.set, height=10)
228     packet_summary_text.pack(padx=10, pady=10, fill="x", expand=True)
229     packet_summary_scroll.config(command=packet_summary_text.yview)
230
231     columns = ("PID", "Name", "Upload", "Download", "Upload Speed", "Download Speed")
232     stats_tree = ttk.Treeview(display_frame, columns=columns, show="headings")
233     for col in columns:
234         stats_tree.heading(col, text=col)
235         stats_tree.column(col, width=100)
236     stats_tree.pack(padx=10, pady=10, fill="both", expand=True)

```

Рисунок 3.15 – Реалізації відображення виводу

Джерело: запропоновано автором

3.3 Опис програмного забезпечення

Перед початком роботи з даною системою необхідно впевнитися, що на комп'ютері присутні мережеві інтерфейси, що вони підключені до мережі та правильно налаштовані. Також необхідно впевнитися, що існують клієнти які знаходяться в мережі.

Для запуску додатку існують обов'язкові умови:

- наявність мережевого інтерфейсу, що підтримує режим прослуховування мережі;
- версія nPcap не нижче 1.79;

Початок роботи з вебзастосунком:

1. Запуск програми (див. рисунок 3.16).
2. Вибір інтерфейсу:
 - 1) знайти розділ "Select Interface" (Виберіть інтерфейс) у верхній частині GUI (див. рисунок 3.17);
 - 2) відкрити меню, натиснувши на нього;
 - 3) вибрати мережевий інтерфейс зі списку (див. рисунок 3.18). Якщо інтерфейси не відображаються, переконатися, що мережеві інтерфейси на комп'ютері налаштовані правильно.
3. Запуск моніторингу:
 - 1) натиснути кнопку "Start Monitoring" (Почати моніторинг) для початку відстеження мережевого трафіку на вибраному інтерфейсі (див. рисунок 3.19);
 - 2) якщо інтерфейс невірний або не вибраний, у текстовому полі підсумку пакетів з'явиться повідомлення про помилку та не будуть доступні кнопки керування (див. рисунок 3.20).
4. Переключення відображення підсумків пакетів:
 - 1) після початку моніторингу кнопка "Toggle Packet Summary" (Переключити підсумки пакетів) стане активною (див. рисунок 3.21);

2) натиснути цю кнопку, щоб почати відображати підсумки пакетів у текстовому полі нижче (див. рисунок 3.22). Ще один натиск зупинить відображення підсумків пакетів;

3) у текстовому полі підсумку пакетів будуть відображатися повідомлення, що вказують, чи відображення підсумків пакетів увімкнене або вимкнене.

5. Переключити відображення статистики:

1) аналогічно, кнопка "Toggle Stats Display" (Переключити відображення статистики) стане активною після початку моніторингу;

2) натиснувши цю кнопку, почнеться відображення мережевої статистики для кожного процесу в дереві статистики нижче (див. рисунок 3.23). Ще один натиск зупинить оновлення статистики;

3) коли ввімкнене, статистика для кожного процесу (така як PID, ім'я процесу, завантажені та завантажені дані, швидкість завантаження і завантаження) буде періодично оновлюватися і відображатися в дереві.

6. Відображення результатів:

1) у текстовому полі підсумку пакетів (див. рисунок 3.34) можна переглядати підсумки пакетів у реальному часі тут, якщо увімкнено відображення підсумків пакетів. Тут будуть показані короткі деталі про кожен оброблений пакет.

2) у дереві статистики (див. рисунок 3.35) можна спостерігати детальну мережеву статистику для процесів у реальному часі, якщо ввімкнене відображення статистики. Відображені стовпці:

- PID (ідентифікатор процесу);
- Name (ім'я процесу);
- Upload (загальні завантажені дані);
- Download (загальні завантажені дані);
- Upload Speed (швидкість завантаження);
- Download Speed (швидкість завантаження).

3) прокручуйте дерево, щоб побачити всі процеси. Дерево буде автоматично оновлюватися з останніми даними.

7. Закриття програми:

- 1) натисніть кнопку закриття (X) у верхньому правому куті вікна, щоб вийти з програми;
- 2) це зупинить усі потоки і завершить роботу програми.

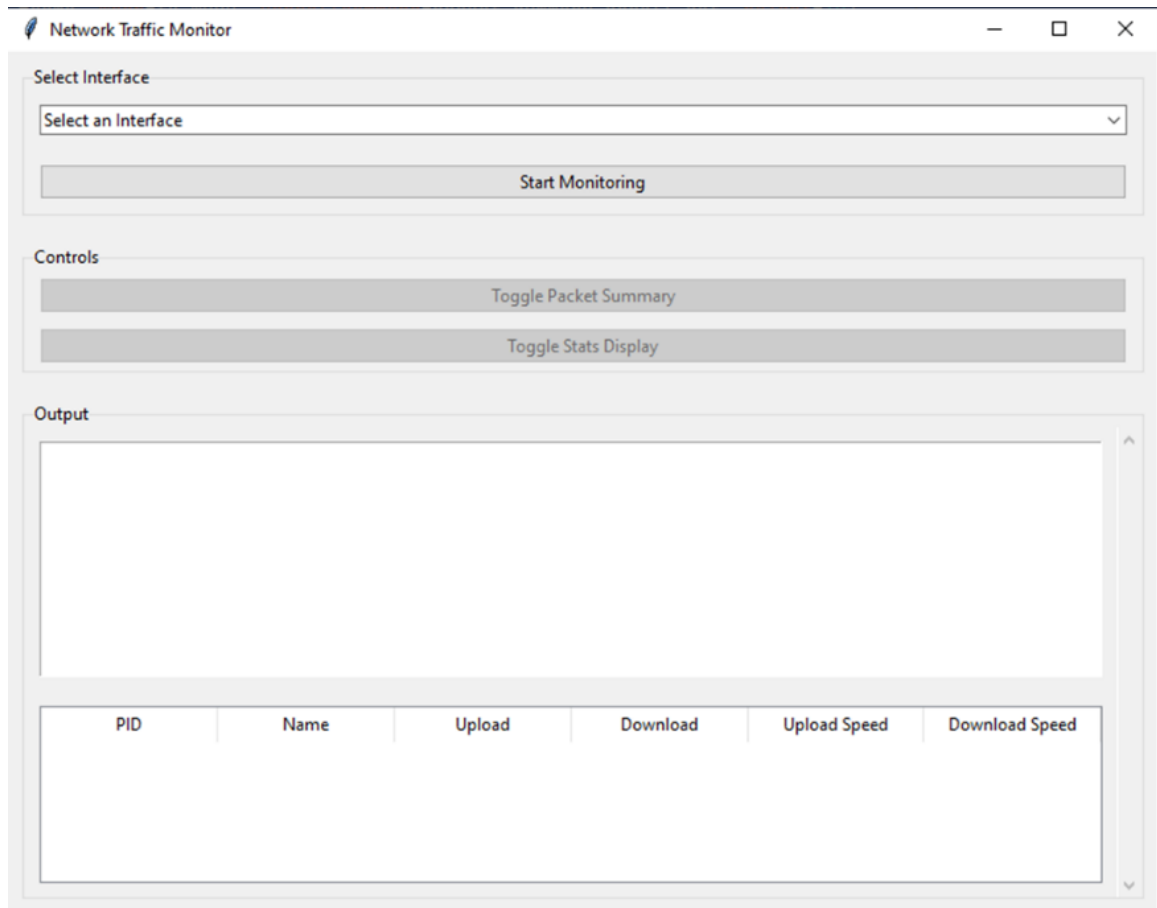


Рисунок 3.16 – Інтерфейс вебзастосунку

Джерело: запропоновано автором

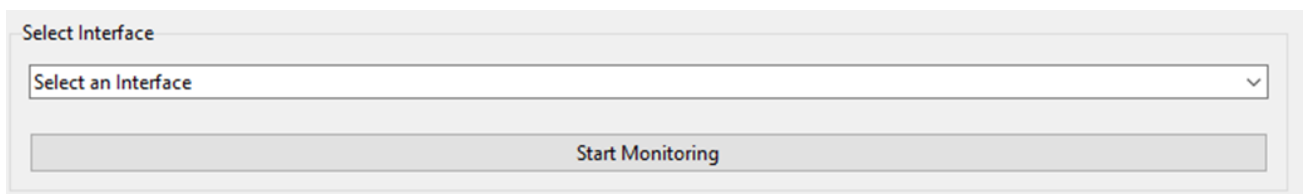


Рисунок 3.17 – Меню вибору інтерфейсів

Джерело: запропоновано автором

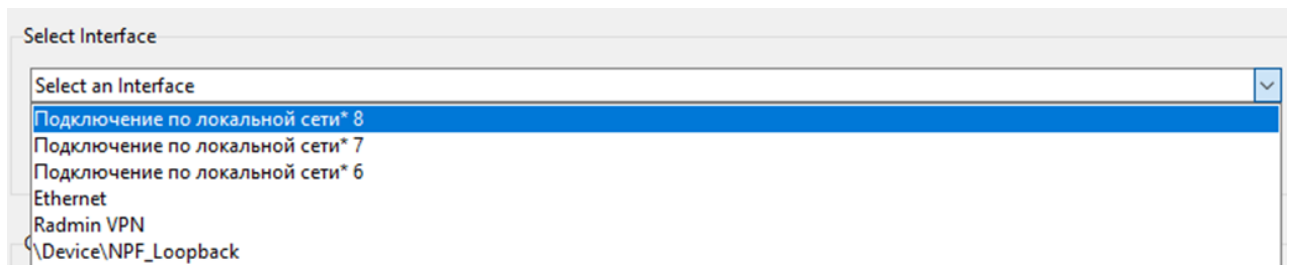


Рисунок 3.18 – Вибір інтерфейсу

Джерело: запропоновано автором

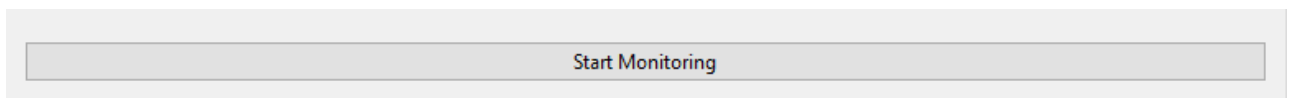


Рисунок 3.19 – Кнопка "Start Monitoring"

Джерело: запропоновано автором

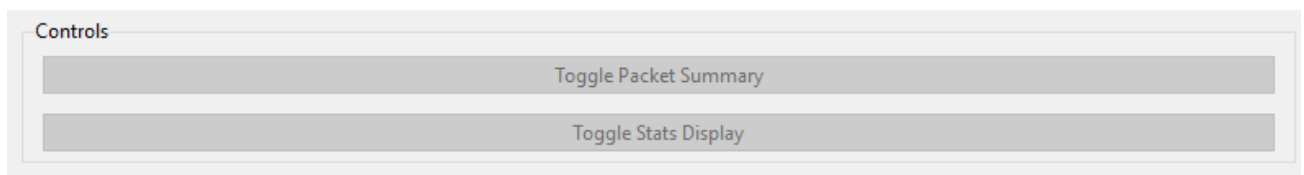


Рисунок 3.20 – Виключені кнопки керування

Джерело: запропоновано автором

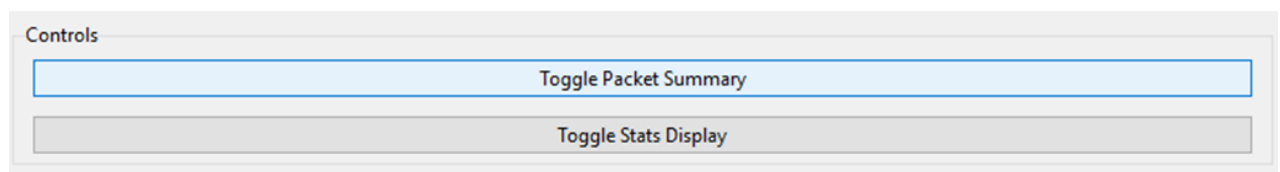


Рисунок 3.21 – Кнопки керування відображення статистики процесів та резюме пакетів

Джерело: запропоновано автором

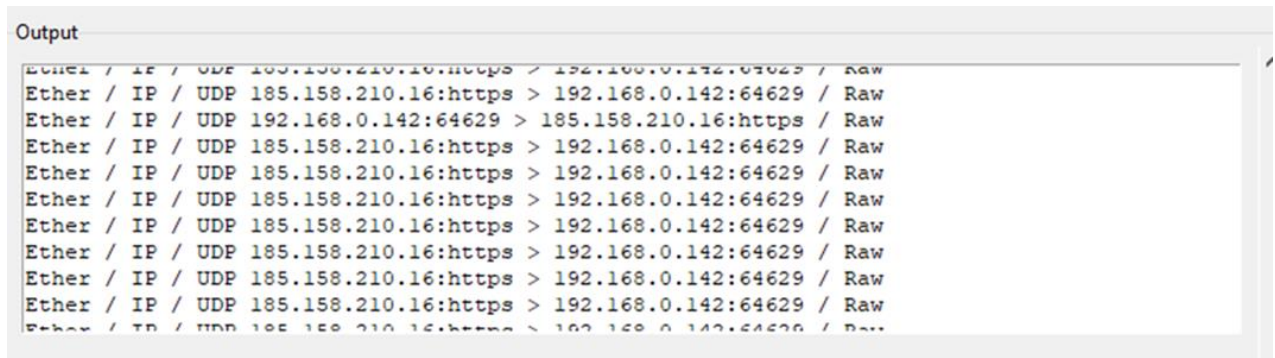


Рисунок 3.22 – Відображення підсумків пакетів

Джерело: запропоновано автором

PID	Name	Upload	Download	Upload Speed	Download Speed
2768	EpicGamesLauncher.e	6.37KB	10.05KB	0.00B/s	0.00B/s
3068	msedge.exe	5.80KB	9.63KB	328.00B/s	612.00B/s
22520	OverwolfBrowser.exe	1.02KB	1.11KB	0.00B/s	0.00B/s
19336	Discord.exe	993.00B	1013.00B	0.00B/s	0.00B/s
21888	WINWORD.EXE	54.00B	566.00B	0.00B/s	0.00B/s
20856	msedgewebview2.exe	493.00B	474.00B	0.00B/s	0.00B/s
12540	steam.exe	452.00B	240.00B	0.00B/s	0.00B/s
3748	RvControlSvc.exe	288.00B	180.00B	0.00B/s	0.00B/s
3632	DSASvc.exe	108.00B	145.00B	0.00B/s	0.00B/s
7160	DSAUpdateService.exe	108.00B	145.00B	0.00B/s	0.00B/s
15012	OneDrive.exe	0.00B	60.00B	0.00B/s	0.00B/s

Рисунок 3.23 – Відображення мережевої статистики

Джерело: запропоновано автором

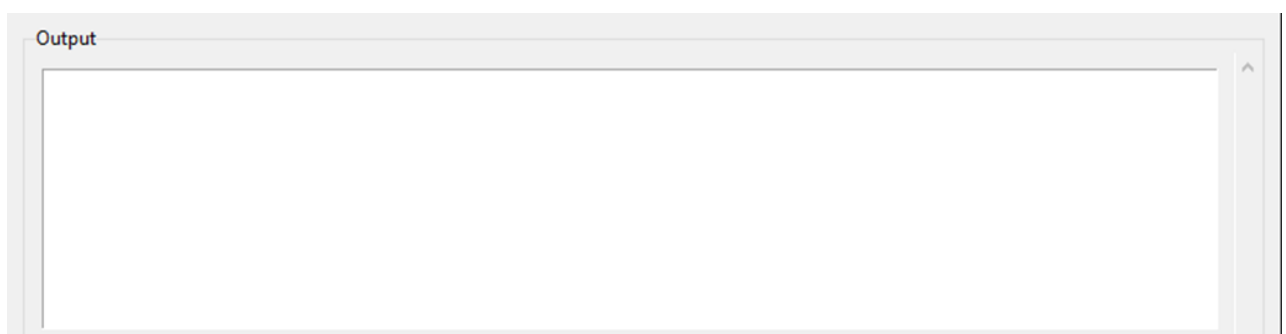
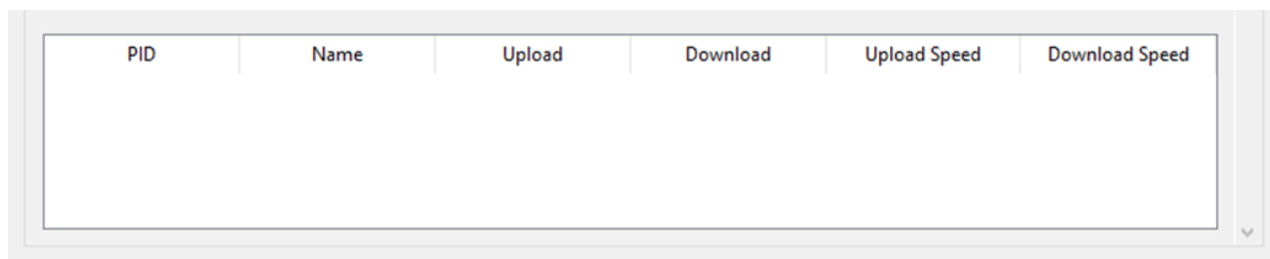


Рисунок 3.24 – Текстове поле підсумку пакетів

Джерело: запропоновано автором

A screenshot of a software window with a light gray border. Inside, there is a table with six columns. The first row contains the headers: 'PID', 'Name', 'Upload', 'Download', 'Upload Speed', and 'Download Speed'. The rest of the table is empty. A small downward arrow is visible on the right side of the table area.

PID	Name	Upload	Download	Upload Speed	Download Speed
-----	------	--------	----------	--------------	----------------

Рисунок 3.25 – Дерево статистики

Джерело: запропоновано автором

ВИСНОВКИ

У ході виконання даної роботи були досліджені основні методи програмного перехоплення пакетів даних та аналізу трафіку в локальній мережі. Створений вебзастосунок моніторить мережевий трафік на вибраному інтерфейсі, асоціюючи його з процесами на комп'ютері.

В кваліфікаційній роботі запропоновано варіант реалізації вебзастосунку для операційних систем Windows для моніторингу мережевого трафіку та вирішені такі задачі як:

- вибір оптимального методу;
- використання драйвера для перехоплення мережевих даних;

Подальші дослідження полягають у вдосконаленні та розширенні функціональних можливостей даного вебзастосунку. Можливі напрямки розвитку для розробленого вебзастосунку:

- створення бази даних для запису статистики процесів та резюме пакетів;
- вдосконалення інтерфейсу користувача;
- розширення можливостей захоплення трафіку.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Корецький В.А., Долгіх Я.В. Аналіз сучасних засобів виявлення мережесих атак. Технології ХХІ сторіччя: зб. тез за матеріалами 29ої міжнародної наук.-практ. конф. (20-22 листопада 2023 р.). Ч.2. Суми : СНАУ, 2023. С. 52-54.
2. Корецький В.А. Проектування та реалізація вебінформаційної підсистеми моніторингу інтернет-трафіку: матеріали науково-практичної конференції викладачів, аспірантів та студентів Сумського НАУ, 14-16 травня 2024 р. Суми: СНАУ. С.425.
3. Feldmann A. Internet traffic analysis at scale. Proceedings of the VLDB Endowment. 2021. Vol. 14, no. 13. P. 3415. URL: <https://doi.org/10.14778/3484224.3484237>.
4. Bhandari, Aishwarya, et al. "Packet Sniffing and Network Traffic Analysis Using TCP—A New Approach." Advances in Electronics, Communication and Computing. Springer, Singapore, 2018.
5. Explainable Internet Traffic Classification / C. Callegari et al. Applied Sciences. 2021. Vol. 11, no. 10. P. 4697. URL: <https://doi.org/10.3390/app111046974>.
6. Cardwell K. Capturing Network Traffic. Tactical Wireshark. Berkeley, CA, 2023. P. 27–51. URL: https://doi.org/10.1007/978-1-4842-9291-4_2.
7. Cardwell K. Customization of the Wireshark Interface. Tactical Wireshark. Berkeley, CA, 2023. P. 1–26. URL: https://doi.org/10.1007/978-1-4842-9291-4_1.
8. Guznova E. S., Kazantsev M. A. Zabbix monitoring system for analyzing and predicting events. Achievements of Modern Radioelectronics. 2023. URL: <https://doi.org/10.18127/j20700784-202312-17>.
9. B R., K V. Deploying Nagios for Service Monitoring and Reporting. International Journal of Innovative Research in Engineering. 2023. P. 39–41. URL: <https://doi.org/10.59256/ijire.20230405005>.
10. Held G. Working with NT's Performance Monitor. Local Area Network. 2020. P. 599–607. URL: <https://doi.org/10.1201/9781003069393-c51>.

11. A Systematic Approach of Analysing Network Traffic Using Packet Sniffing with Scapy Framework / S. H. Brahmanand et al. Computer Networks and Inventive Communication Technologies. Singapore, 2021. P. 811–820. URL: https://doi.org/10.1007/978-981-16-3728-5_60.
12. Unpingco J. Pandas. Python Programming for Data Analysis. Cham, 2021. P. 127–156. URL: https://doi.org/10.1007/978-3-030-68952-0_5.
13. Nguyen Q. Hands-On Application Development with Pycharm: Accelerate Your Python Applications Using Practical Coding Techniques in Pycharm. Packt Publishing, Limited, 2019.
14. Wireshark · Go Deep. Wireshark. URL: <https://www.wireshark.org/> (date of access: 22.05.2024).
15. Zabbix :: The Enterprise-Class Open Source Network Monitoring Solution. URL: <https://www.zabbix.com> (date of access: 22.05.2024).
16. Nagios Open Source | Nagios Open Source. Nagios Open Source. URL: <https://www.nagios.org> (date of access: 22.05.2024).
17. Cisco Security Products and Solutions. Cisco. URL: <https://www.cisco.com/site/us/en/products/security/index.html> (date of access: 22.05.2024).
18. Network Performance Monitor - Onsite & Remote Monitoring | SolarWinds. Observability and IT Management Platform | SolarWinds. URL: <https://www.solarwinds.com/network-performance-monitor> (date of access: 22.05.2024).
19. Cacti® - The Complete RRDTool-based Graphing Solution. Cacti® - The Complete RRDTool-based Graphing Solution. URL: <https://www.cacti.net> (date of access: 22.05.2024).
20. WinPcap: NPF driver internals manual. WinPcap - Home. URL: https://www.winpcap.org/docs/docs_412/html/group__NPF.html (date of access: 22.05.2024).

ДОДАТКИ

ДОДАТОК А

ТЕХНІЧНЕ ЗАВДАННЯ

Розробка вебзастосунку для моніторингу трафіку інтернет-провайдера

Погоджено:

Кандидат економічних наук, доцент кафедри
кібернетики та інформатики

Студент групи ІСТ 2201 с.т.

Корецький Василь Андрійович

СУМИ 2024

1. Призначення й мета створення вебзастосунку

1.1 Призначення вебзастосунку

Вебзастосунок призначений для моніторингу та підвищення ефективності використання трафіку.

1.2 Мета створення вебзастосунку

Розробка вебзастосунку моніторингу інтернет-трафіку провайдера для моніторингу та оптимізації використання трафіку.

1.3 Цільова аудиторія

Цільова аудиторія вебзастосунку це системні адміністратори.

2. Вимоги до вебзастосунку

2.1 Вимоги до вебзастосунку в цілому

2.1.1 Вимоги до структури й функціонування вебзастосунку

Вебзастосунок повинен бути реалізований у вигляді застосунку. Застосунок повинен складатися із розділів із чітко розділеними функціями.

2.1.2 Вимоги до персоналу

Від персоналу не має вимагатися особливих технічних навичок для підтримки й експлуатації вебзастосунку, окрім загальних навичок роботи з персональним комп'ютером.

2.2 Структура вебзастосунку

2.2.1 Навігація

На головній сторінці будуть представлені такі елементи:

- меню вибору інтерфейсів;
- кнопка «Start Monitoring»;
- кнопка «Toggle Stats Display»;
- кнопка «Toggle Packet Summary»;
- таблиця для запису статистики використання трафіку;
- поле для запису резюме пакетів

2.3 Вимоги до функціонування системи

2.3.1 Потреби користувача

Можливість зупинити та продовжити відображення статистики використання трафіку процесів та резюме пакетів. Також необхідна можливість вибору бажаного інтерфейсу для моніторингу трафіку.

2.3.2 Функціональні вимоги

На головній сторінці будуть представлені такі елементи:

- меню вибору інтерфейсів;
- таблиця для запису статистики використання трафіку;
- поле для запису резюме пакетів.

2.3.3 Системні вимоги

Системні вимоги відповідно до потреб користувачів:

- наявність вибору інтерфейсу моніторингу;
- таблиця для запису статистики використання трафіку;
- поле для запису резюме пакетів

2.4 Вимоги до видів забезпечення

2.4.1 Вимоги до інформаційного забезпечення

Реалізація вебзастосунку відбувається з використанням:

- Python;
- Pycharm;
- Scapy;
- Pandas.

2.4.2 Вимоги до лінгвістичного забезпечення

Вебзастосунок має бути виконаний англійською мовою.

2.4.3 Вимоги до програмного забезпечення

Програмне забезпечення клієнтської частини повинне задовольняти наступним вимогам:

- прсар 1.79 та вище;
- мережевий інтерфейс, що підтримує режим прослуховування мережі.

3. Склад і зміст робіт зі створення вебзастосунку

Докладний опис етапів роботи зі створення вебзастосунку наведено на рисунку А.1.

Режим задачі	Task Name	Тривалість	Початок	Завершення	Попередники	Імена ресурсів
★	Розробка вебзастосунку для моніторингу трафіку інтернет-провайдерів	95 днів	Пн 08.01.24	Пт 17.05.24		
★	Підготовка проєкту	5 днів	Пн 08.01.24	Пт 12.01.24		
★	Визначення стратегії проєкту	2 днів	Пн 08.01.24	Вт 09.01.24		Developer
★	Встановлення комунікаційних процесів	1 день	Ср 10.01.24	Ср 10.01.24	3	Tester
★	Аналіз конкурентів	3 днів	Ср 10.01.24	Пт 12.01.24	3	Developer
★	Визначення технічних вимог	2 днів	Чт 11.01.24	Пт 12.01.24	4	Tester
★	Збір вимог	16 днів	Пн 15.01.24	Пн 05.02.24		
★	Визначення вимог	6 днів	Пн 15.01.24	Пн 22.01.24	6	Developer
★	Розробка варіантів вимог	2 днів	Вт 23.01.24	Ср 24.01.24	8	Developer
★	Відстеження змін вимог	5 днів	Чт 25.01.24	Ср 31.01.24	9	Developer
★	Перевірка вимог на реалізованість	3 днів	Чт 01.02.24	Пн 05.02.24	10	Developer
★	Архітектура та дизайн	16 днів	Вт 06.02.24	Вт 27.02.24		
★	Розробка технічного проєкту	9 днів	Вт 06.02.24	Пт 16.02.24	11	Developer
★	Створення варіантів дизайну	4 днів	Пн 19.02.24	Чт 22.02.24	13	Developer
★	Розробка сценаріїв взаємодії	3 днів	Пт 23.02.24	Вт 27.02.24	14	Developer
★	Розробка	29 днів	Ср 28.02.24	Пн 08.04.24		
★	Написання коду	25 днів	Ср 28.02.24	Вт 02.04.24	15	Developer
★	Оптимізація продуктивності	4 днів	Ср 03.04.24	Пн 08.04.24	17	Developer
★	Тестування	16 днів	Вт 09.04.24	Вт 30.04.24		
★	Створення тест-кейсів	10 днів	Вт 09.04.24	Пн 22.04.24	6	Tester
★	Виконання функціональних тестів	2 днів	Вт 23.04.24	Ср 24.04.24	20	Tester
★	Проведення тестів з навантаженнями	4 днів	Чт 25.04.24	Вт 30.04.24	21	Tester
★	Підтримка	13 днів	Ср 01.05.24	Пт 17.05.24		
★	Проведення навчання та підтримки персоналу	1 день	Ср 01.05.24	Ср 01.05.24	22	Tester
★	Моніторинг і аналіз функціонування системи	8 днів	Чт 02.05.24	Пн 13.05.24	24	Tester
★	Виявлення та виправлення помилок	4 днів	Вт 14.05.24	Пт 17.05.24	25;18	Developer;Tester

Рисунок А.1 – Етапи створення вебзастосунку

Джерело: запропоновано автором

4. Вимоги до складу й змісту робіт із введення вебзастосунку в експлуатацію

Для того, щоб вебзастосунок могли користуватися робітники, необхідно на робочій станції користувача встановити прсар останньої версії та мати мережевий інтерфейс, що підтримує режим прослуховування мережі.

ДОДАТОК Б

ПЛАНУВАННЯ РОБІТ

Діаграма Ганта – це графічний інструмент, що використовується для візуалізації послідовності та тривалості завдань у проєкті. Вона названа на честь інженера Генріка Ганта, який вперше її застосував в сфері виробництва.

Мета діаграми Ганта – показати послідовність завдань і етапів проєкту в графічній формі. Це корисно для планування проєкту і для візуалізації взаємозв'язків і взаємодії між завданнями в часі.

Діаграма Ганта відображає часову шкалу горизонтально, а завдання представлені у вигляді прямокутників, розташованих на цій шкалі. Довжина прямокутників відповідає тривалості завдань, а їх положення показує їх послідовність. Також, за допомогою діаграми Ганта можна відображати залежності між завданнями, ресурси, виконавців та мілістоуні.

Основні елементи діаграми Ганта:

1) список завдань: зазвичай розташований зліва від діаграми. Кожен рядок представляє окреме завдання або етап проєкту.

2) шкала часу: розташована горизонтально в верхній частині діаграми. Вона може бути поділена на дні, тижні, місяці або інші часові інтервали в залежності від тривалості проєкту.

3) смуги завдань: горизонтальні смуги, що відображають тривалість кожного завдання. Початок смуги відповідає даті початку завдання, а кінець – даті завершення.

4) взаємозв'язки між завданнями: Лінії або стрілки, які показують залежності між завданнями, наприклад, які завдання повинні бути завершені перед початком інших.

Діаграма Ганта є важливим інструментом управління проєктами, який надає велику користь в розумінні послідовності робіт, визначенні тривалості завдань та

контролі прогресу проекту. Основні висновки, що можна зробити з використанням діаграми Ганта, включають:

- послідовність робіт: діаграма Ганта допомагає зорієнтуватися в послідовності завдань, необхідних для виконання проекту. Вона показує, які завдання потрібно виконати перед іншими, що допомагає зорієнтуватися в хронологічному порядку робіт;
- тривалість завдань: діаграма Ганта надає інформацію про тривалість кожного завдання, що дозволяє оцінити загальний час виконання проекту. Завдяки цьому можна планувати ресурси, визначати критичні завдання і визначати терміни виконання проекту;
- залежності між завданнями: діаграма Ганта візуалізує залежності між різними завданнями, показуючи, як одне завдання впливає на початок або завершення інших. Це допомагає визначити критичні шляхи та визначити, які завдання можуть вплинути на тривалість проекту;
- контроль прогресу: діаграма Ганта дозволяє відстежувати прогрес виконання завдань і проекту в цілому. Вона надає можливість позначати завдання, що виконані, та спостерігати за затримками або відхиленнями від плану.

На рисунках Б.1 – Б.3 наведена діаграма Ганта та використання завдань по створенню вебзастосунку для моніторингу трафіка інтернет-провайдера.

Сформований проєкт надасть значну кількість інформації про систему, оскільки він описує загальні функції системи. Користувачі, представники клієнтів, менеджери проєктів, аналітики, розробники, фахівці з контролю якості і всі, хто зацікавлений у системі.

Функції можна легко описати простою мовою. Описи функцій складаються з коротких фраз, і рідко коли функції детально розробляються. Кожна функція пов'язана з елементами даних, які надають додаткову інформацію про цю функцію. Ці елементи даних називаються атрибутами функції.

Режим задачі	Task Name	Тривалість	Початок	Завершення	Попередники	Імена ресурсів
★	Розробка вебзастосунку для моніторингу трафіку інтернет-провайдерів	95 днів	Пн 08.01.24	Пт 17.05.24		
★	Підготовка проекту	5 днів	Пн 08.01.24	Пт 12.01.24		
★	Визначення стратегії проекту	2 днів	Пн 08.01.24	Вт 09.01.24		Developer
★	Встановлення комунікаційних процесів	1 день	Ср 10.01.24	Ср 10.01.24	3	Tester
★	Аналіз конкурентів	3 днів	Ср 10.01.24	Пт 12.01.24	3	Developer
★	Визначення технічних вимог	2 днів	Чт 11.01.24	Пт 12.01.24	4	Tester
★	Збір вимог	16 днів	Пн 15.01.24	Пн 05.02.24		
★	Визначення вимог	6 днів	Пн 15.01.24	Пн 22.01.24	6	Developer
★	Розробка варіантів вимог	2 днів	Вт 23.01.24	Ср 24.01.24	8	Developer
★	Відстеження змін вимог	5 днів	Чт 25.01.24	Ср 31.01.24	9	Developer
★	Перевірка вимог на реалізованість	3 днів	Чт 01.02.24	Пн 05.02.24	10	Developer
★	Архітектура та дизайн	16 днів	Вт 06.02.24	Вт 27.02.24		
★	Розробка технічного проекту	9 днів	Вт 06.02.24	Пт 16.02.24	11	Developer
★	Створення варіантів дизайну	4 днів	Пн 19.02.24	Чт 22.02.24	13	Developer
★	Розробка сценаріїв взаємодії	3 днів	Пт 23.02.24	Вт 27.02.24	14	Developer
★	Розробка	29 днів	Ср 28.02.24	Пн 08.04.24		
★	Написання коду	25 днів	Ср 28.02.24	Вт 02.04.24	15	Developer
★	Оптимізація продуктивності	4 днів	Ср 03.04.24	Пн 08.04.24	17	Developer
★	Тестування	16 днів	Вт 09.04.24	Вт 30.04.24		
★	Створення тест-кейсів	10 днів	Вт 09.04.24	Пн 22.04.24	6	Tester
★	Виконання функціональних тестів	2 днів	Вт 23.04.24	Ср 24.04.24	20	Tester
★	Проведення тестів з навантаженнями	4 днів	Чт 25.04.24	Вт 30.04.24	21	Tester
★	Підтримка	13 днів	Ср 01.05.24	Пт 17.05.24		
★	Проведення навчання та підтримки персоналу	1 день	Ср 01.05.24	Ср 01.05.24	22	Tester
★	Моніторинг і аналіз функціонування системи	8 днів	Чт 02.05.24	Пн 13.05.24	24	Tester
★	Виявлення та виправлення помилок	4 днів	Вт 14.05.24	Пт 17.05.24	25;18	Developer;Tester

Рисунок Б.1 – Інформаційна частина діаграми Ганта

Джерело: запропоновано автором

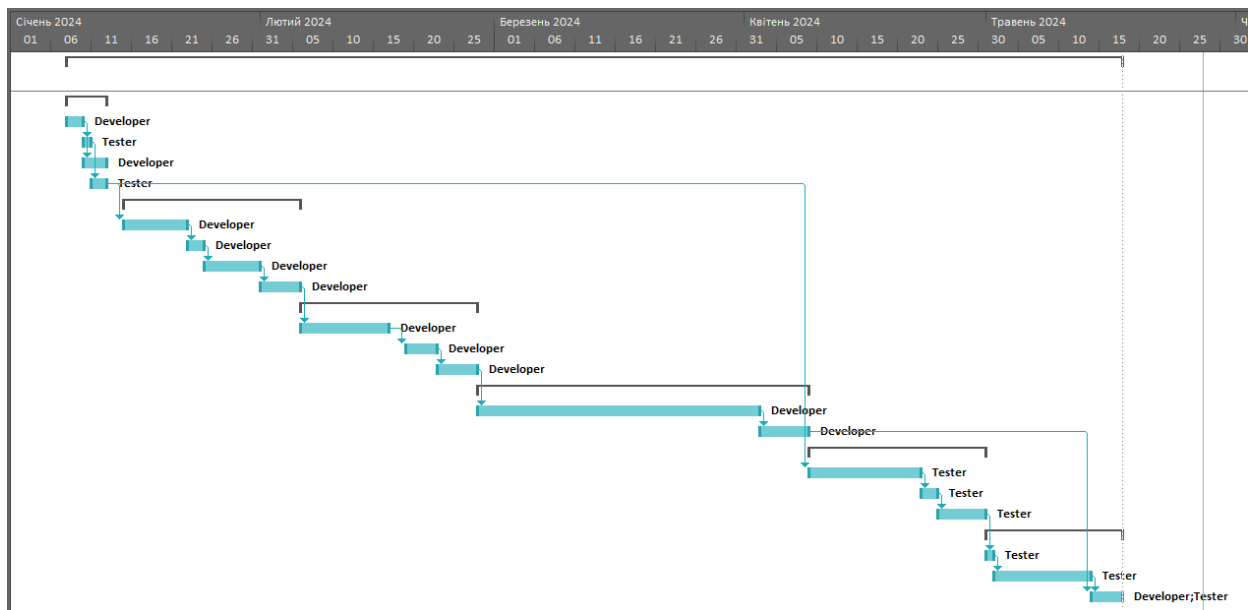


Рисунок Б.2 – Діаграма Ганта

Джерело: запропоновано автором

Рішення Завдання	Назва задачі	Робота	Тривалість	Початок	Завершення	Логі новий ст.	Відомі	Г	Ч	Півріччя 2, 2023							Півріччя 1, 2024							Півріччя 2, 2024						
										А	С	В	Ж	Л	Г	С	Л	В	К	Т	Ч	А	С	В	Ж	Л	Г			
✚	• Розробка вебасосунку для моніторингу трафіку інтернет-провайдера		816 год, 95 днів	Пн 08.01.24	Пн 17.05.24		Робот:										168r	168r	168r	178r	136r									
✚	• Підготовка проекту		64 год, 5 днів	Пн 08.01.24	Пн 12.01.24		Робот:										64r													
✚	• Визначення стратегії проекту	Developer	16 год, 2 днів	Пн 08.01.24	Вт 09.01.24		Робот:										16r													
✚	• Встановлення комунікаційних процесів	Developer	16 год	Пн 08.01.24	Вт 09.01.24		Робот:										16r													
✚	• Встановлення комунікаційних процесів	Developer	8 год, 1 день	Ср 10.01.24	Ср 10.01.24		Робот:										8r													
✚	• Tester		8 год	Ср 10.01.24	Ср 10.01.24		Робот:										8r													
✚	• Аналіз конкурентів	Developer	24 год, 3 днів	Ср 10.01.24	Пн 12.01.24		Робот:										24r													
✚	• Визначення технічних вимог	Developer	24 год	Ср 10.01.24	Пн 12.01.24		Робот:										24r													
✚	• Визначення технічних вимог	Tester	16 год, 2 днів	Чт 11.01.24	Пн 12.01.24		Робот:										16r													
✚	• Визначення технічних вимог	Tester	16 год	Чт 11.01.24	Пн 12.01.24		Робот:										16r													
✚	• Збір вимог		128 год, 16 днів	Пн 15.01.24	Пн 05.02.24		Робот:										104r	24r												
✚	• Визначення вимог	Developer	48 год, 6 днів	Пн 15.01.24	Пн 22.01.24		Робот:										48r													
✚	• Визначення вимог	Developer	48 год	Пн 15.01.24	Пн 22.01.24		Робот:										48r													
✚	• Розробка варіантів вимог	Developer	16 год, 2 днів	Вт 23.01.24	Ср 24.01.24		Робот:										16r													
✚	• Відстеження змін вимог	Developer	16 год	Вт 23.01.24	Ср 24.01.24		Робот:										16r													
✚	• Відстеження змін вимог	Developer	40 год, 5 днів	Чт 25.01.24	Ср 31.01.24		Робот:										40r													
✚	• Відстеження змін вимог	Developer	40 год	Чт 25.01.24	Ср 31.01.24		Робот:										40r													
✚	• Перевірка вимог на реалізованість	Developer	24 год, 3 днів	Чт 01.02.24	Пн 05.02.24		Робот:											24r												
✚	• Перевірка вимог на реалізованість	Developer	24 год	Чт 01.02.24	Пн 05.02.24		Робот:											24r												
✚	• Архітектура та дизайн		128 год, 16 днів	Вт 06.02.24	Вт 27.02.24		Робот:										128r													
✚	• Розробка технічного проекту	Developer	72 год, 9 днів	Вт 06.02.24	Пн 16.02.24		Робот:										72r													
✚	• Розробка технічного проекту	Developer	72 год	Вт 06.02.24	Пн 16.02.24		Робот:										72r													
✚	• Створення варіантів дизайну	Developer	32 год, 4 днів	Пн 19.02.24	Чт 22.02.24		Робот:										32r													
✚	• Створення варіантів дизайну	Developer	32 год	Пн 19.02.24	Чт 22.02.24		Робот:										32r													
✚	• Розробка сценаріїв взаємодії	Developer	24 год, 3 днів	Пн 23.02.24	Вт 27.02.24		Робот:										24r													
✚	• Розробка сценаріїв взаємодії	Developer	24 год	Пн 23.02.24	Вт 27.02.24		Робот:										24r													
✚	• Розробка		232 год, 29 днів	Ср 28.02.24	Пн 08.04.24		Робот:										16r	168r		48r										
✚	• Написання коду		200 год, 25 днів	Ср 28.02.24	Вт 02.04.24		Робот:										16r	168r		16r										
✚	• Написання коду	Developer	200 год	Ср 28.02.24	Вт 02.04.24		Робот:										16r	168r		16r										
✚	• Оптимізація продуктивності	Developer	32 год, 4 днів	Ср 03.04.24	Пн 08.04.24		Робот:														32r									
✚	• Оптимізація продуктивності	Developer	32 год	Ср 03.04.24	Пн 08.04.24		Робот:														32r									
✚	• Тестування		128 год, 16 днів	Вт 09.04.24	Вт 30.04.24		Робот:														128r									

Рисунок Б.3 – Використання завдань

Джерело: запропоновано автором

Атрибути використовуються для зв'язування функцій з іншими функціями та інформацією в проекті. В переліку атрибутів немає жорстких обмежень, тому розробник формує їх на свій розсуд.

Управління ризиками проекту – це процес виявлення, аналізу та реагування на будь-який ризик, що виникає протягом життєвого циклу проекту. Крім того, управління ризиками є частиною процесу планування, що необхідно для своєчасного з'ясування можливих ризиків, і контролю цих ризиків, якщо вони дійсно виникають. Ризик – це все, що потенційно може вплинути на терміни, продуктивність або бюджет проекту. Процес ідентифікації ризиків передбачає встановлення та документування характеристик ризиків, які можуть вплинути на проект. Цей процес є ітеративним, адже з розвитком проекту можуть з'являтися нові ризики, що потребуватимуть ідентифікації та реагування.

Перелічимо можливі ризики проекту, визначимо відносну критичність, а також можливі способи їх уникнення. Введемо умовну шкалу рівнів критичності ризиків (див. табл. Б.1):

Таблиця Б.1 – Рівні критичності ризиків

1	Незначний
2	Низький
3	Середній
4	Високий
5	Критичний

Джерело: запропоновано автором

Ризики проекту, визначені в результаті ідентифікації, а також їх потенційний вплив та ймовірність виникнення наведені у таблиці Б.2.

Таблиця Б.2 – Ризики проекту

№	Опис	Ймовірність	Вплив
1	Втрата актуальності проекту	1	4
2	Зміна вимог на пізніх етапах розробки	2	4
3	Незаплановані роботи, що потребують обов'язкового виконання	2	3
4	Вимушене зміщення термінів виконання завдань або паралельне виконання цих завдань	3	3
5	Нездійснені або важкі задачі	3	5
6	Пошкодження або відсутність обладнання	1	5
7	Непрацездатність розробника	1	5
8	Низьке залучення замовника	2	4

Джерело: запропоновано автором

Тепер, складемо матрицю залежності втрат від ймовірності виникнення кожного із зазначених ризиків (див. таблицю Б.3).

Таблиця Б.3 – Матриця «Ймовірність – Втрати»

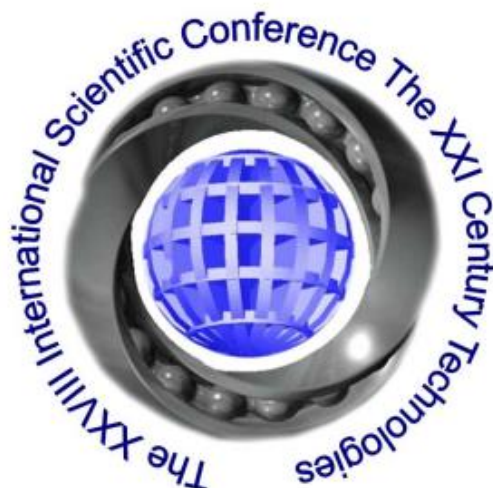
Ймовірність\Втрати	1	2	3	4	5
1				1	6,7
2			3	2,8	
3			4		5
4					
5					

Джерело: запропоновано автором

ДОДАТОК В

КОПІЇ ПУБЛІКАЦІЙ

	Сумський національний аграрний університет
	Національний технічний університет «ХПІ»
	Політехніка Свентокржинська в Кельцах (Польща)
	ТОВ «ТРІЗ»
	Сумський державний університет
	Державний біотехнологічний університет
	Українська технологічна академія



ТЕХНОЛОГІЇ ХХІ СТОРІЧЧЯ

Збірник тез за матеріалами 29^{ої} міжнародної науково-практичної конференції
(20-22 листопада 2023 р.)

Частина 2

Секції: «Прогресивні технології в економіці»,
«Прогресивні технології в екології та енергоефективність»,
«Прогресивні технології у педагогіці»

Суми – 2023

<i>Куліш Д.О., магістрант, СНАУ</i> СВІТОВИЙ ДОСВІД І ПРАКТИЧНІ НАПРАЦЮВАННЯ В ОБЛАСТІ ОЦІНКИ ЗЕМЕЛЬНИХ УГІДЬ СІЛЬСЬКОГОСПОДАРСЬКОГО ПРИЗНАЧЕННЯ	34
<i>Макарова В.В., д.е.н., професор, СНАУ</i> СТРАТЕГІЧНІ ОРІЄНТИРИ ЕКОЛОГО-БЕЗПЕЧНОГО РОЗВИТКУ СІЛЬСЬКОГОСПОДАРСЬКОГО ЗЕМЛЕКОРИСТУВАННЯ	37
<i>Конєва І.І., к.е.н., доцент, СНАУ, Суми, Україна</i> РОЗВИТОК VIP-БАНКІНГУ В УКРАЇНІ В УМОВАХ ВОЄННОГО СТАНУ	40
<i>Amola Adegoke Olayinka, Bachelor's degree student, Mohylna L. M., Ph.D in Economics, Associate Professor, SNAU, Sumy, Ukraine</i> MANAGING THE PROMOTION OF SERVICES AT THE ENTERPRISE OF DEUTSCHE POST DHL GROUP	42
<i>Баталова А.Б., старший викладач, СНАУ, Суми, Україна</i> ЗАСТОСУВАННЯ ЛІНІЙНИХ МОДЕЛЕЙ МАТЕМАТИЧНОГО ПРОГРАМУВАННЯ ПРИ РОЗВ'ЯЗУВАННІ ЗАДАЧ ЕКОНОМІКИ	43
<i>Долгіх Я.В., доцент, СНАУ, м. Суми</i> АНАЛІЗ ЕФЕКТИВНОСТІ ОСВІТНЬОЇ ДІЯЛЬНОСТІ ЗАКЛАДІВ ВИЩОЇ ОСВІТИ МЕТОДОМ DEA	45
<i>Кривокоритов П.М., бакалаврант, СНАУ, Могильна Л.М., к.е.н., доцент, СНАУ</i> УПРАВЛІННЯ ЯКІСТЮ ПРОДУКЦІЇ НА ДЕРЕВООБРОБНОМУ ПІДПРИЄМСТВІ	46
<i>Линник В.О., магістрант, СНАУ, Суми, Україна</i> ТЕОРЕТИЧНІ АСПЕКТИ УПРАВЛІННЯ КОНКУРЕНТОСПРОМОЖНІСТЮ ПІДПРИЄМСТВА	47
<i>Петренко О.С., магістрант, Могильна Л.М., к.е.н., доцент, СНАУ, Суми, Україна</i> ТЕОРЕТИЧНІ АСПЕКТИ УПРАВЛІННЯ ПЕРСОНАЛОМ В ПРИВАТНОМУ ПІДПРИЄМСТВІ	48
<i>Биченко Є.В., бакалаврант, Долгіх Я.В., к.е.н., доцент, СНАУ</i> ДЕЯКІ АСПЕКТИ ЗАСТОСУВАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ АВТОМАТИЗАЦІЇ ПРОЦЕСІВ УПРАВЛІННЯ ДІЯЛЬНІСТЮ ПІДПРИЄМСТВ	49
<i>Корецький В.А., бакалаврант, Долгіх Я.В., к.е.н., доцент, СНАУ, м. Суми</i> АНАЛІЗ СУЧАСНИХ ЗАСОБІВ ВИЯВЛЕННЯ МЕРЕЖЕВИХ АТАК	52
<i>Липченко І.В., бакалаврант, Долгіх Я.В., к.е.н., доцент, СНАУ, м. Суми</i> ДЕЯКІ АСПЕКТИ ЗАСТОСУВАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ НАВЧАННЯ	54
<i>Ярмоленко М.В., магістрант, СНАУ, Суми, Україна</i> СТРАТЕГІЧНІ АЛЬТЕРНАТИВИ РОЗВИТКУ АГРАРНИХ ПІДПРИЄМСТВ	55
<i>Хоменко Р.М., бакалаврант, Долгіх Я.В., к.е.н., доцент, СНАУ, м. Суми</i> ДЕЯКІ АСПЕКТИ ТЕХНОЛОГІЇ РОЗРОБКИ ЧАТ-БОТУ ДЛЯ МЕСЕНДЖЕРУ ТЕЛЕГРАМ ПО КУПІВЛІ ТА ПРОДАЖУ СПОЖИВАНОЇ ЕЛЕКТРОТЕХНІКИ	57
<i>Устік Т.В., д.е.н., професорка, Сумський національний аграрний університет, Суми, Україна</i> ЗМІЦНЕННЯ ПРОДОВОЛЬЧОЇ БЕЗПЕКИ ЯК ІНСТРУМЕНТ РЕАЛІЗАЦІЇ ПОТЕНЦІАЛУ КОНКУРЕНТОСПРОМОЖНОГО РОЗВИТКУ РЕГІОНУ	59
<i>Хромушина Л.А., к.е.н., доцент, СНАУ, Суми, Україна</i> ПЕРСПЕКТИВИ РОЗВИТКУ УПРАВЛІННЯ ЛЮДСЬКИМ КАПІТАЛОМ В УМОВАХ ЦИФРОВІЗАЦІЇ СУСПІЛЬСТВА	61
<i>Юр'єв Д.Ю., магістрант, СНАУ, Суми, Україна</i> ПЛАНУВАННЯ МАРКЕТИНГОВОЇ ДІЯЛЬНОСТІ ПІДПРИЄМСТВА В МЕЖАХ ОРГАНІЗАЦІЙНИХ РІВНІВ УПРАВЛІННЯ	62
<i>Кознієнко Олена, магістрант, СНАУ, Суми, Україна</i> АРХІТЕКТОНІКА СКЛАДОВИХ СИСТЕМИ УПРАВЛІННЯ ІМІДЖЕМ АГРАРНОГО ПІДПРИЄМСТВА	64

5. Kenan A., Ihsan T. M. Efficiency of electronic document management systems: a case study. *Science, Education and Innovations in the context of modern problems*. 2022. Vol. 5(3). Pp. 75. URL: <http://surl.li/mlggt>.
6. Wiencierz C., Röttger U., Fuhrmann C. Agile Cooperation between Communication Agencies and Companies. *International Journal of Strategic Communication*. 2021. Vol. 15, no. 2. Pp. 144-158. doi: 10.1080/1553118x.2021.1898144.
7. Madanhire I., Mbohwa Ch. Enterprise resource planning (ERP) in improving operational efficiency: Case study. 13th Global Conference on Sustainable Manufacturing - Decoupling Growth from Resource Use. *Procedia CIRP* 40. 2016. Pp. 225-229. doi: 10.1016/j.procir.2016.01.108.
8. Kryvstun T. V., Slabinoha M. O., Zayachuk Y. I. Web-oriented System for Monitoring and Management of Agile Methodology Projects. *Methods and Devices of Quality Control*. 2021. No. 1(46). Pp. 132-137. doi: 10.31471/1993-9981-2021-1(46)-132-137.
9. Колесніков К. В., Гарячий В. С. Особливості проектування web-орієнтованих інформаційних систем на основі гнучкої методології Scrum. *Вісник Черкаського державного технологічного університету*. 2020. № 4. С. 36-41. doi: 10.24025/2306-4412.4.2019.182763.
10. Мельничук Ю. Розвиток алгоритмів створення веб-орієнтованих інформаційних систем. *Наука і техніка сьогодні*. 2023. № 2(16). doi:10.52058/2786-6025-2023-2(16)-392-400.
11. Encyclopedia of GIS / ed. by S. Shekhar, H. Xiong, X. Zhou. Cham : Springer International Publishing, 2020. doi: 10.1007/978-3-319-23519-6.
12. Ali A. Assessment of Groundwater Quality using Geographical Information System: A Case Study of Faisalabad, Pakistan. *Journal of Environmental & Agricultural Sciences*. 2021. Vol. 23, no. 1&2. Pp. 30-35. URL: <http://surl.li/mlhcr>.
13. Geoinformation Technologies in Support of Environmental Hazards Monitoring under Climate Change: An Extensive Review / A. Tsatsaris et al. *ISPRS International Journal of Geo-Information*. 2021. Vol. 10, no. 2. Pp. 94. doi: 10.3390/ijgi10020094.
14. Sokolova N., Petryha M., Bulana T. Time Management Web -oriented Information System Based on the Mern Technology Stack. *Computer systems and information technologies*. 2022. No. 4. Pp. 91-100. doi: 10.31891/csit-2022-4-12.
15. Kulawiak M., Dawidowicz A., Pacholczyk M. E. Analysis of server-side and client-side Web-GIS data processing methods on the example of JTS and JSTS using open data from OSM and geoportal. *Computers & Geosciences*. 2019. Vol. 129. Pp. 26-37. doi: 10.1016/j.cageo.2019.04.011.
16. Олійник Н., Мокрицька Г., Рошин І. Застосування інформаційних технологій у сучасному менеджменті. *Measuring and Computing Devices in Technological Processes*. 2021. № 1. С. 105-112. doi: 10.31891/2219-9365-2021-67-1-15.
17. Trofymchuk O., Nesterenko O., Netesin I. Methodology for Designing Analytical Information Systems for Administrative Management. *Science and Innovation*. 2022. Vol. 18(4). Pp. 25-40. doi: 10.15407/scine18.04.025.

Корецький В.А., бакалаврант, Долгих Я.В., к.е.н., доцент, СНАУ, м. Суми

АНАЛІЗ СУЧАСНИХ ЗАСОБІВ ВИЯВЛЕННЯ МЕРЕЖЕВИХ АТАК

Постановка проблеми. Постійний розвиток і широке використання Інтернету приносить користь багатьом користувачам мережі з багатьох аспектів. Широке використання мережі обумовлює важливість її безпеки, а саме захисту мережі від несанкціонованого доступу та модифікації.

Зростаюча кількість підключених до Інтернету систем у фінансах, електронній комерції та військових сферах робить їх об'єктами мережеских атак, що призводить до великого ризику та шкоди. Мережескі атаки є серйозною загрозою для інформаційної безпеки. Сучасні засоби виявлення мережеских атак є важливим елементом захисту комп'ютерних мереж,

оскільки можуть надати допомогу організаціям виявити мережеві атаки та запобігти їм, перш ніж вони призведуть до збитків. Отже ця тема є актуальною.

Викладення основного матеріалу. Сучасні системи виявлення мережевих атак необхідні для захисту організацій від широкого спектру кіберзагроз. Ці системи використовують різноманітні методи для ідентифікації та блокування зловмисної активності, зокрема виявлення на основі сигнатур, евристичних методів, аномалій, машинного та глибокого навчання.

Сучасні засоби виявлення атак можна розділити на два основних типи:

- системи виявлення вторгнень (Intrusion Detection Systems, IDS);
- системи запобігання вторгненням (Intrusion Prevention Systems, IPS).

Системи запобігання вторгнень (IPS) – це безпекові пристрої або програмні рішення, які мають на меті виявлення та блокування неправомірних атак або спроб вторгнення до комп'ютерних систем чи мереж. IPS використовують різні методи для ідентифікації та зупинення потенційно шкідливих дій користувачів або програм.

Системи виявлення вторгнень (IDS) – це засоби комп'ютерної безпеки, призначені для виявлення неправомірних або шкідливих активностей в комп'ютерних системах чи мережах. IDS моніторять трафік в мережі чи активність в системі та сповіщають адміністраторів або вживають інших заходів у разі виявлення аномалій чи потенційно небезпечних дій.

Виявлення на основі сигнатур є найбільш традиційним методом виявлення мережевих атак. Цей метод передбачає зіставлення мережевого трафіку з відомими сигнатурами атак. Сигнатури атак – це унікальні шаблони даних, пов'язані з певними типами атак.

Виявлення на основі евристики є більш складним методом виявлення мережевих атак. Цей метод використовує евристичні методи, або емпіричні правила, для виявлення підозрілої активності в мережевому трафіку. Евристика базується на досвіді та знанні типових шаблонів атак.

Виявлення на основі аномалій – це інший підхід до виявлення мережевих атак. Цей метод визначає мережевий трафік, який відхиляється від звичайних шаблонів. Системи виявлення аномалій можуть бути ефективними для виявлення нових і невідомих атак, але вони також можуть бути схильні до помилкових спрацювань.

Виявлення на основі машинного навчання є новішим підходом до виявлення мережевих атак. Цей метод використовує алгоритми машинного навчання для навчання моделей ідентифікації мережевих атак. Алгоритми машинного навчання можна навчити на великих наборах даних поміченого мережевого трафіку, що дозволяє їм вивчати складні шаблони, які людям важко ідентифікувати. У роботі [1] запропоновано обрати датасети KDD CUP99 та NSL-KDD для проведення порівняльних експериментів про дерева рішень, наївний Байєс, опорні векторні машини, випадкові ліси, XGBoost, згорткові нейронні мережі та рекурентні нейронні мережі.

Виявлення на основі глибокого навчання – це підмножина машинного навчання, яке використовує глибокі нейронні мережі для вивчення складних моделей мережевого трафіку. Системи виявлення на основі глибокого навчання потенційно можуть бути ефективнішими, ніж традиційні системи виявлення мережевих атак, але вони складніші й дорожчі для навчання та розгортання. Порівняння та аналіз методів виявлення мережевих атак на основі глибокого навчання проведено у роботах [2-6].

У статті [7] пропонується метод виявлення атак LDoS на основі багатофункціональної нейронної мережі (CNN). У цьому методі обчислюються різні характеристики мережі, які об'єднують у карту функцій. Карта функцій використовується для характеристики стану мережі.

За останні роки дослідження щодо застосування методів глибокого навчання для виявлення атак досягли значного прогресу. Однак, багато проблем все ще існує. Серед основних проблем можна відмітити наступні:

- 1) складно модифікувати методи глибокого навчання як класифікатори в реальному часі для виявлення атак;

2) більшість методів глибокого навчання підходять для аналізу розпізнавання зображень і образів. Отже, цікавим буде питання, як розумно провести класифікацію мережевого трафіку за допомогою методів глибокого навчання;

3) з більшою кількістю даних, що включають експерименти, результати класифікації будуть кращими. Однак, для більшості проблем виявлення атак бракує достатньої кількості даних.

Отже, поєднання навчання під контролем і без нього може забезпечити кращу продуктивність, що було доведено багатьма випробуваннями.

Дослідження в галузі виявлення мережевих атак активно проводяться в усьому світі. Вчені розробляють нові методи виявлення атак, які є більш ефективними, точними та економічними. Аналіз сучасних засобів виявлення мережевих атак є важливим завданням, яке дозволяє організаціям забезпечити ефективний захист від мережевих атак.

ЛІТЕРАТУРА

1. Comparative Research on Network Intrusion Detection Methods Based on Machine Learning / C. Zhang et al. *Computers & Security*. 2022. P. 102861. doi:10.1016/j.cose.2022.102861.
2. Wu Y., Wei D., Feng J. Network Attacks Detection Methods Based on Deep Learning Techniques: 2. A Survey. *Security and Communication Networks*. 2020. Vol. 2020. P. 1-17. doi: 10.1155/2020/8872923.
3. Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study / M. A. Ferrag et al. *Journal of Information Security and Applications*. 2020. Vol. 50. P. 102419. doi: 10.1016/j.jisa.2019.102419.
4. Deep Learning Approach for Intelligent Intrusion Detection System / R. Vinayakumar et al. *IEEE Access*. 2019. Vol. 7. P. 41525-41550. doi: 10.1109/access.2019.2895334.
5. Vinayakumar R., Soman K. P., Poornachandran P. Evaluating effectiveness of shallow and deep networks to intrusion detection system. 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Udupi, 13-16 September 2017. 2017. doi: 10.1109/icacci.2017.8126018.
6. Network Intrusion Detection: Based on Deep Hierarchical Network and Original Flow Data / Y. Zhang et al. *IEEE Access*. 2019. Vol. 7. P. 37004-37016. doi: 10.1109/access.2019.2905041.
7. MF-CNN: a New Approach for LDoS Attack Detection Based on Multi-feature Fusion and CNN / D. Tang et al. *Mobile Networks and Applications*. 2020. doi: 10.1007/s11036-019-01506-1.

Липченко І.В., бакалаврант, Долгих Я.В., к.е.н., доцент, СНАУ, м. Суми

ДЕЯКІ АСПЕКТИ ЗАСТОСУВАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ НАВЧАННЯ

Постановка проблеми. В теперішній час інформаційні технології мають значний вплив на систему освіти. Пандемія COVID-19, воєнний стан в Україні стали чинниками впровадження дистанційної освіти, що ще більше ініціювало розвиток цифрових технологій в освіті. Інформаційні технології суттєво вплинули на зміну якості всієї системи освіти. Інформаційні технології навчання є постачальником знань, співавтором інформації, наставником, оцінювачем та тренажером. Отже, дослідження теоретичних та практичних аспектів застосування інформаційних технологій в навчанні є актуальним.

Викладення основного матеріалу. Аналіз наукових робіт щодо застосування інформаційних технологій (далі – ІТ) навчання надає можливість краще розуміти позитивні та негативні наслідки використання ІТ в освіті, з'ясувати практичні аспекти впровадження цих технологій у навчальний процес. В роботі [1] проводиться дослідження щодо необхідності застосування цифрових технологій в освіті, охарактеризовано основні застосування для навчання та обговорюються можливі виклики в освіті. Автори визначили наступні напрямки застосування ІТ навчання, що висвітлюються у наукових дослідженнях: підвищення продук-

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

**СУМСЬКИЙ НАЦІОНАЛЬНИЙ
АГРАРНИЙ УНІВЕРСИТЕТ**

МАТЕРІАЛИ

науково-практичної конференції
викладачів, аспірантів та студентів
Сумського НАУ

(14-16 травня 2024 р.)

Luo Hao. A STUDY ON CONSUMER BEHAVIOUR MANAGEMENT THROUGH ADVANCED MARKETING TOOLS	414
Wang Baichao. MANAGEMENT CRISIS IN AGRICULTURAL COMPANIES IN CURRENT SITUATIONS	415
Wang Ruijie. HUMAN RESOURCES MANAGEMENT OPTIMIZATION AND INNOVATION BASED ON ARTIFICIAL INTELLIGENCE	416
Yu Hang. RESEARCH ON THE INTRODUCTION OF MODERN MANAGEMENT TECHNOLOGY INTO ENTERPRISES	417
Zhou Kunming. PECULIARITIES OF MANAGING HIGH-TECH ENTERPRISES IN A TRANSPARENT ECONOMY	418
Бусенко І.В. АНАЛІЗ ОСНОВНИХ РІЗНОВИДІВ SQL-ІН'ЄКЦІЙ, АНАЛІТИКА ЗАСТОСУВАННЯ ТА СПОСОБИ ЗАХИСТУ ВІД НИХ	419
Белоконь П.А. РОЗРОБКА ІНФОРМАЦІЙНОЇ ПІДСИСТЕМИ ПІДТРИМКИ РОБОТИ HR-МЕНЕДЖЕРА	420
Стеблянко В. В. РОЗРОБКА ВЕБ-ЗАСТОСУНКУ ОБЛІКУ ВИТРАТ НА ВИРОБНИЦТВО ПРОДУКЦІЇ. РЕАЛІЗАЦІЯ ПРОЄКТУ	421
Фідірко Т.Р. РОЗРОБКА ВЕБЗАСТОСУНКУ ОБЛІКУ ТЕХНІЧНОГО СТАНУ ЛЕГКОВИХ АВТОМОБІЛІВ. ПОСТАНОВКА ЗАДАЧІ	422
Фідірко Т.Р. РОЗРОБКА ВЕБЗАСТОСУНКУ ОБЛІКУ ТЕХНІЧНОГО СТАНУ ЛЕГКОВИХ АВТОМОБІЛІВ. РЕАЛІЗАЦІЯ ПРОЄКТУ	423
Биченко Є.В. РОЗРОБКА ПРОТОТИПУ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ ІНТЕРФЕЙСУ КОРИСТУВАЧА ВЕБОРІЄНТОВАНОЇ ІНФОРМАЦІЙНОЇ СИСТЕМИ ПІДТРИМКИ УПРАВЛІННЯ ДІЯЛЬНІСТЮ ПІДПРИЄМСТВА	424
Корецький В.А. ПРОЕКТУВАННЯ ТА РЕАЛІЗАЦІЯ ВЕБІНФОРМАЦІЙНОЇ ПІДСИСТЕМИ МОНІТОРИНГУ ІНТЕРНЕТ-ТРАФІКУ	425
Лапченко І. В. ТА РЕАЛІЗАЦІЯ ВЕБЗАСТОСУНКУ ОРГАНІЗАЦІЇ ДИСТАНЦІЙНИХ КУРСІВ НЕФОРМАЛЬНОЇ ОСВІТИ	426
Хоменко Р.М. ПРОЕКТУВАННЯ ІНТЕРФЕЙСУ КОРИСТУВАЧА ЧАТ-БОТА ДЛЯ ЗРУЧНИХ ТА ЕФЕКТИВНИХ ОНЛАЙН-ЗАМОВЛЕНЬ ЕЛЕКТРОНІКИ	427
Блаженко Д. О. ПРОБЛЕМИ ПРОЕКТУВАННЯ БАЗИ ДАНИХ ПІДСИСТЕМИ ВЕДЕННЯ ЗАМОВЛЕНЬ ДЛЯ ВИГОТОВЛЕННЯ СІЛЬСЬКОГОСПОДАРСЬКОЇ ПРОДУКЦІЇ	428
Ковальов Р. С. ПРАКТИЧНЕ ПРОГРАМУВАННЯ ГЕНЕРАТОРІВ 3D КОНТЕНТУ НА ПРИКЛАДІ ГЕНЕРАТОРУ МЕША РЕЛЬЄФУ МІСЦЕВОСТІ	429
Ковальов Р. С. ПРАКТИЧНЕ ПРОГРАМУВАННЯ ГЕНЕРАТОРІВ 3D КОНТЕНТУ НА ПРИКЛАДІ ГЕНЕРАТОРУ МЕША РЕЛЬЄФУ МІСЦЕВОСТІ	430
Руденок О. А. МІНІМІЗАЦІЯ ЗАТРИМОК У ВЕБДОДАТКАХ ВИКОРИСТОВУЮЧИ ГРАНИЧНІ ТЕХНОЛОГІЇ	431
Сіденко А. С. ПРОБЛЕМИ РЕАЛІЗАЦІЇ ПІДСИСТЕМИ ОБЛІКУ ПРИРОДООХОРОННИХ ЗАХОДІВ	432
Терещенко С.С. АВТОМАТИЗАЦІЯ ОБЛІКОВИХ ПРОЦЕСІВ: КЛЮЧ ДО ЕФЕКТИВНОГО УПРАВЛІННЯ В АТЕЛЬЄ З ВИГОТОВЛЕННЯ ОДЯГУ	433
Михайлюк О.С. ОГРУНТУВАННЯ НЕОБХІДНОСТІ РОЗРОБКИ WEB-ОРІЄНТОВАНОЇ ПІДСИСТЕМИ ПІДТРИМКИ ОБЛІКОВИХ ПРОЦЕСІВ РЕГІОНАЛЬНОГО ІНТЕРНЕТ-ПРОВАЙДЕРА	434
Олійник А.В. ПРОЕКТУВАННЯ ТА РЕАЛІЗАЦІЯ ПІДСИСТЕМИ ОБЛІКУ ПАЛИВНО-МАСТИЛЬНИХ МАТЕРІАЛІВ В ТЕРИТОРІАЛЬНІЙ ГРОМАДІ	435
Чубун М.П. АНАЛІЗ ТА ВИЗНАЧЕННЯ ВИМОГ ДО СУЧАСНОГО ВЕБ-САЙТУ	436
Дубінчин В. С. ВИЗНАЧЕННЯ ФУНКЦІОНАЛЬНИХ ВИМОГ ДО ІНФОРМАЦІЙНОЇ ПІДСИСТЕМИ ОЦІНЮВАННЯ ПРАЦЕЗДАТНОСТІ КОМП'ЮТЕРИЗОВАНИХ РОБОЧИХ МІСЦЬ	437
Санжаков А. А. НОВИЙ ПІДХІД РОЗРОБКИ ДИЗАЙНУ ВЕБ-ЗАСТОСУНКІВ – MATERIAL 3	438
Федірко Д.Д. ПРОБЛЕМИ ПРОЕКТУВАННЯ ВЕБ-ЗАСТОСУНКІВ ОЦІНКИ ТАРИФНИХ ПРОПОЗИЦІЙ МОБІЛЬНИХ ОПЕРАТОРІВ УКРАЇНИ	439
Федірко Д.Д. РОЗРОБКА ВЕБ-ЗАСТОСУНКУ ДЛЯ АНАЛІЗУ МОБІЛЬНИХ ОПЕРАТОРІВ УКРАЇНИ	440
Маслак О. М. ДЕРЖАВНЕ РЕГУЛЮВАННЯ ПРОЦЕСІВ АГРОПРОМИСЛОВОГО ВИРОБНИЦТВА НА РЕГІОНАЛЬНОМУ РІВНІ (НА ПРИКЛАДІ СУМСЬКОЇ ОБЛАСТІ)	441
Горбань С.В. ІМІДЖ ПІДПРИЄМСТВА: ВПЛИВ, ФОРМУВАННЯ ТА ОЦІНКА	442
Заболотна А.І. НОРМАТИВНО-ПРАВОВЕ ЗАБЕЗПЕЧЕННЯ ЕКОНОМІЧНОГО РОЗВИТКУ ДРУЖІВСЬКОЇ ТЕРИТОРІАЛЬНОЇ ГРОМАДИ	443
Гулинніна О.Ю. ШЛЯХИ ОПТИМІЗАЦІЇ ОРГАНІЗАЦІЙНОЇ СТРУКТУРИ УПРАВЛІННЯ ЗАКЛАДУ ОХОРОНИ ЗДОРОВ'Я	444
Сущенко В. О. ПРОЦЕС РОЗВИТКУ ПЕРСОНАЛУ, ЯК ЧИННИК КОНКУРЕНТОЗДАТНОСТІ ПІДПРИЄМСТВА В РИНКОВИХ УМОВАХ	445
Каплун В.М. УПРАВЛІННЯ СІЛЬСЬКОГОСПОДАРСЬКИХ ПІДПРИЄМСТВ ІЗ ЗАСТОСУВАННЯ ПРОГРАМНОГО КОНТРОЛЮ І МОНІТОРИНГУ	446
Бондар А.В. ТЕОРЕТИКО-МЕТОДИЧНІ ПІДХОДИ ДО ВПРОВАДЖЕННЯ ТА ОЦІНКИ СТРАТЕГІЧНОГО УПРАВЛІННЯ РОЗВИТКОМ СІЛЬСЬКОГОСПОДАРСЬКИХ ПІДПРИЄМСТВ	447
Романенко Т.А. СУЧАСНІ НАПРЯМИ ІНТЕНСИФІКАЦІЇ ВИРОБНИЦТВА ПРОДУКЦІЇ РОСЛИННИЦТВА В КОНТЕКСТІ ПРОДОВОЛЬЧОЇ БЕЗПЕКИ	448
Мішенін Є. В., Ярова І. Є. ОРГАНІЗАЦІЙНО ЕКОНОМІЧНИЙ МЕХАНІЗМ ЗАБЕЗПЕЧЕННЯ СТАЛОГО ПРОСТОРОВОГО РОЗВИТКУ ЛІСОВОГО ГОСПОДАРСТВА	449
Hu Guannan CRISIS MANAGEMENT THEORY	450
Zhang Liming HOW DOES BYD COMPANY IMPROVE ITS COMPETITIVENESS?	451
Zhao Jing THE INFLUENCE OF CORPORATE CULTURE ON EMPLOYEE SATISFACTION	452
Cui Xiaoping DEVELOPMENT PATH OF ENTERPRISE MANAGEMENT IN THE ERA OF ARTIFICIAL INTELLIGENCE	453

ПРОЕКТУВАННЯ ТА РЕАЛІЗАЦІЯ ВЕБІНФОРМАЦІЙНОЇ ПІДСИСТЕМИ МОНІТОРИНГУ ІНТЕРНЕТ-ТРАФІКУ

Корецький В.А., студ. 2 ФЕІМ, спец. «Інформаційні системи та технології»
Науковий керівник: к.е.н., доц. Я.В. Долгих
Сумський НАУ

Постановка проблеми. Оскільки Інтернет-трафік невідомо зростає, визначення того, який хост і яка програма генерує обсяг мережевого трафіку стає необхідним для ефективного керування та використання мережевих ресурсів. Корпоративне обчислювальне середовище стає все більш орієнтованим на мережу, важливість моніторингу та аналізу мережевого трафіку зростає.

Викладення основного матеріалу. Моніторинг мережевого трафіку надає дані, за якими здійснюється аналіз. Необроблені дані про стан отримують шляхом перевірки мережевих пакетів, а аналіз надає розширені дані на основі необроблених даних.

Для того, щоб ефективно використовувати обмежені мережеві ресурси, мережеві менеджери повинні отримувати точну та надійну інформацію зі своїх мереж, таку як:

- скільки трафіку передається;
- який тип трафіку передається;
- скільки трафіку генерується з якої системи;
- яка система чи програма спричиняє вузькі місця;
- наскільки високий пік трафіку та коли він досягає піку.

Моніторинг Інтернет-трафіку може бути здійснений за допомогою різних інструментів мережевого моніторингу, які дозволяють: відстежувати, аналізувати та управляти інтернет-трафіком для забезпечення ефективності, безпеки та надійності мережі. Приклади реалізації цих інструментів запропоновані у статтях [1-4].

Якщо мережеві менеджери не можуть дати надійні відповіді на ці питання, цінні ресурси мережі будуть витрачені даремно. Розроблено низку автоматизованих інструментів, які дозволяють мережевим адміністраторам контролювати та аналізувати мережевий трафік. Прикладами таких інструментів є Multi-Router Traffic Grapher (MRTG), Argus, Etherfind, NFSwatch і TCPdump.

Інтернет і корпоративні мережі підтримують нові можливості для бізнесу. Як наслідок, – на сучасні мережі з більшою ймовірністю буде націлено напади. Різні атаки та їх динамічний характер вимагають гнучкого підходу для забезпечення мережевої безпеки. Для точного виявлення широкого спектру нападів необхідна гнучка система безпеки. Методи виявлення вторгнень є корисним способом виявлення загроз у системах виявлення вторгнень у цій ситуації. Виявлення аномалій забезпечує техніку визначення потенційних ризиків шляхом постійного моніторингу та моделювання звичайної активності мережі [5].

У виявленні аномалій необхідним є моніторинг трафіку, який вказує на серйозні, рідкісні події. Наприклад, незвичайна поведінка мережевого трафіку може означати, що комп'ютер зазнав атаки та дані надсилаються неавторизованими пунктами. Аномалії можна віднести до трьох категорій [6], включаючи точкову аномалію, контекстну аномалію та колективну аномалію. Це різні види аномалій мають зв'язок з атаками на захист мережі, включаючи DoS, Probe, U2R і R2L [6].

Література

1. La Lau R. Backup and Monitoring. Practical Internet Server Configuration. Berkeley, CA, 2021. P. 367-374. URL: https://doi.org/10.1007/978-1-4842-6960-2_15 (date of access: 07.04.2024).
2. S. Ayoubi et al., "Machine learning for cognitive network management," IEEE Commun. Mag., vol. 56, no. 1, pp. 158-165, Jan. 2018
3. P. W. Tsai, C. W. Tsai, C. W. Hsu, and C. S. Yang, "Network monitoring in software-defined networking: A review," IEEE Syst. J., vol. 12, no. 4, pp. 3958-3969, Dec. 2018.
4. W. Queiroz, M. A. Capretz, and M. Dantas, "An approach for SDN traffic monitoring based on big data techniques," J. Netw. Comput. Appl., vol. 131, pp. 28-39, 2019
5. Monitoring Network Traffic for Suspicious Activity / B. Ravi Teja et al. International Journal of Advanced Research in Science, Communication and Technology. 2023. P. 422-429. URL: <https://doi.org/10.48175/ijarsct-9153> (date of access: 07.04.2024).
6. Wang H., Gu J., Wang S. An effective intrusion detection framework based on SVM with feature augmentation. Knowledge-Based Systems. 2017. Vol. 136. P. 130-139. URL: <https://doi.org/10.1016/j.knosys.2017.09.014> (date of access: 07.04.2024).

ДОДАТОК Г

ЛІСТИНГИ ПРОГРАМ

```
import tkinter as tk
from tkinter import ttk
from scapy.all import *
import psutil
from collections import defaultdict
from threading import Thread, Event
import pandas as pd
import time
from datetime import datetime

all_macs = {iface.mac for iface in ifaces.values()}
connection2pid = {}
pid2traffic = defaultdict(lambda: [0, 0])
global_df = None
is_program_running = True
display_packet_summary = False
print_stats_enabled = False
print_stats_thread = None
command_event = Event()

def get_size(bytes):
    for unit in ['', 'K', 'M', 'G', 'T', 'P']:
        if bytes < 1024:
            return f"{bytes:.2f}{unit}B"
        bytes /= 1024

def process_packet(packet):
    global pid2traffic, display_packet_summary
    if display_packet_summary:
```

```

    packet_summary_text.insert(tk.END, packet.summary() + "\n")
    packet_summary_text.see(tk.END)
try:
    packet_connection = (packet.sport, packet.dport)
except (AttributeError, IndexError):
    pass
else:
    packet_pid = connection2pid.get(packet_connection)
    if packet_pid:
        if packet.src in all_macs:
            pid2traffic[packet_pid][0] += len(packet)
        else:
            pid2traffic[packet_pid][1] += len(packet)

def get_connections():
    global connection2pid
    while is_program_running:
        if command_event.is_set():
            return
        for c in psutil.net_connections():
            if c.laddr and c.raddr and c.pid:
                connection2pid[(c.laddr.port, c.raddr.port)] = c.pid
                connection2pid[(c.raddr.port, c.laddr.port)] = c.pid
        time.sleep(1)

def print_pid2traffic():
    global global_df
    processes = []
    for pid, traffic in pid2traffic.items():
        if command_event.is_set():
            return
        try:
            p = psutil.Process(pid)
        except psutil.NoSuchProcess:

```

```

        continue
    name = p.name()
    try:
        create_time = datetime.fromtimestamp(p.create_time())
    except OSError:
        create_time = datetime.fromtimestamp(psutil.boot_time())
    process = {
        "pid": pid, "name": name, "create_time": create_time, "Upload": traffic[0],
        "Download": traffic[1],
    }
    try:
        process["Upload Speed"] = traffic[0] - global_df.at[pid, "Upload"]
        process["Download Speed"] = traffic[1] - global_df.at[pid, "Download"]
    except (KeyError, AttributeError):
        process["Upload Speed"] = traffic[0]
        process["Download Speed"] = traffic[1]
    processes.append(process)
df = pd.DataFrame(processes)
try:
    df = df.set_index("pid")
    df.sort_values("Download", inplace=True, ascending=False)
except KeyError as e:
    pass
global_df = df

for row in stats_tree.get_children():
    stats_tree.delete(row)
for index, row in df.iterrows():
    stats_tree.insert("", "end", values=(index, row["name"], get_size(row["Upload"]),
                                         get_size(row["Download"]),
                                         get_size(row["Upload Speed"]) + "/s",
                                         get_size(row["Download Speed"]) + "/s"))

def print_stats():
    global stats_thread_started

```

```

stats_thread_started.set()
while is_program_running:
    if print_stats_enabled:
        print_pid2traffic()
    if command_event.wait(timeout=1):
        return

```

```

def toggle_packet_summary():
    global display_packet_summary
    display_packet_summary = not display_packet_summary
    state = "enabled" if display_packet_summary else "disabled"
    packet_summary_text.insert(tk.END, f"Packet summary display {state}.\n")
    packet_summary_text.see(tk.END)

```

```

def toggle_print_stats():
    global print_stats_enabled, print_stats_thread, stats_thread_started
    if print_stats_enabled:
        print_stats_enabled = False
        if print_stats_thread is not None and print_stats_thread.is_alive():
            command_event.set()
            print_stats_thread.join()
            command_event.clear()
    else:
        print_stats_enabled = True
        if print_stats_thread is None or not print_stats_thread.is_alive():
            command_event.clear()
            stats_thread_started = Event()
            print_stats_thread = Thread(target=print_stats)
            print_stats_thread.start()
            stats_thread_started.wait()
        command_event.clear()

```

```

def start_sniffing(interface):
    try:
        sniff(prn=process_packet, store=False, iface=interface, stop_filter=lambda x: command_event.is_set())
    except Exception as e:
        packet_summary_text.insert(tk.END, f"Exception in sniffing thread: {e}\n")
        packet_summary_text.see(tk.END)

def handle_user_input():
    global is_program_running, print_stats_enabled, display_packet_summary
    while is_program_running:
        cmd = input("Enter command (psum to display packet summary, stats to display stats printing,"
                    " p to stop packet summary display, s to stop stats printing): \n")
        print("You entered:", cmd, end='\n')
        if cmd == "psum":
            toggle_packet_summary()
        elif cmd == "stats":
            toggle_print_stats()
        elif cmd == "p":
            if display_packet_summary:
                display_packet_summary = False
                print("Packet summary display stopped.")
            else:
                print("Packet summary display is not currently enabled.")
        elif cmd == "s":
            print_stats_enabled = False
            if print_stats_thread is not None and print_stats_thread.is_alive():
                command_event.set()
                print_stats_thread.join()
                command_event.clear()
                print("Stats printing stopped.")
            else:
                print("Stats printing is not currently running.")
        else:
            print("Unknown command.")

```

```

def start_monitoring():
    global chosen_interface
    chosen_interface = interface_var.get()
    if chosen_interface not in [iface_info.name for iface_info in ifaces.values()]:
        packet_summary_text.insert(tk.END, "This interface does not exist. Please choose a valid interface.\n")
        packet_summary_text.see(tk.END)
    return

connections_thread = Thread(target=get_connections)
connections_thread.start()

sniffing_thread = Thread(target=start_sniffing, args=(chosen_interface,))
sniffing_thread.start()

toggle_packet_summary_button.config(state=tk.NORMAL)
toggle_print_stats_button.config(state=tk.NORMAL)

if __name__ == "__main__":

    root = tk.Tk()
    root.title("Network Traffic Monitor")
    root.geometry("800x600")

    interface_frame = ttk.LabelFrame(root, text="Select Interface")
    interface_frame.pack(padx=10, pady=10, fill="x")

    interface_var = tk.StringVar(value="Select an Interface")
    interfaces = [iface_info.name for iface_info in ifaces.values()]
    interface_menu = ttk.Combobox(interface_frame, textvariable=interface_var, values=interfaces,
state="readonly")
    interface_menu.pack(padx=10, pady=10, fill="x")

    start_button = ttk.Button(interface_frame, text="Start Monitoring", command=start_monitoring)
    start_button.pack(padx=10, pady=10, fill="x")

```

```

control_frame = ttk.LabelFrame(root, text="Controls")
control_frame.pack(padx=10, pady=10, fill="x")

toggle_packet_summary_button = ttk.Button(control_frame, text="Toggle Packet Summary",
command=toggle_packet_summary, state=tk.DISABLED)
toggle_packet_summary_button.pack(padx=10, pady=5, fill="x")

toggle_print_stats_button = ttk.Button(control_frame, text="Toggle Stats Display",
command=toggle_print_stats, state=tk.DISABLED)
toggle_print_stats_button.pack(padx=10, pady=5, fill="x")

display_frame = ttk.LabelFrame(root, text="Output")
display_frame.pack(padx=10, pady=10, fill="both", expand=True)

packet_summary_scroll = ttk.Scrollbar(display_frame)
packet_summary_scroll.pack(side=tk.RIGHT, fill=tk.Y)

packet_summary_text = tk.Text(display_frame, wrap=tk.WORD,
yscrollcommand=packet_summary_scroll.set, height=10)
packet_summary_text.pack(padx=10, pady=10, fill="x", expand=True)
packet_summary_scroll.config(command=packet_summary_text.yview)

columns = ("PID", "Name", "Upload", "Download", "Upload Speed", "Download Speed")
stats_tree = ttk.Treeview(display_frame, columns=columns, show="headings")
for col in columns:
    stats_tree.heading(col, text=col)
    stats_tree.column(col, width=100)
stats_tree.pack(padx=10, pady=10, fill="both", expand=True)

root.mainloop()

is_program_running = False
command_event.set()
if print_stats_thread is not None and print_stats_thread.is_alive():
    print_stats_thread.join()

```