ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Розробка системи моніторингу мережевого трафіку»

на здобуття освітнього ступеня бакалавра
зі спеціальності 123 Комп'ютерної інженерії
(код, найменування спеціальності)
освітньо-професійної програми Комп'ютерна інженерія
(назва)

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ *Гліб МАЛЬЧЕНКО*
(підпис) *Ім'я, ПРІЗВИЩЕ здобувача*

Виконав: здобувач вищої освіти гр. КІД-41
Гліб МАЛЬЧЕНКО
Ім'я, ПРІЗВИЩЕ
Керівник: Ігор БУЧЕНКО
науковий ступінь, вчене звання Ім'я, ПРІЗВИЩЕ
Рецензент: _____
науковий ступінь, вчене звання Ім'я, ПРІЗВИЩЕ

Київ 2024

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут Інформаційних технологій

Кафедра 123 Комп'ютерної інженерії

Ступінь вищої освіти бакалавр

Спеціальність 123 Комп'ютерної інженерії

Освітньо-професійна програма Комп'ютерна інженерія

ЗАТВЕРДЖУЮ
Завідувач кафедрою Наталія ЛАЩЕВСЬКА
_____Ім'я, ПРІЗВИЩЕ
«_____» _____ 2024р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Мальченко Гліб Сергійович
(прізвище, ім'я, по батькові здобувача)

1. Тема кваліфікаційної роботи: Розробка системи моніторингу мережевого трафіку
керівник кваліфікаційної роботи Ігор БУЧЕНКО,
(Ім'я, ПРІЗВИЩЕ, науковий ступінь, вчене звання)
затверджені наказом Державного університету інформаційно-комунікаційних технологій від «27» лютого 2024р. № 36
2. Строк подання кваліфікаційної роботи «3» червня 2024р.
3. Вихідні дані до кваліфікаційної роботи: Мережі, мережеві протоколи, C++, моніторинг мережі, Wireshark, Qt, Pcap, PyTorch, машинне навчання, методи аналізу мережі, протокольні та потокові аналізатори, пакет, протокол, порти, SolarWinds, PRTG Network Monitor
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Основи мережевого трафіку
2. Моніторинг мережевого трафіку
3. Розробка системи моніторингу мережевого трафіку
5. Перелік ілюстративного матеріалу: *презентація*
6. Дата видачі завдання «27» лютого 2024р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Збір даних	28.02-24.03.2024	Виконано
2	Обробка матеріалу	23.03-05.04.2024	Виконано
3	Розробка теоретичної частини	06.04-10.04.2024	Виконано
4	Аналіз технологій	11.04-20.04.2024	Виконано
5	Висновки за результатами аналізу	21.04-06.05.2024	Виконано
6	Оформлення пояснювальної записки	07.05-08.05.2024	Виконано
7	Розробка презентації	09.05-11.05.2024	Виконано
8	Передзахист	14.05-17.05.2024	Виконано
9	Здача в деканат	27.05-3.06.2024	Виконано

Здобувач вищої освіти _____ Гліб МАЛЬЧЕНКО
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

Керівник кваліфікаційної роботи _____ Ігор БУЧЕНКО
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня бакалавра: 41 стор., 9 рис., 35 джерел.

Мета роботи – Аналіз та вдосконалення існуючих рішень для моніторингу мережевого трафіку

Об'єкт дослідження – Моніторинг мережевого трафіку

Предмет дослідження – Розробка системи моніторингу мережевого трафіку

Короткий зміст роботи: У сучасному світі інформаційних технологій мережевий трафік відіграє ключову роль у функціонуванні бізнес-процесів, комунікації та передачі даних. З кожним днем обсяг мережевого трафіку зростає, що ставить перед адміністраторами мережі та інженерами завдання забезпечення ефективності, надійності та безпеки мережевих інфраструктур. Одним з основних інструментів для досягнення цих цілей є системи моніторингу мережевого трафіку. Ці системи дозволяють адміністраторам мережі відслідковувати та аналізувати роботу мережі в реальному часі, виявляти аномалії та проблеми, а також приймати вчасні рішення для їх вирішення.

За допомогою поєднання різних методів моніторингу та аналізу мережевого трафіку було розроблено нову систему моніторингу, яка поєднує в собі сильні сторони різних методів, зменшуючи їхні недоліки.

ЗМІСТ

ВСТУП…… 8
РОЗДІЛ 1 ОСНОВИ МЕРЕЖЕВОГО ТРАФІКУ ……..... 9
1.1 Важливість розуміння мережевого трафіку……..... 9
1.2 Основні поняття мережевого трафіку…… 10
1.3 Види мережевого трафіку…… 12
1.4 Формування мережевого трафіку……... 13
РОЗДІЛ 2 МОНІТОРИНГ МЕРЕЖЕВОГО ТРАФІКУ ……..... 15
2.1 Роль моніторингу мережевого трафіку, його важливість та сутність … 15
2.2 Методи та засоби моніторингу мережевого трафіку……...……. 17
2.3 Цілі та завдання моніторингу…….. 20
2.4 Практичні аспекти моніторингу…… 22
2.5 Аналіз результатів моніторингу …… 24
2.6 Застосування та використання даних ...…… 30
РОЗДІЛ 3 РОЗРОБКА СИСТЕМИ МОНІТОРИНГУ МЕРЕЖЕВОГО ТРАФІКУ ...…… 33
3.1 Мета розробки застосунку моніторингу мережевого трафіку ……... 33
3.2 Огляд існуючих рішень для моніторингу ……. 35
3.3. Опис технологій та методів для розробки системи моніторингу ..…… 41
ВИСНОВКИ…… 48
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ ...…….. 49
Додаток А КОД ЗАСТОСУНКУ З МОНІТОРИНГУ……..... 53
Додаток Б ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ (Презентація)…… ……. 69

ВСТУП

У наш час, інформаційні технології стали не тільки необхідністю, але й основою для функціонування суспільства, бізнесу та науки. З кожним днем ми стаємо свідками швидкого росту та розвитку інтернету, хмарних технологій, мобільних додатків та безлічі інших інновацій, які значно полегшують наше життя, але при цьому створюють серйозні виклики в галузі управління та безпеки інформації.

Один з цих викликів - це ефективне управління мережевим трафіком. Обсяги даних, що обробляються у мережах, щодня зростають, а разом з ними зростають і потреби в ефективному моніторингу та керуванні цим трафіком. Величезний обсяг інформації, що проходить через мережі, створює потребу в надійних інструментах, які дозволять забезпечити безпеку, стабільність та ефективність роботи мережевої інфраструктури.

Системи моніторингу мережевого трафіку стають ключовим інструментом для досягнення цих цілей. Вони дозволяють отримувати інформацію про обсяги, характеристики та напрямки трафіку, а також виявляти аномалії та загрози безпеці, що дозволяє оперативно реагувати на можливі проблеми та запобігати їхньому виникненню.

Проте, існуючі підходи до моніторингу мережевого трафіку мають свої обмеження. Зростання обсягів трафіку, різноманітність пристроїв та технологій, а також поява нових форм передачі даних, наприклад, зашифрованого трафіку, ускладнюють завдання моніторингу та аналізу трафіку.

Тому метою даної дипломної роботи є розробка та реалізація ефективної системи моніторингу мережевого трафіку, яка враховуватиме сучасні тенденції та вимоги до безпеки та продуктивності. Основними завданнями дослідження будуть розробка алгоритмів збору та аналізу даних, виявлення аномалій та загроз безпеці, а також створення інтерфейсу для візуалізації результатів моніторингу[18].

РОЗДІЛ 1. ОСНОВИ МЕРЕЖЕВОГО ТРАФІКУ

1.1 Важливість розуміння мережевого трафіку

В сучасному світі комп'ютерні мережі є невід'ємною складовою багатьох аспектів нашого життя, від особистого користування до великих корпоративних систем. Основи мережевого трафіку становлять фундамент для розуміння та ефективного використання мережевої інфраструктури. Цей розділ оглядає ключові концепції та принципи мережевого трафіку та розкриває їх важливість у сучасному інформаційному середовищі.

Знання мережевого трафіку дозволяє підтримувати та оптимізувати ефективність роботи мережі, включаючи виявлення та виправлення проблем зі затримками, перевантаженням та іншими аспектами, які можуть впливати на продуктивність. Крім того, розуміння мережевого трафіку важливе для забезпечення безпеки мережі. Воно допомагає виявляти та відстежувати аномальну активність, що може свідчити про зловмисну діяльність або атаки на мережу.

Основи мережевого трафіку також необхідні для планування майбутнього розвитку мережі. Аналіз паттернів та трендів у трафіку допомагає прогнозувати зростання обсягів трафіку та вчасно вживати заходи для покращення мережевої інфраструктури. Крім того, розуміння мережевого трафіку є важливим для лагодження та діагностики проблем у мережевій інфраструктурі. Відслідковування трафіку дозволяє ідентифікувати джерела проблем та швидко вирішувати їх.

Крім того, знання основ мережевого трафіку дозволяє ефективно керувати ресурсами мережі. Це включає в себе оптимізацію використання ресурсів та підтримку високого рівня ефективності. Враховуючи всі ці аспекти, розуміння основ мережевого трафіку виявляється невід'ємною частиною будь-якої мережевої інфраструктури, що прагне забезпечити ефективну та безпечну роботу.

1.2 Основні поняття мережевого трафіку

Основні поняття мережевого трафіку становлять основу для розуміння принципів та процесів передачі даних у комп'ютерних мережах. Перед початком детального огляду понять, важливо розуміти, що мережевий трафік – це потік даних, що пересилаються між пристроями у мережі. Поняття мережевого трафіку включає в себе такі ключові аспекти, як пакети даних, протоколи та порти[12].

Пакети даних – це невеликі блоки даних, які передаються через мережу. Кожен пакет містить інформацію, таку як адреса відправника та отримувача, дані, тип протоколу тощо. Пакети даних є основною одиницею передачі інформації у мережі, і їх використання дозволяє ефективно передавати дані на великі відстані[15].

Протоколи визначають правила та стандарти для передачі даних у мережі. Найпоширеніші протоколи включають TCP (Transmission Control Protocol) та UDP (User Datagram Protocol)[5], які використовуються для надійної та ненадійної передачі даних відповідно. Інші протоколи, такі як HTTP, SMTP, SNMP[6], використовуються для специфічних цілей, таких як веб-перегляд, електронна пошта, управління мережею тощо[8].

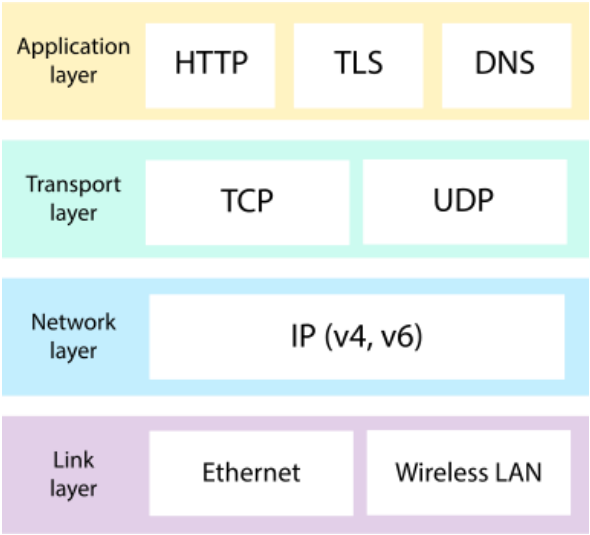


Рисунок 1.1 – Приклади протоколів мережі

Порти – це числа, які використовуються для ідентифікації конкретних служб та програм, які працюють у мережі. Кожна служба або програма має свій унікальний номер порту, що дозволяє іншим пристроям знаходити та взаємодіяти з нею. Наприклад, веб-сервери зазвичай працюють на порті 80 для HTTP-запитів та на порті 443 для HTTPS-запитів[14].

Розуміння цих основних понять мережевого трафіку є критично важливим для будь-якого IT-спеціаліста або адміністратора мережі. Воно допомагає забезпечити безпечну та ефективну роботу мережі, а також виявити та вирішити проблеми, які можуть виникнути в процесі передачі даних.

Визначення мети та призначення мережевого трафіку в комп'ютерних мережах є ключовим аспектом розуміння їхньої ролі та значення. Передусім, мережевий трафік служить для обміну даними між пристроями, які знаходяться у різних місцях мережі. Це може бути передача файлів, веб-сторінок, електронної пошти, потокового відео та багато іншого. Головна мета мережевого трафіку полягає в забезпеченні зв'язку між користувачами, серверами та іншими пристроями у мережі.

Крім того, мережевий трафік виконує ряд інших функцій, які важливі для нормальної роботи мережі. Наприклад, він дозволяє мережевим пристроям взаємодіяти між собою, передавати управління та контролювати ресурси. Також мережевий трафік може використовуватися для віддаленого керування пристроями, адміністрування мережі та моніторингу її стану.

Ще однією важливою функцією мережевого трафіку є передача даних через різні мережеві пристрої та маршрутизатори. Це дозволяє даним надійно дістатися від відправника до отримувача, незважаючи на складність мережі та її топологію[17].

Крім того, мережевий трафік може бути використаний для забезпечення безпеки мережі. Наприклад, системи виявлення вторгнень можуть аналізувати мережевий трафік для виявлення підозрілої або зловмисної активності. Також мережевий трафік може бути використаний для моніторингу та аналізу пропускної

здатності мережі, що допомагає виявляти проблеми та оптимізувати її продуктивність.

У цілому, мережевий трафік відіграє ключову роль у функціонуванні комп'ютерних мереж. Його мета та призначення полягають у забезпеченні зв'язку, передачі даних, керуванні ресурсами та забезпеченні безпеки, що робить його невід'ємною складовою будь-якої сучасної мережевої інфраструктури.

1.3 Види мережевого трафіку

Види мережевого трафіку визначаються його призначенням та характером передачі даних у мережі. Перш ніж розглядати окремі типи трафіку, важливо зазначити, що в мережі може бути одночасно декілька видів трафіку, які взаємодіють та впливають один на одного. Розуміння кожного типу допомагає ефективно керувати та оптимізувати мережу.

Потоковий трафік - це тип трафіку, що характеризується постійним потоком даних, які передаються в реальному часі. Такий трафік часто пов'язаний з мультимедійними застосунками, такими як стрімінгове відео або аудіо. Потоковий трафік вимагає стабільної пропускної здатності для забезпечення безперервності передачі даних.

Пакетний трафік відрізняється тим, що дані передаються у вигляді окремих пакетів, які пересилаються по мережі із задіянням різних маршрутів та вузлів. Кожен пакет має заголовок з метаданими, що дозволяють мережевим пристроям правильно маршрутизувати його до місця призначення. Пакетний трафік є більш гнучким, оскільки дозволяє мережі ефективно використовувати пропускну здатність.

Внутрішньомережевий трафік відбувається між пристроями, що належать до однієї і тієї ж мережі. Це можуть бути комп'ютери в офісі, пристрої в домашній мережі або сервери у дата-центрі. Внутрішньомережевий трафік зазвичай має

високу пропускну здатність та низьку затримку, оскільки він передається по прямій лінії без зовнішніх втручань.

Міжмережевий трафік відбувається між різними мережами або підмережами, які з'єднані маршрутизаторами або іншими пристроями мережі. Це може бути передача даних між компаніями через Інтернет або комунікація між різними підмережами у великих корпораціях.

Транзитний трафік - це трафік, що проходить через мережу з виключною метою пересилання до кінцевого призначення. Це може бути трафік, що прямує через певний маршрут або мережевий вузол, не зупиняючись на ньому. Транзитний трафік часто виникає при підключенні мережі до глобального Інтернету або при використанні VPN-з'єднань.

Порівнюючи ці типи трафіку, можна відзначити їхні особливості та характеристики. Потоковий трафік вимагає стабільної пропускної здатності та низької затримки, в той час як пакетний трафік є більш гнучким і ефективним у використанні ресурсів мережі. Внутрішньомережевий трафік зазвичай має високу пропускну здатність та низьку затримку, оскільки він обмежений внутрішньою мережею. Міжмережевий трафік може бути більш різноманітним та вимагати більшої уваги до забезпечення безпеки та надійності. Транзитний трафік, в свою чергу, може бути менш передбачуваним і вимагати спеціальних заходів для моніторингу та контролю[11].

1.4 Формування мережевого трафіку

Формування мережевого трафіку є одним з ключових аспектів функціонування будь-якої комп'ютерної мережі. Цей процес постійно еволюціонує та змінюється під впливом різноманітних факторів, таких як зростання користувацького попиту, розвиток нових технологій, зміна бізнес-процесів та внутрішніх політик компаній. Дії користувачів у мережі включають в себе широкий

спектр активностей, починаючи від перегляду веб-сторінок та спілкування в соціальних мережах до відправлення електронних листів, завантаження файлів та використання онлайн-сервісів.

Процеси, що відбуваються у системах та пристроях у мережі, також впливають на формування мережевого трафіку. Автоматичні оновлення програмного забезпечення, синхронізація даних з хмарними сервісами, а також віддалені резервні копії є лише деякими з процесів, які можуть спричиняти передачу великої кількості даних через мережу. Ці процеси можуть бути автоматизовані та виконуватися у фоновому режимі, але вони все одно зумовлюють певний рівень трафіку, який може мати велике значення для ефективності мережі.

Нарешті, служби та програми, що працюють у мережі, також є важливими джерелами мережевого трафіку. Вони надають різноманітні сервіси та можливості користувачам, що призводить до обміну даними між клієнтами та серверами. Веб-сервери, поштові сервери, файлообмінні системи, відео- та аудіо-стрімінгові сервіси - це лише кілька прикладів програм та сервісів, які генерують трафік у мережі. Важливою характеристикою цього трафіку є його різноманітність та неоднорідність, оскільки різні сервіси та програми можуть генерувати різні обсяги трафіку та мати різні вимоги до якості обслуговування.

Зрозуміння та аналіз формування мережевого трафіку є ключовими завданнями для адміністраторів мережі. Це дозволяє їм ефективно керувати мережевими ресурсами, виявляти та вирішувати проблеми з мережевим трафіком, а також забезпечувати високу якість обслуговування для користувачів. До цього можна використовувати різні інструменти моніторингу та аналізу мережевого трафіку, такі як програми для збору та аналізу даних про трафік, системи моніторингу мережі, а також аналітичні інструменти для виявлення та аналізу трафіку.

РОЗДІЛ 2. МОНІТОРИНГ МЕРЕЖЕВОГО ТРАФІКУ

2.1 Роль моніторингу мережевого трафіку, його важливість та сутність

Моніторинг мережевого трафіку є важливою складовою управління комп'ютерною мережею. Він передбачає систематичне спостереження, збір та аналіз даних про трафік, який протікає через мережеві пристрої, з метою забезпечення ефективності, безпеки та доступності мережевих ресурсів. Важливість моніторингу мережевого трафіку стає дедалі більшою у зв'язку зі зростанням обсягів даних, складності мережевих топологій та загроз безпеці[19].

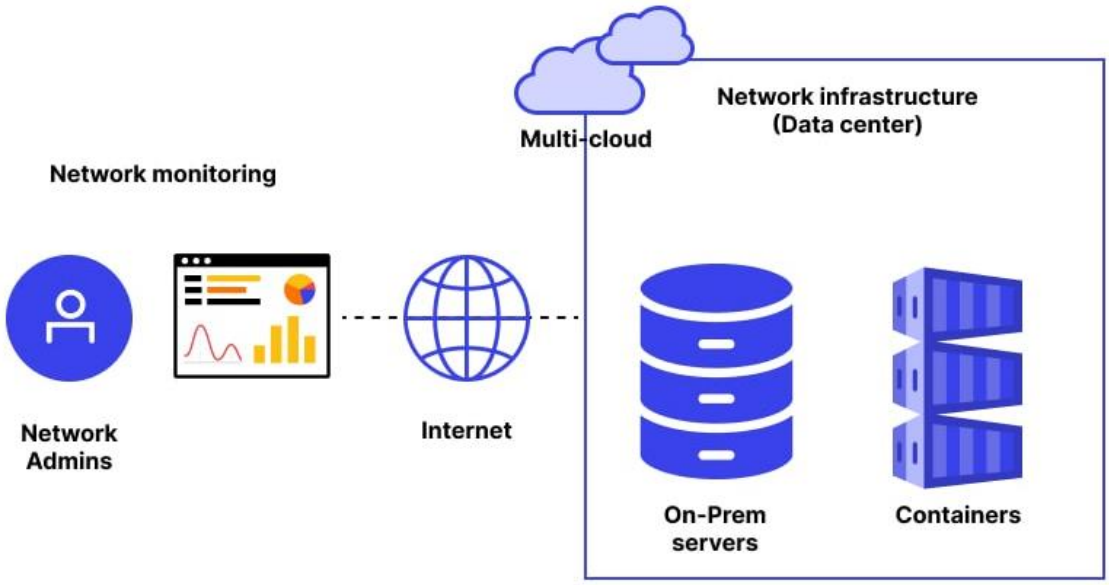


Рисунок 2.1 – Схема моніторингу мережевого трафіку

Моніторинг мережевого трафіку відіграє ключову роль у забезпеченні стабільності та продуктивності мережі. Шляхом аналізу трафіку можна виявити та вирішити проблеми зі швидкістю передачі даних, перевантаженням каналів зв'язку та інші неполадки, які можуть впливати на роботу мережі та продуктивність користувачів.

Крім того, моніторинг мережевого трафіку є необхідним для забезпечення безпеки мережі. Шляхом аналізу трафіку можна виявити аномальні активності, які можуть свідчити про атаки ззовні або внутрішні порушення безпеки. Раннє виявлення таких загроз дозволяє прийняти відповідні заходи для їх запобігання та мінімізації можливих шкідливих наслідків.

Додатково, моніторинг мережевого трафіку може допомогти в оптимізації ресурсів мережі. Аналізуючи трафік, можна виявити неефективне використання пропускної здатності та ідентифікувати можливості для оптимізації роботи мережевих пристроїв. Це дозволяє забезпечити кращу продуктивність та ефективність мережі при мінімальних витратах на ресурси[9].

У великих мережах моніторинг мережевого трафіку може бути складним завданням через велику кількість даних, які необхідно аналізувати. Тому важливо мати в своєму розпорядженні ефективні засоби та програмні рішення для збору, візуалізації та аналізу даних про трафік. Тільки такий підхід дозволить забезпечити ефективний та надійний моніторинг мережі[4].

Визначення мети моніторингу мережевого трафіку - це важливий етап у процесі управління комп'ютерною мережею, який передбачає чітке визначення цілей та завдань, що мають бути досягнуті за допомогою моніторингу трафіку. Головна мета моніторингу мережевого трафіку полягає в забезпеченні ефективності, стабільності та безпеки мережі, а також в постійному контролі та аналізі її роботи.

Перш за все, мета моніторингу мережевого трафіку полягає в забезпеченні ефективності мережі. Шляхом аналізу обсягів та характеристик трафіку можна виявити можливі проблеми зі швидкістю передачі даних, перевантаженням каналів зв'язку та інші аспекти, які можуть негативно впливати на роботу мережі та продуктивність користувачів. Такий аналіз дозволяє приймати вчасні заходи для вирішення проблем та оптимізації роботи мережі.

Друга важлива мета моніторингу мережевого трафіку - забезпечення безпеки мережі. Шляхом аналізу трафіку можна виявити аномальні активності, які можуть свідчити про атаки ззовні або внутрішні порушення безпеки. Раннє виявлення

таких загроз дозволяє прийняти відповідні заходи для їх запобігання та мінімізації можливих шкідливих наслідків.

Крім того, моніторинг мережевого трафіку допомагає в оптимізації ресурсів мережі. Аналізуючи трафік, можна виявити неефективне використання пропускної здатності та ідентифікувати можливості для оптимізації роботи мережевих пристроїв. Це дозволяє забезпечити кращу продуктивність та ефективність мережі при мінімальних витратах на ресурси[20].

Нарешті, мета моніторингу мережевого трафіку полягає в забезпеченні доступності мережевих ресурсів для користувачів. Шляхом аналізу трафіку можна виявити проблеми з доступністю ресурсів та шляхи їх вирішення, що дозволяє забезпечити безперебійну роботу мережі та задоволення потреб користувачів[1][2][3].

2.2 Методи та засоби моніторингу мережевого трафіку

Огляд різних методів та інструментів моніторингу мережі є важливим аспектом забезпечення ефективного та надійного функціонування комп'ютерних мереж. Оскільки мережевий трафік є основною складовою мережевого середовища, важливо мати засоби для його аналізу, візуалізації та моніторингу.

Один з найпоширеніших методів моніторингу мережевого трафіку - використання сніферів або мережевих аналізаторів. Ці інструменти дозволяють перехоплювати, аналізувати та відображати мережевий трафік у реальному часі. Вони надають детальну інформацію про передачу даних, включаючи джерело та призначення пакетів, протоколи, типи сервісів та багато іншого.

Протокольні аналізатори є ще одним важливим інструментом моніторингу мережі, які спеціалізуються на аналізі трафіку на рівні протоколів. Вони дозволяють аналізувати заголовки пакетів та виявляти потенційні проблеми на рівні протоколів, такі як помилки в адресації, фрагментація пакетів, конфлікти IP-

адрес тощо. Ці інструменти дозволяють операторам мережі швидко виявляти та усувати проблеми, що виникають на рівні протоколів[16].

Системи моніторингу мережі є комплексними рішеннями, які забезпечують надійний та постійний моніторинг роботи мережі. Вони надають централізоване керування та аналіз трафіку, а також забезпечують можливості відстеження та реагування на події в мережі. Деякі системи моніторингу мережі також надають можливості автоматизованого аналізу та прогнозування роботи мережі на основі зібраних даних.

Крім того, існують різноманітні інструменти для візуалізації мережевого трафіку, такі як графіки, діаграми та звіти. Ці інструменти дозволяють операторам мережі швидко виявляти та аналізувати зміни в трафіку, виявляти проблемні ділянки мережі та приймати відповідні заходи для їх вирішення. Такі інструменти забезпечують зручний та ефективний спосіб візуалізації та аналізу мережевого трафіку для операторів мережі.

Сніфери та мережеві аналізатори мають свої переваги та недоліки. Перевагою цих інструментів є їхня здатність перехоплювати та аналізувати мережевий трафік у реальному часі, що дозволяє операторам мережі швидко виявляти та реагувати на проблеми. Вони також надають детальну інформацію про передачу даних, що дозволяє проводити детальний аналіз трафіку на різних рівнях.

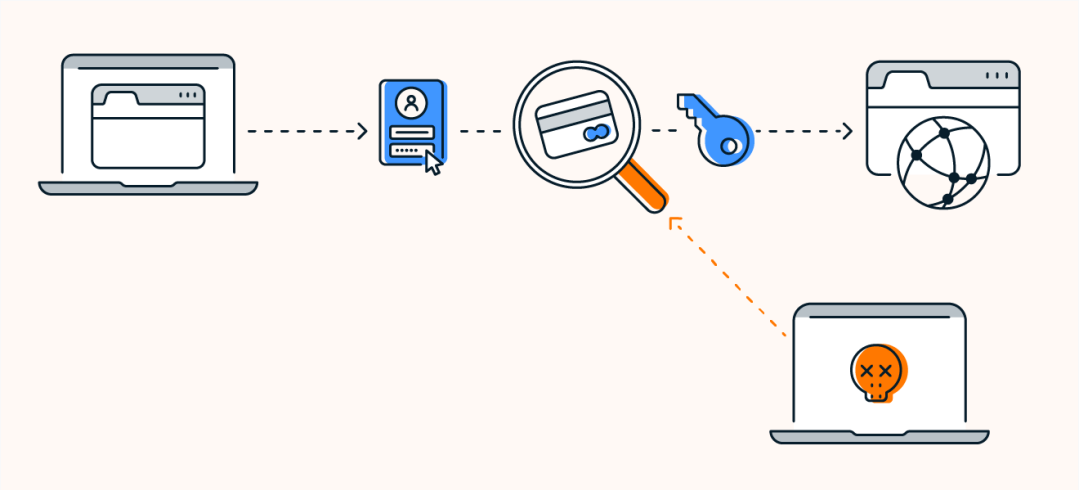


Рисунок 2.2 – Схема робота сніферів

Однак у сніферів та мережевих аналізаторів є і свої недоліки. Перш за все, вони можуть бути складні у використанні та вимагати спеціальних навичок для ефективного використання. Також вони можуть бути обмежені в можливостях аналізу трафіку на великих мережах через обмежену пропускну здатність та ресурси[10].

Протокольні аналізатори також мають свої переваги та недоліки. Основною перевагою є їхня спроможність аналізувати трафік на рівні протоколів та виявляти проблеми на цьому рівні. Вони надають детальну інформацію про роботу мережі та можуть бути ефективними в виявленні та усуненні проблем на рівні протоколів.

Однак протокольні аналізатори також мають свої обмеження. Вони можуть бути менш ефективними у виявленні проблем на вищих рівнях мережі, таких як проблеми з пропускною здатністю або навантаженням каналів зв'язку. Також вони можуть бути складні у використанні та вимагати спеціальних знань про протоколи мережі.

Системи моніторингу мережі мають свої переваги та недоліки. Однією з головних переваг є їхня здатність забезпечувати централізоване керування та аналіз трафіку, що дозволяє операторам мережі отримувати повний огляд стану мережі та приймати вчасні рішення. Також вони можуть надавати автоматизовані засоби аналізу та прогнозування роботи мережі.

Однак системи моніторингу мережі можуть мати свої обмеження. Вони можуть бути витратними та складними у впровадженні, особливо для малих та середніх підприємств. Також вони можуть мати обмежену пропускну здатність та обмежені можливості аналізу великих обсягів даних про трафік.

2.3 Цілі та завдання моніторингу

Основні цілі моніторингу мережевого трафіку включають в себе різноманітні завдання та аспекти, спрямовані на забезпечення ефективності, безпеки та стабільності комп'ютерної мережі.

Одна з головних цілей моніторингу мережевого трафіку - виявлення аномального трафіку. Це означає виявлення незвичайних, непередбачуваних або некоректних паттернів передачі даних, які можуть свідчити про атаки ззовні, внутрішні порушення безпеки, віруси або інші проблеми. Здатність вчасно виявляти та реагувати на аномальний трафік дозволяє забезпечити безпеку та надійність мережі.

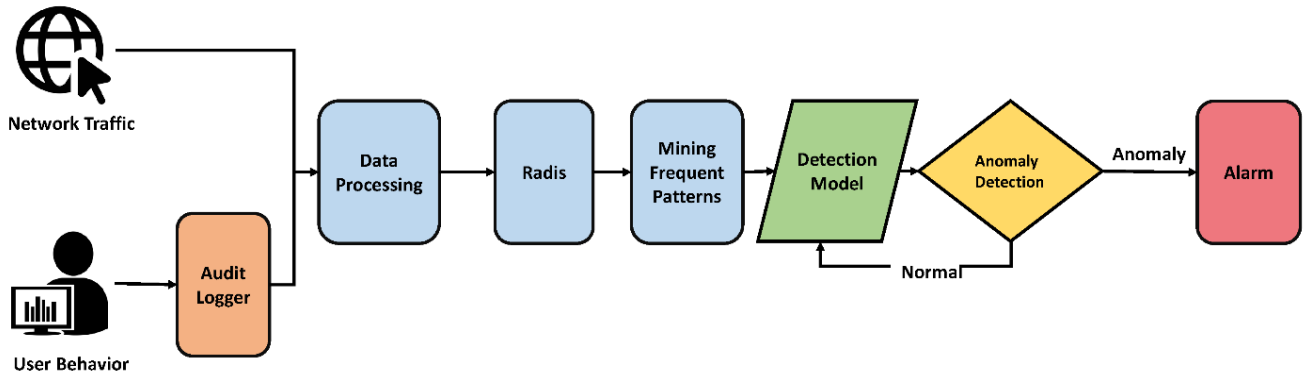


Рисунок 2.2 – Схема аналізу мережевого трафіку

Ще одна важлива ціль моніторингу мережевого трафіку - виявлення проблем у мережевій інфраструктурі. Це може включати в себе виявлення проблем з пропускною здатністю каналів зв'язку, перевантаження мережевих пристроїв, несправності в роботі мережевого обладнання тощо. Шляхом моніторингу трафіку можна виявити ці проблеми та вжити заходів для їх вирішення, що допомагає забезпечити стабільну та ефективну роботу мережі.

Також моніторинг мережевого трафіку допомагає в оптимізації ресурсів мережі. Шляхом аналізу трафіку можна виявити неефективне використання пропускної здатності, проблеми з використанням мережевих пристроїв, зайві

навантаження тощо. Це дозволяє приймати раціональні рішення щодо розподілу та оптимізації ресурсів мережі, що сприяє підвищенню її ефективності та продуктивності.

Крім цього, однією з цілей моніторингу мережевого трафіку є забезпечення високої доступності мережевих ресурсів для користувачів. Шляхом аналізу трафіку можна виявити проблеми з доступністю ресурсів та приймати заходи для їх вирішення, що допомагає забезпечити безперебійну роботу мережі та задоволення потреб користувачів.

Моніторинг мережі вирішує різноманітні завдання, спрямовані на забезпечення ефективності, безпеки та надійності комп'ютерної мережі. Одним з типових завдань, які вирішуються за допомогою моніторингу мережі, є виявлення та аналіз аномального трафіку. Це включає в себе виявлення незвичайних паттернів передачі даних, які можуть свідчити про атаки ззовні, внутрішні порушення безпеки, віруси або інші проблеми[23].

Ще одним важливим завданням є виявлення проблем у мережевій інфраструктурі. Моніторинг мережі допомагає виявити проблеми з пропускною здатністю каналів зв'язку, перевантаження мережевих пристроїв, несправності в роботі мережевого обладнання тощо. Шляхом аналізу трафіку можна виявити ці проблеми та вжити заходів для їх вирішення, що допомагає забезпечити стабільну та ефективну роботу мережі.

Крім того, моніторинг мережі допомагає в оптимізації ресурсів мережі. Шляхом аналізу трафіку можна виявити неефективне використання пропускної здатності, проблеми з використанням мережевих пристроїв, зайві навантаження тощо. Це дозволяє приймати раціональні рішення щодо розподілу та оптимізації ресурсів мережі, що сприяє підвищенню її ефективності та продуктивності.

Однією зі складних задач є забезпечення високої доступності мережевих ресурсів для користувачів. Шляхом аналізу трафіку можна виявити проблеми з доступністю ресурсів та приймати заходи для їх вирішення, що допомагає забезпечити безперебійну роботу мережі та задоволення потреб користувачів.

Також до типових завдань моніторингу мережі входить виявлення та аналіз споживання ресурсів мережі, моніторинг використання пропускної здатності, виявлення та реагування на інциденти безпеки, моніторинг роботи мережевих пристроїв та додатків, аналіз якості обслуговування та багато іншого[30].

2.4 Практичні аспекти моніторингу

Планування та реалізація системи моніторингу мережевого трафіку є важливим етапом у забезпеченні ефективності та безпеки комп'ютерної мережі. Цей процес включає в себе кілька ключових аспектів, починаючи від визначення вимог та цілей моніторингу до вибору відповідного програмного та апаратного забезпечення.

Перший крок у плануванні системи моніторингу - це визначення вимог і цілей. Це включає в себе виявлення потреб організації щодо моніторингу трафіку, таких як виявлення проблем безпеки, оптимізація ресурсів мережі, забезпечення високої доступності тощо. На цьому етапі також важливо враховувати специфічні вимоги та особливості самої мережі.

Після визначення вимог наступним кроком є вибір відповідного програмного та апаратного забезпечення. Це може включати в себе вибір програм для аналізу трафіку, сніферів, протокольних аналізаторів, систем моніторингу мережі тощо. Важливо враховувати потреби організації, її бюджет та технічні можливості при виборі відповідного обладнання та програмного забезпечення.

Після вибору необхідного обладнання та програмного забезпечення необхідно перейти до реалізації системи моніторингу. Це включає в себе налаштування обладнання, налаштування програмного забезпечення, налаштування механізмів збору та аналізу даних, а також інтеграцію з існуючою мережевою інфраструктурою.

Останнім етапом є тестування та налагодження системи моніторингу. Це важливий крок, щоб переконатися, що система працює ефективно та відповідає вимогам та цілям організації. Тестування допомагає виявити та усунути проблеми, які можуть виникнути під час роботи системи моніторингу.

В цілому, планування та реалізація системи моніторингу мережевого трафіку - це складний процес, який вимагає пильного аналізу вимог, вибору програмного забезпечення та відповідного обладнання, а також налагодження та тестування для забезпечення ефективної роботи системи.

Практичні аспекти моніторингу мережевого трафіку включають в себе ряд порад та кращих практик, спрямованих на ефективне використання засобів моніторингу та забезпечення надійності та безпеки мережі.

Однією з ключових порад є визначення чітких метрик та критеріїв успіху для моніторингу. Це допомагає зорієнтуватися у процесі моніторингу та визначити, коли мережа працює належним чином, а коли виникають проблеми.

Також важливо систематично аналізувати зібрані дані та вживати вчасних заходів у випадку виявлення аномалій чи проблем. Регулярне моніторинг та аналіз трафіку дозволяють оперативно реагувати на потенційні проблеми та уникати їх подальшого розвитку.

Для ефективного моніторингу рекомендується використовувати різноманітні інструменти та методики, включаючи сніфери, протокольні аналізатори, системи моніторингу мережі тощо. Кожен з цих інструментів має свої переваги та обмеження, тому їх комбінація дозволяє отримати більш повне уявлення про стан мережі.

Для забезпечення ефективного моніторингу важливо також правильно налаштувати систему збору та аналізу даних. Це включає в себе налаштування фільтрів, вибір необхідних параметрів для моніторингу, регулярне оновлення програмного забезпечення тощо.

Не менш важливою є регулярна перевірка та поновлення системи моніторингу з метою врахування змін у мережевій інфраструктурі, відповідно до

зростаючих потреб та вимог організації. Технології та стандарти швидко змінюються, тому постійне оновлення системи моніторингу є важливим аспектом.

Крім того, рекомендується забезпечити надійне зберігання та захист зібраних даних моніторингу, оскільки ця інформація містить конфіденційну та чутливу інформацію про мережу та її користувачів. Застосування криптографічних методів шифрування та забезпечення обмеженого доступу до даних є ключовими аспектами у забезпеченні безпеки моніторингу мережі[24].

Узагальнюючи, ефективний моніторинг мережі вимагає комплексного підходу, що включає в себе визначення метрик та цілей, використання різноманітних інструментів та методів, налаштування системи моніторингу та постійне оновлення та адаптацію до змін у мережевій інфраструктурі.

2.5 Аналіз результатів моніторингу

Аналіз результатів моніторингу мережевого трафіку відіграє ключову роль у забезпеченні ефективності, безпеки та надійності комп'ютерних мереж. Однією з основних функцій аналізу є використання зібраних даних для отримання інформації про роботу мережі, її навантаження, ефективність та проблеми, що можуть виникнути.

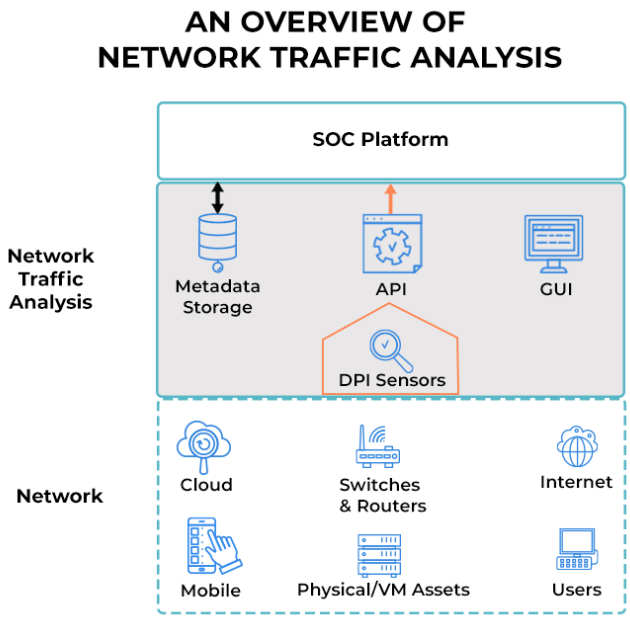


Рисунок 2.3 – Приклад побудови застосунку для аналізу мережевого трафіку

Першим кроком у процесі аналізу є обробка та інтерпретація зібраних даних. Це включає в себе перегляд та фільтрацію інформації, щоб виділити ключові параметри та показники, які відображають стан мережі. Наприклад, це може бути обсяг переданих даних, частота виникнення певних типів пакетів, пропускна здатність каналів зв'язку тощо.

Після обробки даних проводиться аналіз для виявлення ключових трендів та паттернів у трафіку. Це може включати в себе виявлення пікових навантажень, розподіл типів трафіку за часом, ідентифікацію найбільш активних користувачів чи додатків, а також виявлення аномальних паттернів, які можуть свідчити про проблеми або атаки у мережі.

Важливою частиною аналізу результатів є інтерпретація отриманих даних та прийняття відповідних рішень. Це може включати в себе вжиття заходів для оптимізації ресурсів мережі, виявлення та усунення проблем, вжиття заходів для підвищення безпеки мережі тощо.

Застосування аналізу результатів моніторингу мережевого трафіку дозволяє забезпечити ефективну роботу мережі, запобігти виникненню проблем та атак, а також підвищити загальний рівень безпеки та продуктивності у мережевому

середовищі. У наступних абзацах я розгляну методи, такі як flow-based аналіз, аналіз пакетів, статистичний аналіз та машинне навчання, та обговорю їх переваги, недоліки та використання у сучасних системах безпеки мережі.

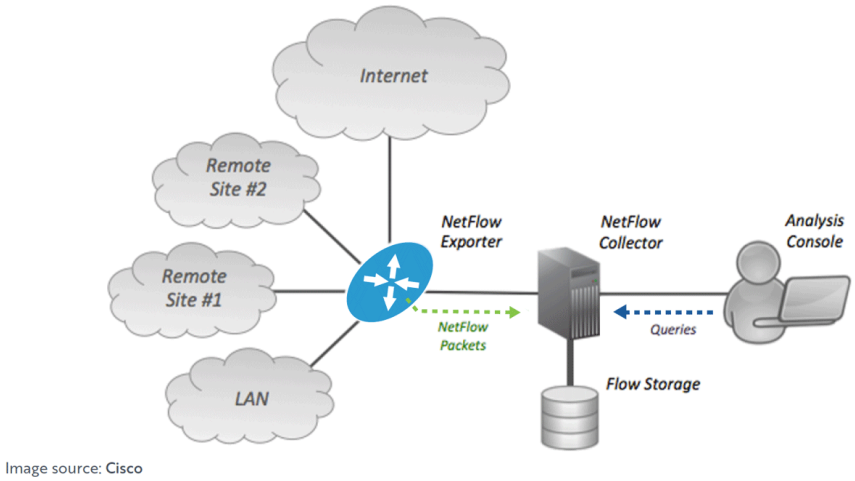


Рисунок 2.4 – Схема flow-based аналізу та аналізу пакетів

Аналіз пакетів – це метод виявлення, інтерпретації та аналізу окремих мережевих пакетів для отримання детальної інформації про їх структуру, вміст та характеристики. Під час аналізу пакетів дослідник аналізує заголовки та дані кожного пакету з метою розуміння його призначення, джерела та призначення, а також виявлення будь-яких ненормальних чи підозрілих ознак.

Переваги аналізу пакетів включають можливість отримати детальну інформацію про кожен окремий пакет у мережі. Цей метод дозволяє виявляти незвичайні, підозрілі або відхилені пакети, які можуть свідчити про атаки або аномалії у мережі. Детальний аналіз пакетів дозволяє також розпізнати різні протоколи, типи сервісів та типи атак, що відбуваються у мережі.

Недоліки аналізу пакетів включають обмежену пропускну здатність, оскільки аналіз кожного пакету може призвести до значного збільшення обсягу даних, особливо в великих мережах. Це може призвести до значного затримування в обробці трафіку та перевищення ресурсів обчислення. Крім того, аналіз пакетів може бути складним у випадку зашифрованого трафіку або використання захисту від вторгнень, що може ускладнити виявлення аномалій чи атак у мережі.

Незважаючи на ці обмеження, аналіз пакетів залишається невід'ємною частиною процесу моніторингу та аналізу мережевого трафіку. Використання цього методу разом із іншими методами, такими як flow-based аналіз чи машинне навчання, дозволяє забезпечити повний та комплексний підхід до виявлення аномалій та захисту мережі[29].

Flow-based аналіз є одним з ключових методів аналізу мережевого трафіку, який базується на групуванні мережевих пакетів у потоки даних за певними характеристиками. У контексті flow-based аналізу, потік даних - це послідовність мережевих пакетів, які мають спільні атрибути, такі як IP-адреса джерела та призначення, порти, протоколи та інші. Цей метод дозволяє агрегувати та узагальнювати інформацію про мережевий трафік, спрощуючи процес аналізу та виявлення аномалій.

Переваги flow-based аналізу включають швидкість та ефективність. Замість аналізу кожного окремого пакету, як це відбувається у методі аналізу пакетів, flow-based аналіз групує пакети в потоки даних, що дозволяє оперативно аналізувати великі обсяги мережевого трафіку. Цей метод також дозволяє виявляти загальні патерни та тенденції у мережі, такі як найбільш активні комунікаційні канали, типи протоколів, які використовуються, та інші аспекти.

Проте, flow-based аналіз має і свої обмеження. Один з найбільших недоліків полягає в втраті деталізації. При групуванні пакетів у потоки даних, деяка детальна інформація може бути втрачена, що може ускладнити виявлення конкретних проблем або аномалій. Крім того, у випадку великих мереж або великих обсягів трафіку, може виникнути проблема масштабування, коли обробка та аналіз великої кількості потоків даних стає витратною за ресурсами.

Незважаючи на ці недоліки, flow-based аналіз залишається ефективним та важливим методом аналізу мережевого трафіку, особливо для оперативного виявлення загальних патернів та аномалій у мережі. Крім того, він часто використовується у поєднанні з іншими методами аналізу, такими як аналіз пакетів чи машинне навчання, для створення комплексного підходу до моніторингу та захисту мережі[35].

Статистичний аналіз – це метод аналізу мережевого трафіку, який використовує статистичні методи та метрики для оцінки різних аспектів трафіку. У цьому методі дані про мережевий трафік агрегуються та аналізуються з точки зору їх розподілу, залежностей та інших статистичних характеристик.

Переваги статистичного аналізу включають можливість отримання комплексної інформації про характеристики мережевого трафіку, такі як пропускна здатність, швидкість передачі даних, розмір пакетів та інші параметри. Цей метод дозволяє виявити тенденції, аномалії та проблеми у мережі, використовуючи аналіз статистичних даних. Крім того, статистичний аналіз може бути ефективним інструментом для моніторингу та управління мережею, дозволяючи оперативно реагувати на зміни та проблеми.

Проте, статистичний аналіз має і свої недоліки. Один з головних недоліків полягає у важкості виявлення складних аномалій чи відхилень у мережевому трафіку. Деякі аномалії можуть бути прихованими або невидимими в статистичних даних, що ускладнює їх виявлення. Крім того, статистичний аналіз може бути відчутним за ресурсами обчислювального обладнання, особливо при аналізі великих обсягів даних. Також важливо враховувати, що статистичний аналіз може вимагати великої кількості даних для надійних висновків та прогнозів.

Незважаючи на ці недоліки, статистичний аналіз залишається важливим інструментом для аналізу мережевого трафіку, особливо у поєднанні з іншими методами аналізу, такими як аналіз пакетів, flow-based аналіз та машинне навчання. Комбінація різних методів дозволяє отримати більш повне та комплексне уявлення про стан та поведінку мережі[31].

Машинне навчання - це галузь штучного інтелекту, яка досліджує розробку алгоритмів, що надають комп'ютерам здатність вчитися на основі даних і робити прогнози та приймати рішення без явного програмування. У контексті аналізу мережевого трафіку, методи машинного навчання використовуються для виявлення аномалій, ідентифікації загроз, класифікації пакетів та багато інших завдань.

Переваги методів машинного навчання в аналізі мережевого трафіку включають можливість виявлення складних та невидимих патернів та аномалій, які можуть бути важко або неможливо виявити іншими методами. Машинне навчання може автоматично адаптуватися до змін в мережевому середовищі та навчатися на нових даних, що дозволяє підтримувати високий рівень ефективності та точності. Крім того, методи машинного навчання можуть працювати з великими обсягами даних та виявляти залежності та взаємозв'язки, які можуть бути складними для сприйняття людиною.

Однак методи машинного навчання мають і свої недоліки. Один з найбільших недоліків полягає у потребі у великій кількості даних для навчання та валідації моделей. Недостатність даних може призвести до перенавчання або недонавчання моделі, що може призвести до поганих результатів. Крім того, методи машинного навчання можуть бути вразливими до атак, які можуть маніпулювати вхідними даними або моделями для отримання неправильних результатів.

Незважаючи на ці недоліки, методи машинного навчання залишаються важливим інструментом для аналізу мережевого трафіку, особливо у поєднанні з іншими методами аналізу, такими як аналіз пакетів, flow-based аналіз та статистичний аналіз. Комбінація різних методів дозволяє отримати більш повне та комплексне уявлення про стан та поведінку мережі[28].

Графічний аналіз мережевого трафіку використовує візуальні засоби, такі як графіки, діаграми, топології та інші графічні представлення, для аналізу та візуалізації характеристик трафіку у мережі. Цей метод дозволяє операторам та аналітикам миттєво розглядати та розуміти дані трафіку, сприяючи швидкому виявленню аномалій, ідентифікації патернів та прийняттю рішень.

Переваги графічного аналізу мережевого трафіку полягають у його інтуїтивності та легкості сприйняття. Візуальні засоби дозволяють операторам швидко відображати складні дані трафіку у зрозумілі та інформативні графіки та діаграми. Це полегшує виявлення взаємозв'язків, тенденцій та аномалій у мережі, що може значно збільшити швидкість реакції на потенційні загрози та проблеми.

Однак графічний аналіз мережевого трафіку має деякі недоліки. Один з найбільших недоліків полягає в обмеженні в обробці великих обсягів даних. Хоча графічні представлення можуть бути ефективними для візуалізації невеликих обсягів даних, вони можуть стати непрактичними для аналізу великих мереж з великою кількістю вузлів та з'єднань. Крім того, графічний аналіз може бути менш ефективним у виявленні складних аномалій або взаємозв'язків, які можуть бути неочевидними з візуального погляду.

Незважаючи на ці недоліки, графічний аналіз мережевого трафіку залишається важливим інструментом для аналізу мережі, особливо як частина комплексного підходу до моніторингу та захисту мережі. Комбінація графічного аналізу з іншими методами аналізу, такими як аналіз пакетів, статистичний аналіз та машинне навчання, може забезпечити більш повне та глибоке розуміння мережевого трафіку та поведінки мережі в цілому[33].

2.6 Застосування та використання даних

Використання результатів моніторингу для оптимізації мережевої інфраструктури є ключовим етапом у забезпеченні ефективності та надійності роботи комп'ютерних мереж. Оптимізація мережевої інфраструктури передбачає аналіз зібраних даних моніторингу з метою виявлення потенційних проблем, визначення ефективного використання ресурсів та розробки стратегій для підвищення продуктивності та безпеки мережі.

Однією з ключових областей застосування даних моніторингу є виявлення та усунення проблем у мережевій інфраструктурі. Аналіз результатів моніторингу дозволяє виявити перевантаження мережевих вузлів, проблеми з підключенням, перешкоди у передачі даних та інші аномалії, що можуть впливати на роботу мережі. На основі цих даних можуть бути прийняті заходи для усунення виявлених проблем та підвищення ефективності мережевої інфраструктури.

Крім того, результати моніторингу можуть бути використані для виявлення неефективного використання ресурсів мережі та розробки стратегій для їх оптимізації. Наприклад, за допомогою аналізу даних можна виявити пристрої чи додатки, що споживають надмірні ресурси мережі, і розробити методи для їх оптимізації або розподілу навантаження. Це допомагає підтримувати високу продуктивність мережі та запобігати перевантаженням, які можуть призвести до відмови системи.

Застосування результатів моніторингу також може включати в себе розробку стратегій для підвищення безпеки мережі. Аналіз даних моніторингу дозволяє виявити потенційні загрози безпеці, незвичайну активність чи аномалії в трафіку, що може свідчити про атаки або інциденти безпеки. На основі цих даних можуть бути прийняті заходи для запобігання атакам та підвищення рівня захисту мережі.

Узагальнюючи, використання результатів моніторингу для оптимізації мережевої інфраструктури є важливим етапом у забезпеченні ефективності, надійності та безпеки комп'ютерних мереж. Аналіз даних моніторингу дозволяє виявляти проблеми, оптимізувати ресурси та розробляти стратегії для підвищення продуктивності та безпеки мережі.

Планування майбутніх заходів щодо покращення ефективності та безпеки мережі на основі аналізу моніторингу є важливою складовою стратегії управління мережевою інфраструктурою. Організації в усіх галузях використовують дані моніторингу для розробки та впровадження конкретних заходів з метою оптимізації та покращення роботи своїх мереж.

На основі результатів аналізу моніторингу, організації можуть виявити слабкі місця у своїй мережевій інфраструктурі, наприклад, збільшення навантаження на певні вузли, проблеми з безпекою, неефективне використання ресурсів тощо. Планування майбутніх заходів полягає в розробці стратегій для вирішення цих проблем та покращення роботи мережі.

Однією з ключових аспектів планування майбутніх заходів є визначення пріоритетів. Організації повинні визначити найбільш критичні аспекти їхньої мережі, які потребують негайного втручання. Це може бути підвищення безпеки,

зменшення затримок у мережі, підвищення пропускної здатності або покращення якості обслуговування для користувачів.

Далі важливо розробити конкретний план дій для впровадження запланованих заходів. Це може включати в себе визначення потрібних ресурсів, визначення відповідальних осіб, розробку часового графіка та розподіл бюджету. План має бути чітким та конкретним, з урахуванням всіх можливих ризиків та викликів, що можуть виникнути під час впровадження.

Окрім цього, організації повинні враховувати масштабність та припустимість запропонованих заходів. Переконуючись, що запропоновані зміни можуть бути успішно впроваджені і не викликають негативних наслідків для існуючої мережі.

Не менш важливою є стеження за результатами після впровадження запланованих заходів. Організації повинні постійно відслідковувати ефективність та результативність запропонованих змін, а також реагувати на будь-які негативні наслідки або проблеми, які можуть виникнути.

Узагальнюючи, планування майбутніх заходів щодо покращення ефективності та безпеки мережі на основі аналізу моніторингу є важливою складовою стратегії управління мережевою інфраструктурою. Цей процес включає в себе визначення пріоритетів, розробку конкретного плану дій, урахування масштабності та припустимості змін, а також стеження за результатами після впровадження.

РОЗДІЛ 3. РОЗРОБКА СИСТЕМИ МОНІТОРИНГУ МЕРЕЖЕВОГО ТРАФІКУ

3.1 Мета розробки застосунку моніторингу мережевого трафіку

В сучасному світі комп'ютерні мережі відіграють ключову роль у функціонуванні практично кожного аспекту нашого життя. Вони стали не лише основою для бізнесу та індустрії, але й невід'ємною частиною нашого повсякденного способу життя. Із зростанням залежності від цифрових технологій, важливість мережевої інфраструктури та її ефективного функціонування надзвичайно велика.

Однак разом зі зростанням складності мереж та об'єму даних, що передаються, зростають і виклики, пов'язані з їх управлінням та підтримкою. Один із таких викликів - це необхідність постійного моніторингу мережевого трафіку. Моніторинг мережевого трафіку є важливим елементом у забезпеченні ефективності та безпеки мережі. Він дозволяє адміністраторам мережі відстежувати стан мережі в реальному часі, виявляти аномалії та проблеми, а також реагувати на них своєчасно.

Розробка системи моніторингу мережевого трафіку є ключовим завданням для будь-якої організації, яка має велику мережеву інфраструктуру. Ця система дозволяє збирати, аналізувати та візуалізувати дані про мережевий трафік, що допомагає адміністраторам мережі отримувати необхідну інформацію для ефективного управління мережею.

У даному розділі буде розглянуто процес розробки системи моніторингу мережевого трафіку, від визначення вимог та обрання потрібних інструментів до впровадження та подальшої експлуатації. Розуміння цього процесу дозволить організаціям забезпечити надійність, ефективність та безпеку своєї мережі, що стає важливим етапом у забезпеченні успішної діяльності в цифровій епохі.

Мета розробки застосунку моніторингу мережевого трафіку полягає в створенні інструменту, який дозволить організаціям ефективно відстежувати, аналізувати та контролювати мережевий трафік у реальному часі. Основним об'єктивом цього процесу є забезпечення надійності, безпеки та ефективності роботи мережі. Розробка такого застосунку має на меті допомогти адміністраторам мережі оперативно виявляти проблеми, виконувати аналіз та приймати вчасні рішення для виправлення неполадок і оптимізації роботи мережі.

При розробці застосунку моніторингу мережевого трафіку важливою метою є забезпечення зручного та інтуїтивно зрозумілого інтерфейсу, який дозволить користувачам легко отримувати доступ до необхідної інформації про мережевий трафік. Крім того, застосунок повинен мати можливості для налаштування та адаптації під конкретні потреби організації, щоб забезпечити максимальну ефективність його використання.

Іншим важливим аспектом є забезпечення високої швидкодії та продуктивності застосунку, особливо в умовах великого обсягу даних та інтенсивного мережевого трафіку. Це означає розробку ефективних алгоритмів збору, обробки та візуалізації даних, а також оптимізацію ресурсів системи для забезпечення швидкого доступу до інформації та реагування на події в реальному часі.

Крім того, однією з ціллю розробки є забезпечення високого рівня безпеки та конфіденційності оброблюваних даних. Застосунок моніторингу мережевого трафіку повинен мати вбудовані засоби захисту від несанкціонованого доступу та зловживання, а також механізми шифрування для збереження конфіденційності інформації користувачів.

Узагальнюючи, мета та об'єктив розробки застосунку моніторингу мережевого трафіку полягають у створенні потужного, надійного та ефективного інструменту, який допоможе організаціям керувати та оптимізувати роботу своїх мереж з високою швидкодією, безпекою та зручністю використання.

3.2 Огляд існуючих рішень для моніторингу

На сьогоднішній день існує велика кількість рішень для моніторингу мережевого трафіку, які варіюються за функціональністю, складністю та ціною. Основні з них включають у себе програмні та апаратні засоби, які надають широкий спектр можливостей для збору, аналізу та візуалізації даних про мережевий трафік.

Одним з найпоширеніших рішень є програмні засоби моніторингу, такі як Wireshark, SolarWinds, PRTG Network Monitor та інші. Ці програми надають інтерфейси для збору та аналізу даних про мережевий трафік, а також різноманітні інструменти для візуалізації результатів. Вони можуть бути встановлені на комп'ютери або сервери в мережі та надають зручні інтерфейси для моніторингу та аналізу трафіку[7].

Крім програмних рішень, існують і апаратні засоби моніторингу мережі. Це, наприклад, мережеві аналізатори трафіку, які можуть бути встановлені в мережі на різних рівнях, від точок входу до розподільчих вузлів. Вони дозволяють адміністраторам отримувати доступ до даних про мережевий трафік на високому рівні та аналізувати його з точністю до пакетів даних.

Також варто зазначити системи моніторингу мережі, які можуть бути інтегровані з іншими інфраструктурними рішеннями, такими як системи управління мережею (NMS). Ці системи надають комплексний підхід до моніторингу мережі, який включає в себе як аналіз трафіку, так і інші аспекти мережевої інфраструктури, такі як стан пристроїв, пропускна здатність та інші.

Нарешті, існують хмарні рішення моніторингу мережі, які надають можливість віддаленого моніторингу трафіку без потреби встановлення додаткового обладнання чи програмного забезпечення. Ці рішення дозволяють адміністраторам моніторити мережу з будь-якої точки з доступом до Інтернету, що зручно для дистанційного адміністрування та моніторингу мережі.

Варто розглянути детальніше існуючі рішення щодо моніторингу мережевого трафіку та дослідити їх переваги та недоліки. Виходячи з цього можна

буде конкретно сформулювати вимоги до застосунку та впроваджувати більш ефективні методи та технології. Далі будуть розглянуті існуючі рішення по моніторингу мережі[13].

Рисунок 3.1 – Логотип Wireshark

Wireshark є однією з найпопулярніших програм для аналізу мережевого трафіку, яка використовується адміністраторами мереж та інженерами для діагностики проблем, виявлення вразливостей і відстеження даних, які передаються через мережу. Вона є відкритою та безкоштовною програмою, що робить її дуже популярною серед спеціалістів з інформаційної безпеки та мережевого адміністрування.

Wireshark забезпечує широкий спектр можливостей для аналізу мережевого трафіку. Вона може захоплювати дані, які пересилаються через мережу, на різних рівнях OSI моделі, від фізичного до додаткового. Це дозволяє аналізувати різноманітні аспекти трафіку, такі як заголовки протоколів, вміст пакетів та способи передачі даних.

Після захоплення даних Wireshark надає користувачам можливість візуалізувати та аналізувати їх у зручному інтерфейсі. Користувачі можуть переглядати пакети даних у реальному часі або аналізувати захоплені дані пізніше. Інтерфейс програми включає різноманітні фільтри та інструменти для пошуку та фільтрації даних, що дозволяє знаходити і виокремлювати важливу інформацію.

Однією з основних переваг Wireshark є підтримка широкого спектру мережевих протоколів. Вона може аналізувати протоколи TCP/IP, UDP, HTTP, DNS, SSL, SSH та багато інших. Це робить Wireshark універсальним інструментом для аналізу мережевого трафіку в різних середовищах та застосуваннях.

Окрім цього, Wireshark має активну спільноту користувачів та розробників, яка постійно оновлює та покращує програму. Це означає, що нові функції та виправлення помилок регулярно надходять у релізах програмного забезпечення. Таким чином, Wireshark залишається одним з найбільш потужних та надійних інструментів для моніторингу та аналізу мережевого трафіку[34].

Переваги Wireshark:

1. *безкоштовність*: Wireshark є безкоштовним програмним забезпеченням, що робить його доступним для широкого кола користувачів без необхідності покупки ліцензії;

2. *кросплатформенність*: Програма підтримується на різних операційних системах, включаючи Windows, macOS і Linux, що дозволяє використовувати її на різних платформах;

3. *можливості аналізу*: Wireshark надає широкі можливості аналізу мережевого трафіку, включаючи фільтрацію пакетів, розбір протоколів і відстеження проблем у мережі;

4. *користувацька підтримка*: Існує активне спільнота користувачів Wireshark, що надає допомогу, поради та розв'язання проблем через форуми та інші ресурси;

5. *зручний інтерфейс*: Інтерфейс Wireshark є інтуїтивно зрозумілим і зручним у використанні, що дозволяє швидко знаходити необхідну інформацію.

Недоліки Wireshark:

1. *складність використання для новачків*: Для новачків у мережевому аналізі може знадобитися час і зусилля для вивчення всіх функцій та можливостей Wireshark;

2. *великий обсяг інформації*: Під час аналізу великих мереж Wireshark може генерувати великий обсяг даних, що потребує відповідного обладнання та ресурсів для обробки і аналізу;

3. *необхідність спеціалізованих знань*: Для повноцінного використання Wireshark може знадобитися розуміння мережевих протоколів та аспектів

мережевого аналізу, що може бути складним для користувачів без відповідних знань.

Рисунок 3.2 – Логотип SolarWinds

SolarWinds - це комплексне програмне забезпечення для моніторингу та управління мережами, яке надає широкий спектр можливостей для аналізу мережевого трафіку. Ця платформа розроблена з метою забезпечення адміністраторам мереж засобів для ефективного моніторингу, аналізу та оптимізації мережевої інфраструктури.

Однією з ключових функцій SolarWinds є можливість відслідковувати різні аспекти мережевого трафіку, включаючи пропускну здатність, використані ресурси, швидкість передачі даних та інші. Програмне забезпечення дає можливість візуалізації цих даних у вигляді графіків, діаграм та звітів, що дозволяє адміністраторам швидко реагувати на проблеми та приймати рішення.

SolarWinds також надає можливість моніторити роботу окремих мережевих пристроїв, таких як маршрутизатори, комутатори, файрволи тощо. Це дозволяє адміністраторам отримувати детальну інформацію про стан пристроїв та їх взаємодію з мережею.

Ще однією важливою функцією SolarWinds є можливість автоматизувати процеси моніторингу та управління мережею. За допомогою різних інтегрованих інструментів та API, адміністратори можуть створювати власні скрипти та автоматизовані рішення для вирішення конкретних завдань.

Крім того, SolarWinds надає інтегровані рішення для моніторингу безпеки мережі. Вона може виявляти потенційні загрози та аномальний трафік, а також забезпечувати контроль доступу до мережевих ресурсів.

Узагальнюючи, SolarWinds є потужним інструментом для моніторингу мереж, який надає адміністраторам засоби для ефективного контролю та управління мережевими ресурсами. Її розширені можливості та інтегровані

рішення роблять її однією з перших у виборі для комплексного моніторингу мережевого трафіку[25].

Переваги SolarWinds:

1. *широкий функціонал*. SolarWinds пропонує широкий набір інструментів для моніторингу мережі, включаючи моніторинг пропускної здатності, аналіз трафіку, моніторинг утилізації ресурсів та багато іншого;

2. *легкість використання*: Інтерфейс користувача SolarWinds є інтуїтивно зрозумілим та легким у використанні, що дозволяє швидко налаштувати моніторинг та отримувати інформацію про стан мережі;

3. *складні аналітичні можливості*: SolarWinds надає розширені аналітичні можливості, такі як виявлення аномалій у мережевому трафіку, прогнозування витрат пропускної здатності та аналіз тенденцій використання ресурсів;

4. *масштабованість*: SolarWinds може бути легко масштабований від невеликих мереж до великих корпоративних інфраструктур, що робить його відмінним вибором для організацій будь-якого розміру;

5. *підтримка*: Користувачі SolarWinds можуть користуватися широким спектром підтримки, включаючи документацію, форуми, вебінари та технічну підтримку, що дозволяє швидко вирішувати проблеми та отримувати допомогу.

Недоліки SolarWinds:

1. *вартість*: SolarWinds може бути витратним рішенням, особливо для невеликих організацій з обмеженим бюджетом;

2. *складність встановлення та налаштування*: Налаштування SolarWinds може бути складним завданням, особливо для користувачів з обмеженим досвідом у моніторингу мережі;

3. *обмежені можливості безпеки*: Деякі функції безпеки в SolarWinds можуть бути обмеженими порівняно з іншими рішеннями.

Рисунок 3.3 – Логотип PRTG Network Monitor

PRTG Network Monitor – це потужна програма для моніторингу мережевого трафіку, яка дозволяє адміністраторам моніторити, аналізувати та керувати мережевими ресурсами з одного центрального місця. Це програмне забезпечення від компанії Paessler AG забезпечує широкий спектр функцій для ефективного відслідковування та управління мережевим трафіком.

Однією з ключових переваг PRTG є його простота в установці та використанні. Після налаштування програма починає моніторити мережу автоматично, збираючи дані про трафік, пропускну здатність та інші метрики. Інтуїтивний веб-інтерфейс дозволяє легко переглядати ці дані та отримувати повідомлення про будь-які проблеми.

Одним з ключових компонентів PRTG є датчики, які дозволяють відслідковувати різні аспекти мережевої інфраструктури. Ці датчики можуть бути налаштовані для вимірювання пропускної здатності, рівня використання ресурсів, стану пристроїв та багато іншого. Завдяки широкому вибору датчиків, користувачі можуть налаштувати моніторинг мережі відповідно до своїх конкретних потреб.

PRTG також забезпечує різноманітні можливості аналізу мережевого трафіку. Користувачі можуть створювати графіки та звіти для візуалізації даних, використовувати фільтри для виявлення аномального трафіку та встановлювати сповіщення про будь-які проблеми чи відхилення від норми.

Однією з ключових переваг PRTG є його масштабованість. Від невеликих мереж до великих корпоративних інфраструктур, програма може легко впоратися з будь-яким обсягом даних та кількістю моніторингових об'єктів. Додаткові можливості, такі як розподілена архітектура та кластеризація, дозволяють забезпечити надійність та продуктивність моніторингу навіть у найбільших мережах.

Узагальнюючи, PRTG Network Monitor є інтегрованою та масштабованою платформою для моніторингу мережевого трафіку, яка надає адміністраторам засоби для ефективного контролю та управління мережевими ресурсами. Його широкий спектр функцій, простота використання та масштабованість роблять його одним із популярних виборів для комплексного моніторингу мереж[22].

Переваги PRTG Network Monitor:

1. *простота використання*: PRTG пропонує інтуїтивний та легкий у використанні інтерфейс, що дозволяє швидко встановити та налаштувати моніторинг мережі;

2. *широкий функціонал*: Цей інструмент надає широкий спектр функціоналу для моніторингу різних аспектів мережі, включаючи пропускну здатність, стан здоров'я пристроїв, сервісів та додатків;

3. *складні аналітичні можливості*: PRTG пропонує розширені аналітичні можливості, такі як виявлення аномального трафіку, прогнозування використання ресурсів та аналіз тенденцій у роботі мережі;

4. *масштабованість*: PRTG може бути масштабований для відповідності потребам різних розмірів мережі, від невеликих підприємств до великих корпоративних інфраструктур;

5. *гнучкість налаштування*: Інструмент надає гнучкі можливості налаштування для відображення та аналізу різних параметрів мережі, що дозволяє користувачам налаштовувати моніторинг під свої потреби.

Недоліки PRTG Network Monitor:

1. *вартість*: PRTG є комерційним рішенням, тому його вартість може бути вищою порівняно з деякими безкоштовними або альтернативами з відкритими початковим кодом;

2. *обмежені можливості безпеки*: Деякі функції безпеки можуть бути обмеженими в порівнянні з іншими рішеннями моніторингу мережі;

3. *підтримка*: Хоча PRTG надає підтримку для своїх користувачів, але рівень підтримки може варіюватися в залежності від обраного пакету абонемента.

3.3 Опис технологій та методів для розробки системи моніторингу

При аналізі існуючих рішень, таких як PRTG Network Monitor, SolarWinds та Wireshark, стає очевидним, що кожен з них має свої переваги та недоліки. Наприклад, Wireshark забезпечує детальний аналіз пакетів, що дозволяє виявляти конкретні проблеми в мережі, але в той же час вимагає значних знань і навичок для ефективного використання. PRTG Network Monitor і SolarWinds, незважаючи на свою легкість використання, можуть бути обмеженими в аналізі трафіку на рівні пакетів, що призводить до втрати деякої глибини деталізації[26].

Для розробки своєї системи моніторингу мережевого трафіку я буду спиратись на можливості Wireshark щодо швидкодії та детальності моніторингу мережі, доповнюючи це простим і зрозумілим інтерфейсом та додатковим аналізом мережевого трафіку, який дозволить поглибити можливості моніторингу мережі. Основні сильні сторони мого застосунку будуть: швидкодія, детальний аналіз, відкритий код, безкоштовність та легкість в освоєнні. Для поєднання сильних сторін існуючих рішень та зменшення недоліків я вирішив використовувати наступні технології та методи.

Для розробки даного застосунку я вибрав мову програмування C++ як основну, оскільки вона має кілька переваг, які роблять її ідеальним вибором для нашої задачі. Перш за все, C++ відома своєю високою швидкодією, що робить її чудовим вибором для розробки високопродуктивних програм, таких як системи моніторингу та аналізу мережевого трафіку. Низький рівень абстракції та прямий доступ до ресурсів комп'ютера відкриває можливість оптимізувати код для максимальної продуктивності.

Використання C++ дозволяє ефективно управляти пам'яттю та ресурсами, що особливо важливо для застосунків, які обробляють великі обсяги даних, такі як аналіз мережевого трафіку. Можна використовувати вказівники та ручне керування пам'яттю, щоб уникнути надмірного використання ресурсів та забезпечити оптимальну продуктивність нашого застосунку.

C++ є мовою загального призначення, яка має широкий спектр бібліотек та інструментів для розробки різноманітних застосунків. Є можливість використовувати різноманітні сторонні бібліотеки та інструменти, такі як Boost, libpcap та інші для розширення можливостей нашого застосунку та полегшення розробки.

Також C++ є мовою програмування, яка має широку спільноту розробників та обширну базу знань. Це означає, що можливо легко знайти підтримку, документацію та рішення для різних технічних питань, які можуть виникнути під час розробки застосунку[32].

Отже, використання C++ для розробки нашого застосунку для аналізу мережевого трафіку надає нам швидкодію, ефективне управління ресурсами та доступ до широкого спектру інструментів та ресурсів, що сприяє створенню потужного та ефективного програмного забезпечення.

Для даного застосунку буде використовуватись бібліотека Qt для візуалізації результатів через її широкі можливості та гнучкість у створенні користувацьких інтерфейсів. Qt - це потужний набір інструментів для розробки сучасних та ефективних графічних інтерфейсів, що відповідають потребам користувачів та поставленим задачам.

Однією з основних переваг використання Qt є його кросплатформенність. За допомогою Qt можна розробляти програмне забезпечення, яке працює на різних операційних системах, включаючи Windows, macOS та Linux, без необхідності писати окремий код для кожної платформи. Це забезпечує широку доступність нашого застосунку для користувачів незалежно від їх операційної системи.

Крім того, Qt надає розширений набір візуальних елементів, таких як кнопки, поля введення, таблиці, графіки тощо, які довзолять створювати зручні та ефективні інтерфейси для взаємодії з користувачем. Є можливість легко налаштувати вигляд та поведінку кожного елементу інтерфейсу для найкращого відображення результатів нашого аналізу мережевого трафіку.

За допомогою Qt можна також інтегрувати графічний інтерфейс з рештою нашого застосунку, забезпечуючи плавну та зручну взаємодію з користувачем.

Можна відображати результати аналізу мережевого трафіку у зручній та інтуїтивно зрозумілій формі, використовуючи графіки, діаграми, таблиці та інші елементи інтерфейсу[21].

Для ефективного аналізу мережевого трафіку у застосунку вирішено використовувати flow-based аналіз. Цей підхід дозволяє згруповувати дані про потоки даних, що проходять через мережу, та швидко отримувати інформацію про їх характеристики та зразки використання.

Flow-based аналіз дозволяє групувати пакети у потоки на основі різних характеристик, таких як IP-адреси джерела та призначення, порти джерела та призначення, протоколи тощо. Групування пакетів у потоки дозволяє аналізувати характеристики цих потоків, такі як тривалість, обсяг даних, частота, середня швидкість тощо.

Однією з переваг flow-based аналізу є його швидкодія. Замість того, щоб аналізувати кожен окремий пакет, можна працювати з потоками даних, що дозволяє зменшити кількість оброблюваних даних та знизити витрати ресурсів обчислення.

Крім того, flow-based аналіз надає можливість легко виявляти паттерни та аномалії у мережевому трафіку. Можливо виявляти незвичні або підозрілі потоки даних, такі як надмірно великі обсяги даних, несподівані порти або протоколи, довгі тривалості тощо, що може свідчити про атаку або ненормальну активність у мережі.

Отже, використання flow-based аналізу у застосунку для аналізу мережевого трафіку дозволяє ефективно та швидко аналізувати великі обсяги даних, виявляти аномалії та патерни у мережевому трафіку та забезпечує потужний інструмент для моніторингу та безпеки мережі.

Для покращення деталізації та точності аналізу мережевого трафіку у буде використаний аналіз пакетів після flow-based аналізу. Хоча flow-based аналіз дозволяє швидко виявляти загальні паттерни та аномалії у мережевому трафіку, аналіз пакетів надасть більш детальну та глибоку інформацію про кожен окремий

пакет, що дозволить отримати більше інформації та зрозуміти причину аномалій, виявлених на рівні потоків.

Після flow-based аналізу можна використовувати аналіз пакетів для розбору кожного пакету у потоці та витягування додаткових даних та метаінформації. Можна аналізувати заголовки пакетів, такі як IP-адреси, порти, протоколи тощо, а також вміст пакетів, щоб зрозуміти їхню природу та мету.

Використання аналізу пакетів дозволить виявити додаткові аномалії та паттерни, які можуть бути приховані на рівні потоків. Легшим буде виявлення ненормальної поведінки певних пакетів, такі як незвичні або недійсні заголовки, аномально великі або малі розміри пакетів, підозрілі типи або зміст тощо, що може вказувати на потенційні загрози або проблеми у мережевому трафіку.

Таким чином, використання аналізу пакетів після flow-based аналізу дозволяє отримати більш детальний, точний та повний образ мережевого трафіку. Цей комплексний підхід дозволяє ефективно виявляти та аналізувати аномалії, загрози та паттерни у мережевому трафіку, забезпечуючи можливість для ефективного моніторингу та захисту мережі.

Використання аналізу пакетів та flow-based аналізу в комбінації надає численні переваги для аналізу мережевого трафіку. Flow-based аналіз дозволяє швидко групувати та агрегувати дані про потоки даних, що проходять через мережу, дозволяючи виявити загальні паттерни та аномалії. Це спрощує процес аналізу та дозволяє оперативно виявляти незвичайну активність у мережі.

З іншого боку, аналіз пакетів надає більш детальну та глибоку інформацію про кожен окремий пакет, що дозволяє отримати більше інформації та розуміння природи мережевої активності. Цей підхід дозволяє виявити ненормальну поведінку окремих пакетів, таку як аномалії у заголовках, незвичайний зміст або невідомі протоколи.

Спільне використання цих двох методів дозволяє отримати комплексний та повний образ мережевого трафіку. Він поєднує швидкість та ефективність flow-based аналізу з детальністю та точністю аналізу пакетів. Ця комбінація дозволяє

ефективно виявляти аномалії, паттерни та загрози у мережі, забезпечуючи комплексний підхід до моніторингу та захисту мережевого середовища.

Для даного застосунку будемо використовуватися Wireshark для захоплення мережевого трафіку в реальному часі та подальшого аналізу пакетів, а також для виконання flow-based аналізу. Wireshark відомий своєю потужністю та багатофункціональністю у сфері аналізу мережевого трафіку і є стандартом для таких завдань серед професіоналів з безпеки мережі.

За допомогою Wireshark можна безпосередньо захоплювати пакети, які переходять через мережевий інтерфейс системи в реальному часі. Це дозволить отримувати оновлені дані про мережевий трафік у момент їх передачі та аналізувати їх для виявлення аномалій, загроз та інших цікавих паттернів.

Одним із ключових функціональних можливостей Wireshark є його здатність докладно аналізувати окремі пакети, розбираючи їх на складові частини, включаючи заголовки протоколів, дані та іншу метаінформацію. Це надає можливість докладно розібрати кожен пакет і зрозуміти, що саме відбувається у мережі.

Поміж іншого, Wireshark також надає можливості для виконання flow-based аналізу, який дозволяє агрегувати дані про потоки даних, що переходять через мережу. Це дозволяє аналізувати та виявляти зразки використання мережевого трафіку, такі як частота та тривалість певних типів з'єднань, обсяг переданих даних тощо.

Отже, використання Wireshark для захоплення, аналізу пакетів та flow-based аналізу надасть необхідні інструменти та дані для виявлення аномалій, загроз та інших цікавих взаємозв'язків у мережевому трафіку нашої системи.

Для даного застосунку планується використовувати бібліотеку PyTorch разом з мовою програмування C++, щоб реалізувати машинне навчання для аналізу мережевого трафіку. PyTorch - це одна з найпопулярніших бібліотек для машинного навчання та глибокого навчання, яка надає зручні та потужні інструменти для розробки та навчання моделей штучного інтелекту.

Використання PyTorch у поєднанні з C++ дозволяє реалізувати ефективні алгоритми машинного навчання безпосередньо у застосунку для аналізу мережевого трафіку. Завдяки швидкодії та ефективності C++, можна оптимізувати роботу моделей та забезпечити їх високу продуктивність навіть для обробки великих обсягів даних.

Однією з основних переваг PyTorch є його гнучкість та простота використання. Завдяки побудові на графі обчислень, PyTorch дозволяє легко створювати та навчати складні моделі штучного інтелекту, такі як нейронні мережі, зручним та інтуїтивно зрозумілим способом.

Крім того, PyTorch має велику спільноту користувачів та розробників, що сприяє постійному розвитку та підтримці бібліотеки. Використовування цього ресурсу для знаходження рішень на основі найновіших досягнень у галузі машинного навчання для аналізу мережевого трафіку полегшить розробку даного застосунку.

Отже, використання PyTorch з C++ дозволяє створити потужний та ефективний інструмент для аналізу мережевого трафіку, який може виявляти аномалії, загрози та паттерни у мережевому трафіку за допомогою методів машинного навчання[27].

ВИСНОВКИ

В ході кваліфікаційної роботи була розроблена система моніторингу та аналізу мережевого трафіку, яка має великий потенціал у виявленні аномалій, оптимізації мережевої інфраструктури та підвищенні рівня кібербезпеки. Дана система поєднує у собі кілька ключових компонентів, що дозволяє проводити детальний аналіз мережевого трафіку у реальному часі та на рівні даних пакетів.

Однією з ключових особливостей розробленої системи є використання мови програмування C++ та фреймворку Qt для розробки гнучкого та продуктивного програмного забезпечення. Ці технології дозволяють ефективно працювати з мережевими даними та створювати потужні інструменти для моніторингу й аналізу мережі.

Для захоплення пакетів з мережі у реальному часі та подальшого аналізу використовується Wireshark, яка надає широкі можливості для роботи з мережевими протоколами та структурами даних. Це дозволяє системі ефективно захоплювати великий обсяг трафіку та аналізувати його у реальному часі.

Крім того, для аналізу трафіку використовується метод flow-based аналізу, який дозволяє швидко оцінювати загальні тенденції та паттерни у мережі. Цей підхід доповнює аналіз на рівні пакетів та дозволяє системі отримувати узагальнену статистику про трафік.

Окремо варто відзначити використання бібліотеки PyTorch для машинного навчання, що дозволяє системі виявляти складні паттерни та аномалії в мережевому трафіку, що може бути важко виявити за допомогою традиційних методів аналізу.

У підсумку, розроблена система моніторингу та аналізу мережевого трафіку є потужним інструментом для забезпечення безпеки, ефективності та надійності мережі. Вона дозволяє оперативно виявляти та реагувати на проблеми у мережі, забезпечуючи надійну роботу інфраструктури та підвищуючи загальний рівень кібербезпеки.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ashtari H. What is network traffic analysis? Definition, importance, implementation, and best practices. spiceworks.com. URL: https://www.spiceworks.com/tech/networking/articles/network-traffic-analysis/ (дата звернення: 01.05.2024).
2. Conrad E., Misenar S., Feldman J. CISSP study guide. 3-тє вид. 2016. С. 219–291. URL: https://www.sciencedirect.com/science/article/abs/pii/B9780128024379000059 (дата звернення: 29.04.2024).
3. Dharmik T. Modern network security: issues and challenges. medium.com. URL: https://medium.com/@tejas.dharmik20/modern-network-security-issues-and-challenges-338aaedc44fc (дата звернення: 01.05.2024).
4. Exploring the Determinants of Network Effectiveness: The Case of Neighborhood Governance Networks in Beijing. 2015Journal of Public Administration Research and Theory. 2015. Vol. 26, no. 2. P. 22–37. URL: https://doi.org/10.1093/jopart/muv017 (дата звернення: 01.05.2024).
5. Gorman B. TCP vs UDP: what's the difference and which protocol is better?. Avast. URL: https://www.avast.com/c-tcp-vs-udp-difference#:~:text=What%20are%20TCP%20and%20UDP,is%20more%20important%20than%20reliability. (дата звернення: 01.05.2024).
6. MrBrownCS. Network Protocols #2: HTTP, FTP, SNMP, POP, SMTP, & ICMP, 2021. YouTube. URL: https://www.youtube.com/watch?v=MR0AC9vGir8 (дата звернення: 01.05.2024).
7. Network Monitoring Software by ManageEngine OpManager. ManageEngine OpManager. URL: https://www.manageengine.com/network-monitoring/basics-of-network-monitoring.html (дата звернення: 01.05.2024).

8. Network Monitoring Software by ManageEngine OpManager. ManageEngine OpManager. URL: https://www.manageengine.com/network-monitoring/network-protocols.html (дата звернення: 01.05.2024).

9. Network Security. GeeksforGeeks. URL: https://www.geeksforgeeks.org/network-security/ (дата звернення: 01.05.2024).

10. Network Sniffers: What Are They and How Can I Use Them?. PagerDuty. URL: https://www.pagerduty.com/resources/learn/what-are-network-sniffers/ (date of access: 01.05.2024).

11. Network Traffic Types. IPCisco. URL: https://ipcisco.com/lesson/network-traffic-types/ (дата звернення: 01.05.2024).

12. PowerCert Animated Videos. Network ports explained, 2022. YouTube. URL: https://www.youtube.com/watch?v=g2fT-g9PX9o (дата звернення: 01.05.2024).

13. Schrader D. Understanding the 4 Types of Network Monitoring Tools and Comparing Available Solutions. Netwrix Blog | Insights for Cybersecurity and IT Pros. URL: https://blog.netwrix.com/2023/12/27/network-monitoring-tools/#:~:text=Performance-focused%20network%20device%20monitoring,capture%20analysis%20and%20streaming%20telemetry. (дата звернення: 01.05.2024).

14. What is a computer port? | Ports in networking. Cloudflare. URL: http://www.cloudflare.com/learning/network-layer/what-is-a-computer-port/ (дата звернення: 01.05.2024).

15. What is a packet? | Network packet definition. cloudflare.com. URL: http://www.cloudflare.com/learning/network-layer/what-is-a-packet/ (дата звернення: 01.05.2024).

16. What is a Protocol Analyzer? Basics & Benefits - ManageEngine NetFlow Analyzer. ManageEngine: IT security, operations & service management. URL: https://www.manageengine.com/products/netflow/what-is-a-protocol-analyzer.html (дата звернення: 02.05.2024).

17. What is a router? | Router definition. Cloudflare. URL: http://www.cloudflare.com/learning/network-layer/what-is-a-router/ (дата звернення: 27.04.2024).

18. What is network capacity?. SolarWinds. URL: https://www.solarwinds.com/resources/it-glossary/network-capacity (дата звернення: 02.05.2024).

19. What is Network Monitoring? | IBM. IBM in Deutschland, Österreich und der Schweiz. URL: https://www.ibm.com/topics/network-monitoring (дата звернення: 01.05.2024).

20. What is Network Traffic Monitoring? | Definition & Basics - ManageEngine NetFlow Analyzer. ManageEngine: IT security, operations & service management. URL: https://www.manageengine.com/products/netflow/basics-of-network-traffic-monitoring.html (дата звернення: 01.05.2024).

21. About The Qt Group - Code Less & Create More. Qt | Tools for Each Stage of Software Development Lifecycle. URL: https://www.qt.io/group (дата звернення: 02.05.2024).

22. Discover the 3 Paessler PRTG monitoring solutions. Home. URL: https://www.paessler.com/prtg?gad_source=1&gclid=Cj0KCQjw6PGxBhCVARIsAIumnWboIJXTU-LXPSm3E3W1yS4LjvIHYnBDUbtKFl450eun-vM0CEMPXeQaAr9zEALw_wcB (дата звернення: 02.05.2024).

23. Maudoux C., Boumerdassi S. Network anomalies detection by unsupervised activity deviations extraction. 2022 global information infrastructure and networking symposium (GIIS). 2022.

24. Mohanakrishnan R. Top 10 best practices for network monitoring in 2023 - spiceworks. Spiceworks Inc. URL: https://www.spiceworks.com/tech/networking/articles/network-monitoring-best-practices/ (дата звернення: 29.04.2024).

25. Observability and IT management platform | solarwinds. Observability and IT Management Platform | SolarWinds. URL: https://www.solarwinds.com/ (дата звернення: 02.05.2024).

26. Pireddu M. How to create an app - 7 steps to make an app in 2024. GoodBarber. URL: https://www.goodbarber.com/blog/how-to-make-an-app/ (дата звернення: 01.05.2024).

27. PyTorch. PyTorch. URL: https://pytorch.org/ (дата звернення: 02.05.2024).

28. RITSEC. Traffic analysis with machine learning - camden robertson, 2023. YouTube. URL: https://www.youtube.com/watch?v=HZ-dLMWI_uI (дата звернення: 30.04.2024).

29. Sanders C., Smith J. Packet analysis. Applied network security monitoring. 2014. С. 341–384. URL: https://doi.org/10.1016/b978-0-12-417208-1.00013-1 (дата звернення: 29.04.2024).

30. Schnorr L. M., Legrand A., Vincent J.-M. Detection and analysis of resource usage anomalies in large distributed systems through multi-scale visualization. Concurrency and computation: practice and experience. 2011. Т. 24, № 15. С. 1792–1816. URL: https://doi.org/10.1002/cpe.1885 (дата звернення: 09.05.2024).

31. The Alan Turing Institute. Statistical analysis of networks - professor gesine reinert, university of oxford, 2018. YouTube. URL: https://www.youtube.com/watch?v=QT2xj9k00q0 (дата звернення: 29.04.2024).

32. Introduction to C++ Programming Language - GeeksforGeeks. GeeksforGeeks. URL: https://www.geeksforgeeks.org/introduction-to-c-programming-language/ (дата звернення: 08.05.2024).

33. The Data Beast. Graph network in data science::part-1. Medium. URL: https://medium.com/@thedatabeast/graph-network-in-data-science-part-1-77971ddd9b4 (дата звернення: 01.05.2024).

34. Wireshark go deep. Wireshark. URL: https://www.wireshark.org/ (дата звернення: 02.05.2024).

35. Zagatto E. A. G., Vida A. C. F., Worsfold P. J. Flow analysis: overview☆. Reference module in chemistry, molecular sciences and chemical engineering. 2016. С. 213–219. URL: https://doi.org/10.1016/b978-0-12-409547-2.12202-4 (дата звернення: 30.04.2024).

Додаток А

КОД ЗАСТОСУНКУ З МОНІТОРИНГУ

flowinfo.h:

```
#ifndef FLOWINFO_H
#define FLOWINFO_H

#include <string>
#include <ctime>

struct FlowInfo {
    std::string srcIP;
    std::string dstIP;
    uint16_t srcPort;
    uint16_t dstPort;
    std::string protocol;
    size_t packetCount;
    size_t byteCount;
    std::time_t lastSeen;
};

#endif // FLOWINFO_H
```

packetinfo.h

```
#ifndef PACKETINFO_H
#define PACKETINFO_H

#include <string>
#include <ctime>

struct PacketInfo {
    std::string srcIP;
    std::string dstIP;
    uint16_t srcPort;
    uint16_t dstPort;
    std::string protocol;
    std::time_t timestamp;
    uint32_t length;
    friend bool operator<(const PacketInfo& l, const PacketInfo& r)
    {
        return l.srcIP < r.srcIP;
    }
};

#endif // PACKETINFO_H
```

anomalydetector.h

```
#ifndef ANOMALYDETECTOR_H
#define ANOMALYDETECTOR_H

#include <torch/script.h>
#include <torch/torch.h>

#include <string>
#include <vector>

class AnomalyDetector {
public:
    AnomalyDetector(const std::string &modelPath);
    bool isAnomalous(const std::vector<float> &features);

private:
    torch::jit::script::Module model;
    torch::Tensor normalize(const torch::Tensor &tensor);
    torch::Tensor toTensor(const std::vector<float> &features);
};

#endif // ANOMALYDETECTOR_H
```

anomalydetector.cpp

```
#include "anomalydetector.h"
```

```
AnomalyDetector::AnomalyDetector(const std::string
&modelPath) {
    try {
        model = torch::jit::load(modelPath);
    } catch (const c10::Error &e) {
        std::cerr << "Error loading the model\n";
    }
}

bool AnomalyDetector::isAnomalous(const std::vector<float>
&features) {
    torch::Tensor inputTensor = normalize(toTensor(features));
    torch::Tensor outputTensor =
model.forward({inputTensor}).toTensor();
    float mse = torch::mse_loss(inputTensor,
outputTensor).item<float>();
    return mse > 0.01;
}

torch::Tensor AnomalyDetector::normalize(const torch::Tensor
&tensor) {
    return tensor;
}

torch::Tensor AnomalyDetector::toTensor(const
std::vector<float> &features) {
    return torch::from_blob((void *)features.data(), {1,
(int64_t)features.size()}, torch::kFloat32);
}
```

packetcapturethread.h

```
#ifndef PACKETCAPTURETHREAD_H
#define PACKETCAPTURETHREAD_H

#include <QObject>
#include <QThread>
#include <QMutex>
#include <vector>
#include <string>
#include <map>
#include <pcap.h>
#include "packetinfo.h"
#include "flowinfo.h"
#include <mutex>
class PacketCaptureThread : public QThread
{
    Q_OBJECT

public:
    explicit PacketCaptureThread(QObject *parent = nullptr);
    void run() override;
    void stop();
    std::vector<PacketInfo> getPackets() const;
    std::map<std::string, FlowInfo> getFlows() const;

signals:
    void packetCaptured();
    void flowUpdated();
    void errorOccurred(const QString &errorMsg);

private:
    bool running;
    pcap_t *handle;
    mutable std::mutex dataMutex;
    std::vector<PacketInfo> packets;
    std::map<std::string, FlowInfo> flows;

    static void packetHandler(u_char *userData, const
pcap_pkthdr *pkthdr, const u_char *packet);
    void processPacket(const pcap_pkthdr *pkthdr, const u_char
*packet);
};

#endif // PACKETCAPTURETHREAD_H
```

packetcapturethread.cpp

```
#include "packetcapturethread.h"
#include <QDebug>
#include <pcap.h>
```

```
#include <netinet/ether.h>

#include <netinet/ip.h>

#include <netinet/tcp.h>

#include <netinet/udp.h>

#include <arpa/inet.h>

#include <cstring>

#include <QMutexLocker>


PacketCaptureThread::PacketCaptureThread(QObject *parent)
: QThread(parent), running(false), handle(nullptr) {}


void PacketCaptureThread::run() {

    char errbuf[PCAP_ERRBUF_SIZE];

    pcap_if_t *allDevs;


    if (pcap_findalldevs(&allDevs, errbuf) == -1) {

        emit errorOccurred(errbuf);

        return;

    }


    if (!allDevs) {

        emit errorOccurred("No devices found.");

        return;

    }


    handle = pcap_open_live(allDevs->name, BUFSIZ, 1, 1000,
errbuf);

    if (!handle) {

        emit errorOccurred(errbuf);

        return;

    }


    running = true;

    while (running) {

        pcap_dispatch(handle, 10, packetHandler,
reinterpret_cast<u_char *>(this));

    }


    pcap_close(handle);

    pcap_freealldevs(allDevs);

}
```

```
void PacketCaptureThread::stop() {

    running = false;

}


std::vector<PacketInfo> PacketCaptureThread::getPackets()
const {

    std::lock_guard<std::mutex> lock(dataMutex);

    return packets;

}


std::map<std::string, FlowInfo>
PacketCaptureThread::getFlows() const {

    std::lock_guard<std::mutex> lock(dataMutex);

    return flows;

}


void PacketCaptureThread::packetHandler(u_char *userData,
const pcap_pkthdr *pkthdr, const u_char *packet) {

    auto *thread = reinterpret_cast<PacketCaptureThread
*>(userData);

    thread->processPacket(pkthdr, packet);

}


void PacketCaptureThread::processPacket(const pcap_pkthdr
*pkthdr, const u_char *packet) {

    struct ether_header *eth_header = (struct ether_header
*)packet;


    if (ntohs(eth_header->ether_type) == ETHERTYPE_IP) {

        struct ip *ip_header = (struct ip*)(packet + sizeof(struct
ether_header));

        std::string src_ip = inet_ntoa(ip_header->ip_src);

        std::string dst_ip = inet_ntoa(ip_header->ip_dst);

        std::string protocol;

        uint16_t src_port = 0;

        uint16_t dst_port = 0;


        switch (ip_header->ip_p) {

        case IPPROTO_TCP: {

            protocol = "TCP";

            struct tcphdr *tcp_header = (struct tcphdr*)(packet +
sizeof(struct ether_header) + sizeof(struct ip));
```

```
            src_port = ntohs(tcp_header->source);
            dst_port = ntohs(tcp_header->dest);
            break;
        }
        case IPPROTO_UDP: {
            protocol = "UDP";
            struct udphdr *udp_header = (struct udphdr*)(packet +
sizeof(struct ether_header) + sizeof(struct ip));
            src_port = ntohs(udp_header->source);
            dst_port = ntohs(udp_header->dest);
            break;
        }
        case IPPROTO_ICMP:
            protocol = "ICMP";
            break;
        default:
            protocol = "Other";
            break;
        }

        PacketInfo packetInfo{src_ip, dst_ip, src_port, dst_port,
protocol, pkthdr->ts.tv_sec, pkthdr->len};
        {
            std::lock_guard<std::mutex> lock(dataMutex);
            packets.push_back(packetInfo);
        }

        emit packetCaptured();

        std::string flowKey = packetInfo.srcIP + "->" +
packetInfo.dstIP + ":" + packetInfo.protocol;
        {
            std::lock_guard<std::mutex> lock(dataMutex);
            if (flows.find(flowKey) == flows.end()) {
                auto &flow = flows[flowKey];
                flow.srcIP = packetInfo.srcIP;
                flow.dstIP = packetInfo.dstIP;
                flow.srcPort = packetInfo.srcPort;
                flow.dstPort = packetInfo.dstPort;
                flow.protocol = packetInfo.protocol;
                flow.packetCount++;
                flow.byteCount += packetInfo.length;
                flow.lastSeen = packetInfo.timestamp;
            } else {
                FlowInfo &flow = flows[flowKey];
                flow.packetCount++;
                flow.byteCount += packetInfo.length;
                flow.lastSeen = packetInfo.timestamp;
            }
        }

        emit flowUpdated();
    }
}
```

mainwindow.h

```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include "anomalydetector.h"
#include "packetcapturethread.h"
#include <QMainWindow>
#include <QTableWidget>
#include <QLabel>

QT_BEGIN_NAMESPACE
namespace Ui { class MainWindow; }
QT_END_NAMESPACE

class MainWindow : public QMainWindow
{
    Q_OBJECT

public:
    MainWindow(QWidget *parent = nullptr);
    ~MainWindow();

private slots:
```

```
    void updatePacketTable();
    void updateFlowTable();
    void handleError(const QString &errorMsg);

private:
    Ui::MainWindow *ui;
    PacketCaptureThread packetCaptureThread;
    QTableWidget *packetTable;
    QTableWidget *flowTable;
    QLabel *statusLabel;
    AnomalyDetector anomalyDetector;
};

#endif // MAINWINDOW_H
```

mainwindow.cpp

```
#include "mainwindow.h"
#include "ui_mainwindow.h"
#include <QMessageBox>

MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
    , packetCaptureThread(this)
    , anomalyDetector("/home/gazr5/My/nothing/build-untitled-
Desktop-Debug/model.pt")
{
    ui->setupUi(this);
    packetTable = new QTableWidget(this);
    packetTable->setColumnCount(5);
    packetTable->setHorizontalHeaderLabels({"Source IP",
"Destination IP", "Source Port", "Destination Port",
"Protocol"});

    // Initialize flow table
    flowTable = new QTableWidget(this);
    flowTable->setColumnCount(6);
    flowTable->setHorizontalHeaderLabels({"Source IP",
"Destination IP", "Protocol", "Packets", "Bytes", "Last
Seen"});

    ui->tabWidget->removeTab(0);
    ui->tabWidget->removeTab(0);
    ui->tabWidget->addTab(packetTable, "Packets");
    ui->tabWidget->addTab(flowTable, "Flows");

    statusLabel = new QLabel("Status: Monitoring", this);
    ui->statusbar->addWidget(statusLabel);

    connect(&packetCaptureThread,
&PacketCaptureThread::packetCaptured, this,
&MainWindow::updatePacketTable);
    connect(&packetCaptureThread,
&PacketCaptureThread::flowUpdated, this,
&MainWindow::updateFlowTable);
    connect(&packetCaptureThread,
&PacketCaptureThread::errorOccurred, this,
&MainWindow::handleError);

    packetCaptureThread.start();
}

MainWindow::~MainWindow() {
    delete ui;
    packetCaptureThread.stop();
    packetCaptureThread.wait();
}

void MainWindow::updatePacketTable() {
    auto packets = packetCaptureThread.getPackets();
    packetTable->setRowCount(packets.size());

    for (size_t i = 0; i < packets.size(); ++i) {
        const auto &packet = packets[i];
        packetTable->setItem(i, 0, new
QTableWidgetItem(QString::fromStdString(packet.srcIP)));
        packetTable->setItem(i, 1, new
QTableWidgetItem(QString::fromStdString(packet.dstIP)));
        packetTable->setItem(i, 2, new
QTableWidgetItem(QString::number(packet.srcPort)));
        packetTable->setItem(i, 3, new
QTableWidgetItem(QString::number(packet.dstPort)));
        packetTable->setItem(i, 4, new
QTableWidgetItem(QString::fromStdString(packet.protocol)));
```

```
        std::vector<float> features =
{static_cast<float>(packet.protocol == "TCP" ? 1 :
packet.protocol == "UDP" ? 2 : 0),
                                static_cast<float>(packet.length)};
        if (anomalyDetector.isAnomalous(features)) {
            statusLabel->setText("Status: Anomaly Detected!");
        }
    }
    packetTable->resizeColumnsToContents();
    statusLabel->setText("Status: Monitoring");
}

void MainWindow::updateFlowTable() {
    auto flows = packetCaptureThread.getFlows();
    flowTable->setRowCount(flows.size());

    size_t row = 0;
    for (const auto &flow : flows) {
        flowTable->setItem(row, 0, new
QTableWidgetItem(QString::fromStdString(flow.second.srcIP
+ ":" + std::to_string(flow.second.srcPort))));
        flowTable->setItem(row, 1, new
QTableWidgetItem(QString::fromStdString(flow.second.dstIP
+ ":" + std::to_string(flow.second.dstPort))));
        flowTable->setItem(row, 2, new
QTableWidgetItem(QString::fromStdString(flow.second.protoc
ol)));
        flowTable->setItem(row, 3, new
QTableWidgetItem(QString::number(flow.second.packetCount
)));
        flowTable->setItem(row, 4, new
QTableWidgetItem(QString::number(flow.second.byteCount)))
;
        flowTable->setItem(row, 5, new
QTableWidgetItem(QString::fromStdString(std::ctime(&flow.s
econd.lastSeen))));
        ++row;
    }
    flowTable->resizeColumnsToContents();
}

void MainWindow::handleError(const QString &errorMsg) {
    QMessageBox::critical(this, "Error", errorMsg);
}
```

main.cpp

```
#include "mainwindow.h"
#include <QApplication>

int main(int argc, char *argv[]) {
    QApplication app(argc, argv);

    MainWindow window;
    window.show();

    return app.exec();
}
```

mainwindow.ui

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
 <class>MainWindow</class>
 <widget class="QMainWindow" name="MainWindow">
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>0</y>
    <width>1920</width>
    <height>1080</height>
   </rect>
  </property>
  <property name="windowTitle">
   <string>MainWindow</string>
  </property>
  <widget class="QWidget" name="centralwidget">
   <widget class="QTabWidget" name="tabWidget">
    <property name="geometry">
     <rect>
      <x>0</x>
      <y>0</y>
      <width>1730</width>
      <height>980</height>
```

```
    </rect>
   </property>
   <property name="currentIndex">
    <number>0</number>
   </property>
   <widget class="QWidget" name="tab">
    <attribute name="title">
     <string>Tab 1</string>
    </attribute>
   </widget>
   <widget class="QWidget" name="tab_2">
    <attribute name="title">
     <string>Tab 2</string>
    </attribute>
   </widget>
  </widget>
 </widget>
 <widget class="QMenuBar" name="menubar">
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>0</y>
    <width>1920</width>
    <height>22</height>
   </rect>
  </property>
 </widget>
 <widget class="QStatusBar" name="statusbar"/>
</widget>
<resources/>
<connections/>
</ui>
```

CMakeLists.txt

```
cmake_minimum_required(VERSION 3.5)

project(SomeTest VERSION 0.1 LANGUAGES CXX)

set(CMAKE_AUTOUIC ON)
set(CMAKE_AUTOMOC ON)
set(CMAKE_AUTORCC ON)

set(CMAKE_CXX_STANDARD 17)
set(CMAKE_CXX_STANDARD_REQUIRED ON)

set(PCAP_LIBRARY pcap)

set(CMAKE_PREFIX_PATH "/home/gazr5/My/libtorch")
set(Torch_DIR "/home/gazr5/My/libtorch/share/cmake/Torch")

find_package(Torch REQUIRED)

find_package(QT NAMES Qt6 Qt5 REQUIRED
COMPONENTS Widgets Network Core Gui)
find_package(Qt${QT_VERSION_MAJOR} REQUIRED
COMPONENTS Widgets Network Core Gui)

set(PROJECT_SOURCES
        main.cpp
        mainwindow.cpp
        mainwindow.h
        packetcapturethread.h
        packetcapturethread.cpp
        anomalydetector.h
        anomalydetector.cpp
        packetinfo.h
        flowinfo.h
        mainwindow.ui
)

if(${QT_VERSION_MAJOR} GREATER_EQUAL 6)
    qt_add_executable(SomeTest
        MANUAL_FINALIZATION
        ${PROJECT_SOURCES}
    )
else()
    if(ANDROID)
```

```
    add_library(SomeTest SHARED
      ${PROJECT_SOURCES}
    )
  else()
    add_executable(SomeTest
      ${PROJECT_SOURCES}
    )
  endif()
endif()

target_link_libraries(SomeTest PRIVATE
  Qt${QT_VERSION_MAJOR}::Widgets
  Qt${QT_VERSION_MAJOR}::Gui
  Qt${QT_VERSION_MAJOR}::Core
  ${PCAP_LIBRARY}
  "${TORCH_LIBRARIES}"
)

set_target_properties(SomeTest PROPERTIES
   MACOSX_BUNDLE_GUI_IDENTIFIER my.example.com
  MACOSX_BUNDLE_BUNDLE_VERSION
${PROJECT_VERSION}

  MACOSX_BUNDLE_SHORT_VERSION_STRING
${PROJECT_VERSION_MAJOR}.${PROJECT_VERSION_
MINOR}
  MACOSX_BUNDLE TRUE
  WIN32_EXECUTABLE TRUE
)

install(TARGETS SomeTest
  BUNDLE DESTINATION .
  LIBRARY DESTINATION
${CMAKE_INSTALL_LIBDIR}
  RUNTIME DESTINATION
${CMAKE_INSTALL_BINDIR}
)

if(QT_VERSION_MAJOR EQUAL 6)
  qt_finalize_executable(SomeTest)
endif()
```

Модель для знаходження аномалій мережевого трафіку:
https://github.com/nb0309/Network-Traffic-Analysis-using-Machine-learning/tree/main

Додаток Б

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ (ПРЕЗЕНТАЦІЯ)

Продовження додатку Б

Державний Університет
Інформаційно-Комунікаційних Технологій
Факультет Інформаційних технологій
Кафедра Комп'ютерної інженерії

Бакалаврська робота

на тему:
«Розробка системи моніторингу мережевого трафіку»

Виконав студент групи КІД-41
Мальченко Гліб Сергійович
Керівник дипломної роботи:
Бученко Ігор Анатолійович,
старший викладач кафедри КІ

Київ – 2024

Продовження додатку Б

Об`єкт дослідження – Розробка системи моніторингу мережевого трафіку

Предмет дослідження – Моніторинг мережевого трафіку

Мета роботи – Аналіз та вдосконалення існуючих рішень для моніторингу мережевого трафіку

2

Продовження додатку Б

Актуальність обраної теми

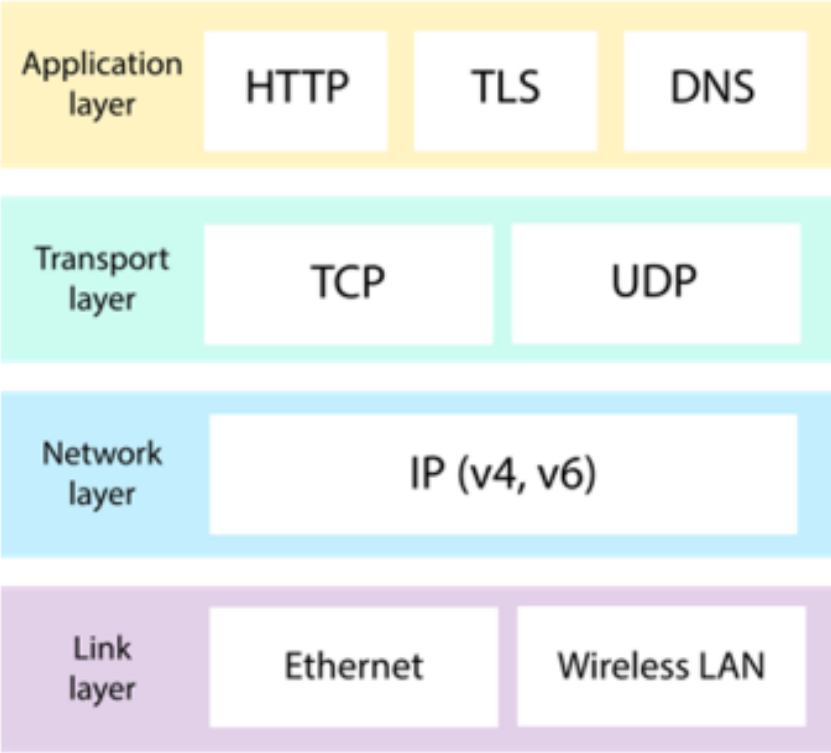
Створення системи моніторингу мережевого трафіку надзвичайно актуальне у сучасному світі, де мережеві технології є основою багатьох аспектів нашого життя. Забезпечення ефективності, безпеки та доступності мережевих ресурсів стає важливою задачею для бізнесу та організацій будь-якого масштабу. Впровадження систем моніторингу допомагає вчасно виявляти та вирішувати проблеми, що може покращити продуктивність та ефективність роботи мережі.

3

Продовження додатку Б

Основні поняття мережі

- *Пакети даних*
- *Протоколи*
- *Порти*

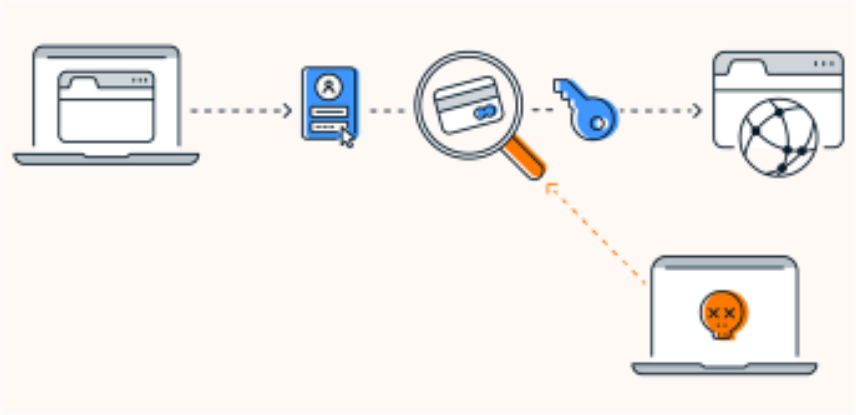


4

Продовження додатку Б

Методи та засоби моніторингу мережевого трафіку

- Мережеві аналізатори або сніфери



- Протокольні аналізатори

5

Продовження додатку Б

Цілі та завдання моніторингу

1. Відстеження стану мережі;

2. виявлення аномалій;

3. моніторинг безпеки;

4. аналіз трендів;

5. вирішення проблем.

Методи аналізу мережевого трафіку

1. Аналіз пакетів;
2. Flow-based аналіз;
3. статистичний аналіз;
4. машинне навчання;
5. графічний аналіз.

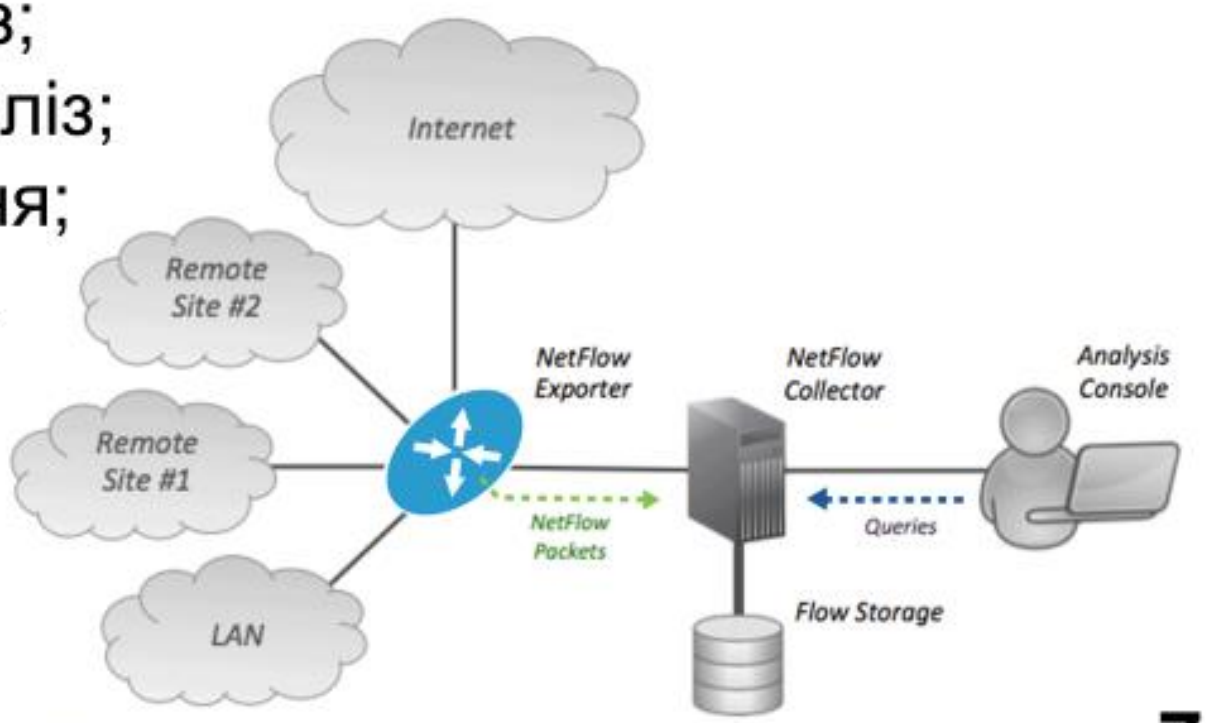


Image source: Cisco

7

Продовження додатку Б

Огляд існуючих рішень для моніторингу

WIRESHARK

PRTG Network Monitor

solarwinds

8

Опис технологій та методів для розробки системи моніторингу

C++ – надає швидкодію та можливість керування ресурсами.

Qt – широкий набір візуальних елементів, забезпечує приємний та зрозумілий UI та UX.

Flow-based аналіз – швидкість та попередній аналіз трафіку

9

Продовження додатку Б

Опис технологій та методів для розробки системи моніторингу

Аналіз пакетів – покращення аналізу після Flow-based аналізу. Також для більш детальної інформації

Wireshark – сніфер для захоплення трафіку в реальному часі, а також для аналізу

PyTorch – для машинного навчання для покращення моніторингу та аналізу трафіку

10

Продовження додатку Б

Реалізована програма з моніторингу мережевого трафіку

Packets | Flows

	Source IP	Destination IP	Source Port	Destination Port	Protocol
1	192.168.0.105	216.58.215.67	55344	80	TCP
2	192.168.0.105	216.58.215.67	60534	80	TCP
3	216.58.215.67	192.168.0.105	80	55344	TCP
4	216.58.215.67	192.168.0.105	80	60534	TCP
5	192.168.0.105	172.217.16.3	60254	443	TCP
6	172.217.16.3	192.168.0.105	443	60254	TCP
7	192.168.0.105	172.217.16.3	60254	443	TCP
8	142.250.75.10	192.168.0.105	443	34800	TCP
9	192.168.0.105	142.250.75.10	34800	443	TCP
10	192.168.0.105	104.18.32.115	46022	443	TCP
11	104.18.32.115	192.168.0.105	443	46022	TCP
12	192.168.0.105	104.18.32.115	46022	443	TCP
13	192.168.0.105	162.159.61.4	42146	443	TCP
14	162.159.61.4	192.168.0.105	443	42146	TCP
15	192.168.0.105	162.159.61.4	42146	443	TCP
16	192.168.0.105	216.58.215.67	50088	80	TCP
17	192.168.0.105	2.16.172.51	56604	80	TCP
18	192.168.0.105	172.217.16.3	60254	443	TCP
19	172.217.16.3	192.168.0.105	443	60254	TCP
20	192.168.0.105	172.217.16.3	60254	443	TCP
21	192.168.0.105	162.159.61.4	42146	443	TCP
22	192.168.0.105	162.159.61.4	42146	443	TCP
23	192.168.0.105	172.217.16.3	60254	443	TCP
24	172.217.16.3	192.168.0.105	443	60254	TCP
25	172.217.16.3	192.168.0.105	443	60254	TCP
26	192.168.0.105	172.217.16.3	60254	443	TCP
27	172.217.16.3	192.168.0.105	443	60254	TCP
28	192.168.0.105	172.217.16.3	60254	443	TCP
29	172.217.16.3	192.168.0.105	443	60254	TCP
30	192.168.0.105	172.217.16.3	60254	443	TCP
31	172.217.16.3	192.168.0.105	443	60254	TCP

Status: Monitoring

Status: Monitoring

Status: Anomaly Detected!

Packets | Flows

	Source IP	Destination IP	Protocol	Packets	Bytes	Last Seen
1	104.18.32.115:443	192.168.0.105:46022	TCP	1	105	Fri May 31 17:32:27 2024...
2	109.86.65.71:36971	224.0.0.251:5353	UDP	4	344	Fri May 31 17:32:45 2024...
3	142.250.203.131:80	192.168.0.105:53728	TCP	2	134	Fri May 31 17:32:43 2024...
4	142.250.203.132:443	192.168.0.105:59758	TCP	1	105	Fri May 31 17:32:36 2024...
5	142.250.203.197:443	192.168.0.105:50134	TCP	2	173	Fri May 31 17:32:46 2024...
6	142.250.75.10:443	192.168.0.105:34800	TCP	1	151	Fri May 31 17:32:26 2024...
7	149.154.167.41:443	192.168.0.105:51224	TCP	7	856	Fri May 31 17:32:43 2024...
8	162.159.61.4:443	192.168.0.105:42146	TCP	7	1207	Fri May 31 17:32:43 2024...
9	172.217.16.2:443	192.168.0.105:44966	TCP	2	185	Fri May 31 17:32:42 2024...
10	172.217.16.3:443	192.168.0.105:60254	TCP	15	1317	Fri May 31 17:32:43 2024...
11	172.217.16.35:443	192.168.0.105:42476	TCP	4	395	Fri May 31 17:32:42 2024...
12	172.217.19.227:443	192.168.0.105:55136	TCP	1	105	Fri May 31 17:32:38 2024...
13	172.22.65.165:55765	224.0.0.251:5353	UDP	3	258	Fri May 31 17:32:42 2024...
14	178.150.28.131:39929	224.0.0.251:5353	UDP	2	172	Fri May 31 17:32:42 2024...
15	178.150.28.58:41491	224.0.0.251:5353	UDP	7	599	Fri May 31 17:32:42 2024...
16	178.150.28.86:33943	224.0.0.251:5353	UDP	10	860	Fri May 31 17:32:44 2024...
17	192.168.0.100:57621	192.168.0.255:57621	UDP	1	82	Fri May 31 17:32:35 2024...
18	192.168.0.100:5353	224.0.0.251:5353	UDP	1	86	Fri May 31 17:32:34 2024...
19	192.168.0.105:46022	104.18.32.115:443	TCP	2	171	Fri May 31 17:32:27 2024...
20	192.168.0.105:53728	142.250.203.131:80	TCP	2	132	Fri May 31 17:32:43 2024...
21	192.168.0.105:59758	142.250.203.132:443	TCP	2	171	Fri May 31 17:32:36 2024...
22	192.168.0.105:50134	142.250.203.197:443	TCP	2	171	Fri May 31 17:32:46 2024...
23	192.168.0.105:34800	142.250.75.10:443	TCP	1	66	Fri May 31 17:32:26 2024...
24	192.168.0.105:51224	149.154.167.41:443	TCP	7	1161	Fri May 31 17:32:43 2024...
25	192.168.0.105:42146	162.159.61.4:443	TCP	11	1229	Fri May 31 17:32:43 2024...
26	192.168.0.105:44966	172.217.16.2:443	TCP	3	276	Fri May 31 17:32:42 2024...
27	192.168.0.105:42476	172.217.16.35:443	TCP	7	618	Fri May 31 17:32:42 2024...
28	192.168.0.105:60254	172.217.16.3:443	TCP	18	1583	Fri May 31 17:32:43 2024...
29	192.168.0.105:55136	172.217.19.227:443	TCP	2	171	Fri May 31 17:32:38 2024...
30	192.168.0.105:54840	192.229.133.221:443	TCP	3	276	Fri May 31 17:32:42 2024...
31	192.168.0.105:56604	2.16.172.51:80	TCP	4	264	Fri May 31 17:32:33 2024...

Status: Monitoring

11

ВИСНОВКИ

В ході кваліфікаційної роботи була розроблена система моніторингу та аналізу мережевого трафіку, яка має великий потенціал у виявленні аномалій, оптимізації мережевої інфраструктури та підвищенні рівня кібербезпеки. Дана система поєднує у собі кілька ключових компонентів, що дозволяє проводити детальний аналіз мережевого трафіку у реальному часі та на рівні даних пакетів.

Однією з ключових особливостей розробленої системи є використання мови програмування C++ та фреймворку Qt для розробки гнучкого та продуктивного програмного забезпечення. Ці технології дозволяють ефективно працювати з мережевими даними та створювати потужні інструменти для моніторингу й аналізу мережі.

Для захоплення пакетів з мережі у реальному часі та подальшого аналізу використовується Wireshark, яка надає широкі можливості для роботи з мережевими протоколами та структурами даних. Це дозволяє системі ефективно захоплювати великий обсяг трафіку та аналізувати його у реальному часі.

ПРОДОВЖЕННЯ ВИСНОВКІВ

Крім того, для аналізу трафіку використовується метод flow-based аналізу, який дозволяє швидко оцінювати загальні тенденції та паттерни у мережі. Цей підхід доповнює аналіз на рівні пакетів та дозволяє системі отримувати узагальнену статистику про трафік.

Окремо варто відзначити використання бібліотеки PyTorch для машинного навчання, що дозволяє системі виявляти складні паттерни та аномалії в мережевому трафіку, що може бути важко виявити за допомогою традиційних методів аналізу.

У підсумку, розроблена система моніторингу та аналізу мережевого трафіку є потужним інструментом для забезпечення безпеки, ефективності та надійності мережі. Вона дозволяє оперативно виявляти та реагувати на проблеми у мережі, забезпечуючи надійну роботу інфраструктури та підвищуючи загальний рівень кібербезпеки.

Продовження додатку Б

РЕЗУЛЬТАТИ ВПРОВАДЖЕННЯ

1. Бученко І. А., Мальченко Г. С. Оптимізація продуктивності та ресурсоємності. «Проблеми комп'ютерної інженерії» : IV науково-практ. конф., м. Київ, 1 груд. 2023 р. С. 102–103. URL: https://duikt.edu.ua/uploads/p_2626_33497568.pdf.
2. Бученко І. А., Малчьенко Г. С. IDS алгоритми для виявлення вторгнень до мережі. «Сучасний стан та перспективи розвитку IOT» м. Київ, 1 груд. 2023 р. відправлено на друк. URL: