МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Спеціальність: 123 «Комп'ютерна інженерія»

Освітньо-професійна програма: «Обслуговування комп'ютерних систем і мереж»

Група: 4КС-58

Дипломний проект

здобувача освіти денної форми навчання
КС.58.05.000.ДП

ВИШНЕВСЬКОГО
АНДРІЯ ОЛЕКСАНДРОВИЧА

м. Одеса
2025 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Спеціальність: **123 «Комп'ютерна інженерія»**

Освітньо-професійна програма: «**Обслуговування комп'ютерних систем і мереж»**

Група: **4КС-58**

ПОЯСНЮВАЛЬНА ЗАПИСКА

до дипломного проекту на тему:

Розробка моделі оптимізації та безпеки передачі даних за допомогою механізму NBAR

Проектний матеріал складається з пояснювальної записки на 83 сторінках та графічного (презентаційного) матеріалу на 16 аркушах (слайдах)

Дипломник _____ (Вишневський А.О.)

Керівник _____ (Скорняков В.С.)

Консультанти:

з економічного розділу _____ (Канський М.Ю.)

з розділу охорони праці та техніки безпеки _____ (Чорновол Н.І.)

з нормоконтролю _____ (Петрашова В.І.)

старший консультант _____ (Кривченко Ю.В.)

До захисту допущений

Голова циклової комісії _____ (Кривченко Ю.В.)

Завідувач відділення _____ (Краснокутська К.Г.)

Захист «21» червня 2025 р. Протокол ЕК № 2

Оцінка ЕК 5 (відмінно) / 90 б.

Секретар ЕК _____

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Відділення *комп'ютерних систем* Комісія *КТ та ПІ*

Спеціальність *123 «Комп'ютерна інженерія»*

Освітньо-професійна програма *«Обслуговування комп'ютерних систем і мереж»*

ЗАТВЕРДЖУЮ:

Заст. дир. з НВР _____

Беркань І.В.

"19" *08* 2025 р.

ЗАВДАННЯ

на дипломний проект

Здобувачеві (здобувачці) освіти *Вишневському Андрію Олександровичу*

(прізвище, ім'я, по батькові)

1. Тема проекту *Розробка моделі оптимізації та безпеки передачі даних за допомогою механізму NBAR*

затверджена наказом по коледжу від "14" *листопада* 2024 р. № *246*

2. Термін здачі закінченого проекту *16.06.25р.*

3. Вихідні данні до проекту (роботи) *1. Використовувати тестові маршрутизатори та міжмережеві екрани у якості мережевих пристроїв; 2. Передбачити зчитування файлу конфігурації, формувати SNMP-запити, формувати звіти та аналізувати їх, формувати політики обмеження, надсилати політики обмеження до пристрою за допомогою SSH, очищувати конфігураційний файл пристрою при аварійному завершенні; 3. Програмне забезпечення реалізовувати мовою Java у середовищі розробки Intellij Idea*

4. Зміст розрахунково-пояснювальної записки (перелік питань, які необхідно розробити)

Огляд технологій аналізу мережевих потоків; Розробка алгоритмів оптимізації та безпеки передачі даних за допомогою механізму NBAR; Програмна реалізація алгоритмів оптимізації та безпеки передачі даних за допомогою механізму NBAR; Тестування моделі; Економічний розділ; Розділ охорони праці і техніки безпеки

5. Перелік графічного (презентаційного) матеріалу (з точним зазначенням обов'язкових креслень, кількості слайдів)

Блок-схема алгоритму оптимізації мережевого трафіку; Блок-схема алгоритму читання таблиці портів маршрутизатору; Блок-схема алгоритму моніторингу навантаження каналу зв'язку; Блок-схема алгоритму скидання файлу конфігурації пристрою; Структурна схема системи оптимізації та безпеки передачі даних; Функціональна схема діаграми класів у програмі; БСА розпізнавання потоку даних та управління політиками шейпінгу; Структура мережі для тестування ПЗ; Визначення навантаження на канал зв'язку в мережі

6. Консультанти по проекту (роботі), із зазначенням розділів проекту, що їх стосується

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
Основний розділ	*Скорняков В.С.*		
Економічний розділ	*Канський М.Ю.*		
Розділ охорони праці	*Чорновол Н.І.*		
Нормоконтроль	*Петрашова В.І.*		
Старший консультант	*Кривченко Ю.В.*		

7. Дата видачі завдання 15.05.25 р.

Керівник *Скорняков В.С.* _____
(підпис)

Завдання прийняв до виконання _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/р	Назва етапів дипломного проекту (роботи)	Термін виконання етапів дипломного проекту (роботи)	Відмітка про виконання
1.	*Постановка задачі проектування*	15.05.25	виконав
2.	*Аналіз моделей розпізнавання мережевих потоків даних*	16.05.25	виконав
3.	*Аналіз засобів оптимізації та безпеки передачі даних*	17.05.25	виконав
4.	*Розробка структури алгоритму оптимізації трафіку*	18.05.25	виконав
5.	*Розробка алгоритму зчитування таблиці портів*	22.05.25	виконав
6.	*Розробка алгоритму моніторингу навантаження на канал*	26.05.25	виконав
7.	*Розробка алгоритму скидання файлу конфігурації*	01.06.25	виконав
8.	*Визначення програмних засобів розробки*	06.06.25	виконав
9.	*Розробка об'єктно-орієнтованої моделі програми*	10.06.25	виконав
10.	*Реалізація інтерфейсу програми розпізнавання потоку*	11.06.25	виконав
11.	*Тестування моделі розпізнавання потоку даних*	12.06.25	виконав
12.	*Аналіз результатів моделювання та ефективності*	13.06.25	виконав
	розробленого алгоритму та моделі		
13.	*Виконання графічної частини проекту*	13.06.25	виконав
14.	*Виконання економічних розрахунків*	14.06.25	виконав
15.	*Розробка заходів з охорони праці*	16.06.25	виконав

Дипломник _____
(підпис)

Керівник _____
(підпис)

Формат	Зона	Поз.	Позначення	Назва	Кіл.	Примітка
				Документація		
			КС 58. 05 000. 00 ДП	Дипломний проект		
А4			КС 58. 05 000. 00 ДП ПЗ	Пояснювальна записка	1	

Зм.	Арк.	№ докум.	Підпис	Дата	КС 58. 05 000. 00 ДП		
Розробив		Вишневський А.О.		16.06	Розробка моделі оптимізації та безпеки передачі даних за допомогою механізму NBAR	Літ.	Аркуш 4 / Аркушів 83
Перевірив		Скорняков В.С.		16.06		Н Д П	
Н. Контр.		Петрашова В.І.		16.06		ВСП "ОТФК ОНТУ" гр.4КС-58	
Затверд.		Кривченко Ю.В.		16.06			

ЗМІСТ

ВСТУП

Сучасні комп'ютерні мережі характеризуються складною структурою та значним зростанням обсягу переданих даних. Це створює додаткові виклики для забезпечення ефективного керування трафіком і гарантування його безпеки. Одним з ключових питань у сфері обслуговування комп'ютерних мереж є оптимізація пропускної здатності каналів зв'язку та контроль за передачею даних різних типів, що є особливо актуальним для мереж із обмеженими ресурсами або в умовах віддалених об'єктів, де єдиними доступними технологіями передачі є мобільний або супутниковий зв'язок. У таких випадках нерегульоване використання мережевих ресурсів може призвести до перевантажень та втрат критично важливої інформації.

Сьогодні одним із поширених рішень для забезпечення належного рівня якості обслуговування (QoS) є впровадження технологій управління трафіком, таких як QoS (Quality of Service). Однак ця технологія часто не забезпечує достатньої гнучкості та не може ефективно вирішувати задачі маршрутизації в реальному часі. Це стає особливо проблематичним на об'єктах з низькою пропускною здатністю каналів зв'язку, де відсутність можливості передати важливі дані може призвести до зупинки критично важливих процесів. У таких ситуаціях виникає потреба в автоматичному розпізнаванні та класифікації типів трафіку з метою динамічного керування його пріоритетами. Одним з ефективних підходів для вирішення цих проблем є застосування технології Network-Based Application Recognition (NBAR), яка дозволяє ідентифікувати типи трафіку на рівні додатків та застосовувати політики обмеження або пріоритизації в залежності від важливості даних. Це дозволяє автоматизувати процес управління мережею, зменшуючи навантаження на адміністратора і знижуючи ризик людських помилок при налаштуванні обладнання.

У рамках даного дипломного проекту передбачається розробка програмного забезпечення, яке працюватиме в автоматичному режимі і дозволятиме відслідковувати навантаження на мережу, класифікувати трафік за допомогою NBAR та застосовувати політики обмеження для неважливих типів трафіку.

1 ОСНОВНИЙ РОЗДІЛ

1.1 Огляд технологій аналізу мережевих потоків

Модель оптимізації та безпеки передачі даних має відповідати актуальним вимогам до мережевих рішень і спрямована на поліпшення якості обслуговування (QoS), підвищення надійності систем та забезпечення стабільної роботи мереж в умовах підвищеного навантаження.

Потоки даних у цифровій мережі мають неоднорідний характер і включають інформаційні обміни від різних програмних продуктів. Кожна з цих систем висуває специфічні вимоги до умов функціонування мережі. Якщо ці умови не будуть дотримані, це може призвести до погіршення якості їх роботи та зниження зручності користування. Тому важливо забезпечити ефективний інструмент для розпізнавання різних типів інформаційних потоків і автоматичного застосування належних правил керування, що сприятиме підтриманню оптимальних параметрів передачі даних, особливо у випадках з обмеженими мережевими можливостями.

1.1.1 Загальні принципи класифікації мережевого трафіку

На локальному рівні з високою пропускною здатністю мережі зазвичай легко відповідати технічним вимогам, однак у глобальних мережах, обмежених пропускною здатністю, ця задача значно ускладнюється. Отже, в глобальних мережах критично важливим є управління трафіком, яке дозволяє розставляти пріоритети між різними додатками в межах наявної пропускної смуги та забезпечувати дотримання їхніх вимог. Окрім цього, розуміння типів додатків і протоколів у мережевих потоках має важливе значення для забезпечення відповідної безпеки. Існує широкий спектр методів і підходів для класифікації мережевого трафіку. Їх використання дозволяє визначити додатки та протоколи, які циркулюють у мережі.

Класифікований трафік стає основою для виконання таких завдань, як моніторинг, виявлення, управління й оптимізація, що в кінцевому результаті сприяє підвищенню ефективності роботи мережі. Сучасні технології класифікації базуються на двох головних підходах: аналіз корисного навантаження пакетів і

статистичний аналіз. У першому випадку класифікація здійснюється шляхом аналізу конкретних параметрів корисного навантаження, таких як використання портів рівня 4 (для джерел чи пунктів призначення). У другому – визначаються характеристики трафіку, наприклад, затримка між пакетами, тривалість сеансу тощо.

Найчастіше у застосуванні знаходиться метод аналізу корисного навантаження, який може бути як базовим, так і розширеним. У базовому варіанті аналіз здійснюється лише на основі інформації з IP-заголовків, таких як IP-адреси, MAC-адреси чи протоколи. Хоча цей метод простий у використанні, він має обмежену функціональність і не здатен забезпечити точну класифікацію багатьох сучасних додатків. Інші методи, такі як аналіз вхідного інтерфейсу, застосовуються рідше через свої обмеження. Загалом усі базові підходи до класифікації, засновані на даних IP-адрес чи портів рівня 4, демонструють свою недостатність, оскільки сучасні програми не завжди дотримуються стандартних портів.

Передові технології класифікації, як-от глибокий аналіз пакетів (DPI), виявляються значно ефективнішими. DPI використовує алгоритми аналізу шаблонів, поведінки чи навіть статистики. Такі методи дозволяють ідентифікувати трафік з набагато вищою точністю. Залежно від підходу класифікація може базуватися на окремих пакетах, потоках або навіть на повідомленнях. Наприклад, технологія PBFS орієнтується на аналіз потоків, визначаючи їх за ключовими параметрами, такими як IP-адреса, порт призначення чи транспортний протокол. В інших підходах, як-от MBFS, аналізуються повідомлення, що дозволяє враховувати контекст протоколу. Дослідження показали, що методи глибокої перевірки, зокрема аналіз підписів або числових характеристик, дозволяють більш ефективно ідентифікувати специфічні програми. Підпис є унікальним для кожного додатка і базується на його характерних ознаках. Однак, щоб залишатися актуальним, цей підхід вимагає регулярного оновлення бази даних. Аналіз підписів можна здійснювати різними способами, як-от через аналіз шаблонів, поведінковий чи статистичний аналіз. Це дозволяє класифікувати програми навіть у тих випадках, коли стандартні шаблони корисного навантаження відсутні.

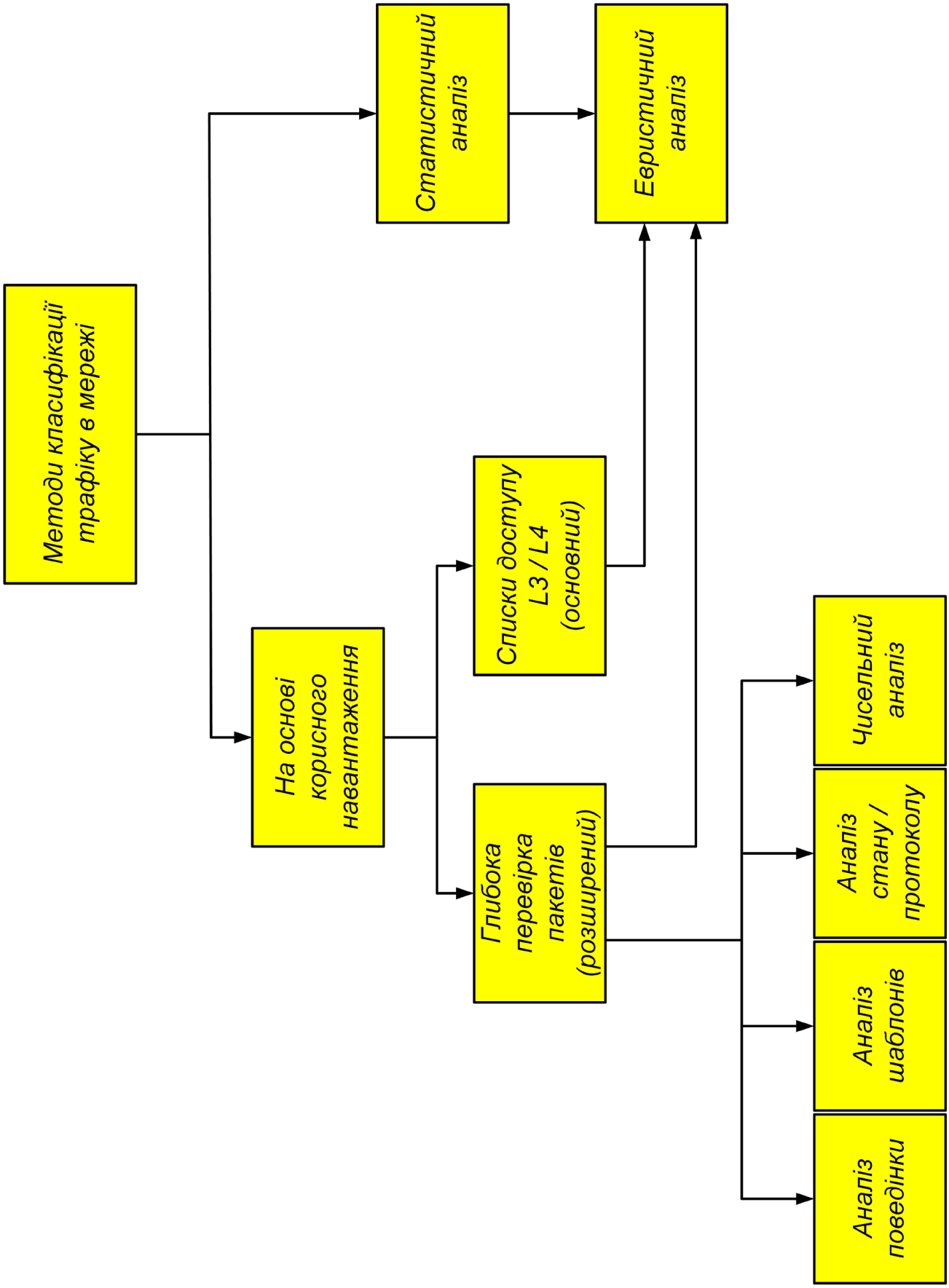


Рисунок 1.1. Підходи до класифікації трафіку комп'ютерної мережі

Зрештою, із зростанням використання зашифрованого трафіку ефективність класичних методів класифікації знижується. Однак передові підходи, які інтегрують алгоритми штучного інтелекту, наприклад, кластеризацію, дозволяють вирішити цю проблему. У типовому середовищі реалізації різні методи класифікації використовуються в комбінації, оскільки жоден з них не здатен забезпечити універсальне рішення для аналізу мережевого трафіку.

У сучасному аналізі мережевого трафіку застосовується багато методів і підходів (рис. 1.1). Виявлення особливостей трафіку дозволяє визначати програми і протоколи, які функціонують у мережі.

Класифікований трафік стає основою для таких завдань, як нагляд, діагностика, регулювання та вдосконалення, що сприяє підвищенню ефективності роботи мережевої інфраструктури.

Для аналізу мережевого трафіку зазвичай застосовуються два основні підходи:

- класифікація пакетів на основі змісту корисного навантаження. У цьому випадку для класифікації використовується аналіз полів корисного навантаження, наприклад портів четвертого рівня (початкового або кінцевого пункту);

- класифікація на базі статистичних параметрів, яка заснована на дослідженні характеристик трафіку, таких як затримка між пакетами, тривалість сеансу тощо.

Найбільш поширеним способом є аналіз корисного навантаження, який може бути як базовим, так і поглибленим. У простішому варіанті оцінюється лише інформація з заголовка IP, до якої належить:

- IP-адреса (рівень 3);

- MAC-адреса (рівень 2);

- специфікація протоколів.

Цей підхід простий і швидкий у використанні, але його головним недоліком є нездатність забезпечити повну класифікацію додатків. Менш популярним залишається метод, що базується на місці розташування трафіку (вхідному інтерфейсі), через його обмежену ефективність.

Усі базові методи, які застосовують інформацію про IP-адреси або протоколи, мають обмежений функціонал, оскільки покладаються лише на аналіз заголовка IP.

Схожі проблеми мають методи, що засновані на аналізі портів четвертого рівня, оскільки чимало сучасних додатків не використовують стандартні порти.

Одним із провідних напрямків залишаються методи класифікації на основі детального аналізу пакетів (DPI), які вважаються більш надійними порівняно з базовими техніками. DPI застосовує такі підходи, як дослідження шаблонів або поведінковий аналіз. Завдяки цим методам зростає точність ідентифікації додатків і протоколів, що активно працюють у мережі.

Існуючі підходи до класифікації мережевого трафіку, які базуються на аналізі корисного навантаження, можуть бути розподілені залежно від застосовуваного способу обробки даних. Усі ці методи, незалежно від їх типу, використовують один або кілька варіантів оцінки корисного навантаження, таких як технологія Deep Packet Inspection для дослідження і сегментації трафіку. Найпростішим методом є PBNS (Packet-Based Non-State), який передбачає перевірку корисного навантаження на певні характеристики, зокрема, такі як номери портів. Цей спосіб забезпечує мінімальне навантаження на процесор. PBNS зазвичай базується на базових принципах класифікації за корисним навантаженням. Утім, він не може гарантувати високої точності, оскільки орієнтується лише на одиничні пакети, не враховуючи параметри сесії додатка, і має обмеження щодо рівня глибини перевірки потоку пакета.

У методі PBFS (Packet-Based Flow State), орієнтованому на потоки, кожен потік визначається як послідовність пакетів, що передаються від програми-відправника до додатка-отримувача. Для кожного потоку формується таблиця, яка дозволяє фіксувати параметри окремої сесії, що включає п'ять основних елементів: початкову адресу, адресу призначення, порт виходу, порт отримувача та транспортний протокол. Завдяки цій методиці, після ідентифікації початкового пакета як частини певного додатка, усі наступні пакети в межах цього потоку автоматично позначаються відповідними параметрами. Наприклад, під час використання сервісів VoIP, протокол H.323 використовується на етапі налаштування виклику, після чого RTP/RTCP відповідає за передачу голосового трафіку. Після ідентифікації потоку H.323 усі наступні RTP/RTCP-пакети, що

мають ту саму пару IP-адрес джерела та призначення, автоматично класифікуються відповідним чином.

Метод MBFS (Message-Based Flow State), що базується на аналізі повідомлень, схожий на PBFS, але враховує повідомлення замість одиничних пакетів. Повідомлення є інформаційним компонентом, специфічним для протоколу, і може охоплювати кілька пакетів чи, навпаки, бути інтегрованим у рамках одного пакета. Для роботи з такими повідомленнями потрібен нормалізатор TCP, який забезпечує коректну обробку IP-фрагментів і TCP-сегментів. Проте цей метод має високі вимоги до пам'яті через необхідність повного аналізу кожного повідомлення.

Метод MBPS (Message-Based Protocol State) не лише дозволяє фіксувати стан додатка, але й забезпечує аналіз передачі між програмами. Для впровадження цього підходу потрібно повне знання внутрішньої логіки протоколу. Такий спосіб потребує значних ресурсів процесора та має підвищені вимоги до обсягу пам'яті.

Методи PBFS, MBFS і MBPS, які базуються на принципах глибокого аналізу пакетів (DPI), є високоточними інструментами для сегментації трафіку.

Визначення більшості поширених програм зазвичай можливе на основі інформації L3 та L4, але для деяких випадків цього недостатньо. Сучасні мережі вимагають детальнішого аналізу для ідентифікації підкласифікацій у програмах, як-от окремі URL-адреси чи певні типи повідомлень. Щоб досягти цього рівня точності, необхідне впровадження глибокого аналізу пакетів (DPI). Цей метод дозволяє розпізнавати програми на основі аналізу характерних особливостей, які ідентифікують кожен додаток.

DPI здебільшого спирається на аналіз підписів. Підписи є своєрідними унікальними візерунками, які пов’язані з окремими програмами. Для кожного додатка створюється база даних, що включає його унікальні характеристики. Потім класифікаційний механізм порівнює мережевий трафік із цими даними, точно визначаючи, якому додатку він належить. Однак такі бази даних потребують регулярного оновлення, щоб враховувати нові програми та зміни у вже існуючих протоколах.

Існують різні методи аналізу підписів, серед яких:

1. Розпізнавання шаблонів, що використовує унікальні структури в корисному навантаженні.

2. Числовий аналіз, заснований на вивченні характеристик пакета, таких як розміри чи затримки.

3. Поведінковий аналіз, який досліджує дії трафіку в реальному часі.

4. Статистичний аналіз даних для виявлення закономірностей.

5. Дослідження станів протоколу для оцінки алгоритмів передачі даних.

Деякі додатки інтегрують у свої пакети особливі шаблони, наприклад, символи чи рядки, що дозволяє їх ідентифікувати. Але не всі протоколи додають такі унікальні маркери. Це створює виклики для механізмів DPI, особливо коли структура пакета ускладнена.

Іншим перспективним підходом є числовий аналіз параметрів. Наприклад, у типовому сценарії запит клієнта може містити повідомлення обсягом 18 байтів, тоді як відповідь сервера — 11 байтів. Аналіз цих показників може потребувати більше часу, адже включає кілька пакетів.

Крім того, поведінковий аналіз дозволяє зрозуміти особливості програмного забезпечення через вивчення динаміки взаємодії між джерелом і отримувачем. У поєднанні зі статистичним підходом цей метод застосовується для визначення основних характеристик протоколів. Його часто використовують антивірусні системи для виявлення шкідливих програм.

Оскільки більшість сучасних додатків шифрує трафік, традиційні механізми класифікації втрачають ефективність. Шифрування робить невидимими інформацію верхніх рівнів протоколів. Проте поведінковий і статистичний аналіз можуть частково компенсувати цю проблему. Використання алгоритмів штучного інтелекту, таких як кластеризація, сприяє ідентифікації зашифрованого трафіку.

У типовій реалізації ці підходи зазвичай комбінуються, адже жоден із них не забезпечує достатньої ефективності в ізольованому вигляді. Їхнє інтегроване застосування дозволяє розширити спектр можливостей класифікації для покращення безпеки та продуктивності мереж.

1.1.2 Класифікації мережевих потоків даних засобами маршрутизаторів

Класифікація мережевих потоків даних є невід'ємною складовою сучасного управління трафіком. Вона дозволяє не лише оптимізувати використання пропускної здатності мережі, а й підвищити її безпеку та ефективність. Завдяки сучасним технологіям маршрутизатори надають широкий спектр засобів для аналізу та диференціації потоків даних.

Одним із найпоширеніших підходів до класифікації є використання списків контролю доступу (ACL). Вони працюють на рівнях L3 та L4 і базуються на параметрах, таких як IP-адреси джерела та призначення, номери портів, а також протоколи. Використання таких списків дозволяє створювати правила для відокремлення трафіку різних категорій. Ці правила можуть застосовуватись як у програмних, так і в апаратних реалізаціях маршрутизаторів, забезпечуючи швидке узгодження параметрів та подальшу обробку.

Окремо варто зазначити впровадження глибокого аналізу пакетів (DPI). Ця технологія є наступним етапом класифікації, оскільки вона дозволяє аналізувати не тільки заголовки пакетів, а й вміст корисного навантаження. У результаті DPI надає змогу ідентифікувати конкретні програми та протоколи, навіть якщо вони маскують свої параметри через ефемерні порти або інші хитрощі. Глибокий аналіз здійснюється за допомогою різноманітних підходів, таких як протокольний аналіз, поведінковий підхід, шаблонний пошук і евристичне тестування.

DPI може бути реалізоване у двох формах: програмній або апаратній. Програмні модулі класифікації є економічно вигідними і підходять для середовищ із низькою або середньою інтенсивністю трафіку. Вони забезпечують базовий рівень продуктивності, але для їхньої роботи потрібні суттєві обчислювальні ресурси. Апаратні рішення, у свою чергу, дозволяють досягти більшої швидкості та точності класифікації, оскільки обробка виконується спеціалізованими процесорами. Це ідеальний варіант для великих об’ємів трафіку, характерних для центрів обробки даних і масштабних корпоративних мереж.

Ще одним ефективним засобом класифікації є механізми розпізнавання додатків на основі аналізу станів. Ця функція передбачає аналіз пакетів із глибоким

зануренням у їхню структуру для визначення специфічних параметрів, таких як унікальні ідентифікатори протоколу. Наприклад, класифікація HTTP-пакетів може здійснюватися на рівні окремих URL-адрес, а пакети інших протоколів — за їхніми внутрішніми характеристиками. Це забезпечує можливість адаптивного управління трафіком відповідно до його пріоритету або типу.

Розширення функціоналу класифікації також досягається за допомогою інтеграції технологій Quality of Service (QoS). QoS дозволяє маршрутизаторам забезпечувати диференційовану обробку трафіку, пріоритетизуючи важливі програми й обмежуючи менш критичні потоки. У поєднанні з методами глибокої перевірки пакетів (DPI) QoS стає потужним інструментом для управління ресурсами мережі.

Однак впровадження таких технологій не обходиться без викликів. Одним із найбільших обмежень є обсяг доступної пам’яті, яка використовується для обробки правил і шаблонів. Пам'ять у маршрутизаторах часто має фіксований розмір, і її ресурси можуть швидко вичерпуватись без ретельного планування. Ще одна проблема полягає у значному навантаженні на процесор при обробці великого обсягу трафіку, особливо у разі використання програмних рішень DPI.

Сучасні маршрутизатори також враховують необхідність забезпечення масштабованості та швидкості роботи. Інтеграція спеціалізованих обчислювальних модулів, підтримка апаратних механізмів пошуку і обробки, а також адаптація до нових протоколів і форматів даних дозволяють вирішувати ці завдання ефективно.

1.1.3 Аналіз методів позначення кадрів на різних рівнях

Процес позначення пакетів є ключовим етапом класифікації мережевого трафіку. Він забезпечує можливість застосування політик обслуговування для пріоритезації, обмеження або диференційованої доставки даних у мережах. Після того як потоки й пакети були ідентифіковані, вони повинні бути марковані, щоб надати маршрутизаторам і комутаторам можливість обробляти їх відповідно до визначених правил і політик.

Механізм маркування може бути реалізований на різних рівнях моделі OSI залежно від технічних вимог:

1. На третьому рівні (L3): Найбільш розповсюдженим методом є маркування на L3. Для цього використовуються поля заголовка IP, такі як:

– Тип послуги (ToS) — сигналізує про необхідність певних характеристик обслуговування (низька затримка, висока надійність або пропускна здатність).

– Точка диференційованого обслуговування (DSCP) — розширює можливості ToS завдяки використанню більшої кількості бітів, що дозволяє точніше налаштовувати політики служби.

2. На другому рівні (L2): Популярні L2-технології, такі як Ethernet, Frame Relay та ATM, також пропонують функціонал для маркування пакетів:

– ATM (Asynchronous Transfer Mode): Використовує біт пріоритету втрати комірки (Cell Loss Priority, CLP). Якщо цей біт встановлений у "1", то комірка може бути скинута у разі перевантаження, звільняючи ресурси для критичного трафіку.

– Frame Relay: Має біт Discard Eligible (DE), який дозволяє ідентифікувати некритичний трафік, що може бути скинутий для полегшення перевантаження.

– Ethernet за стандартом IEEE 802.1p: Використовує три біти в заголовку кадру для класифікації трафіку на вісім окремих класів обслуговування, які можуть включати звук, відео, контроль мережі та інше.

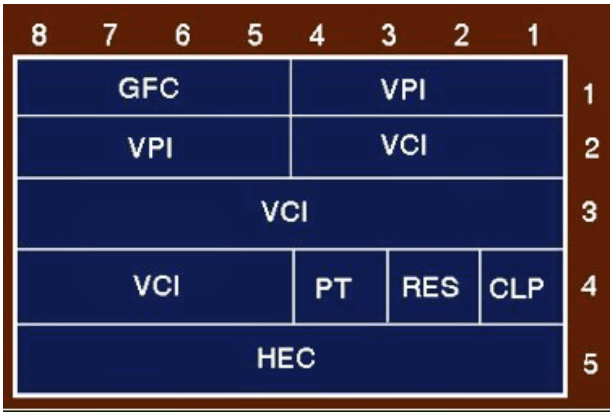


Рисунок 1.2. Формат кадру ATM та розташування біта Cell Loss Priority

ATM-комірка (рис.1.2) складається з заголовка розміром 5 байтів і корисного навантаження на 48 байтів. У заголовку знаходиться біт CLP (Cell Loss Priority), що відповідає за позначення пріоритету втрати. Комірки із встановленим значенням CLP у "1" мають низький пріоритет і можуть бути видалені під час перевантаження, що гарантує збереження ресурсів для важливих потоків.

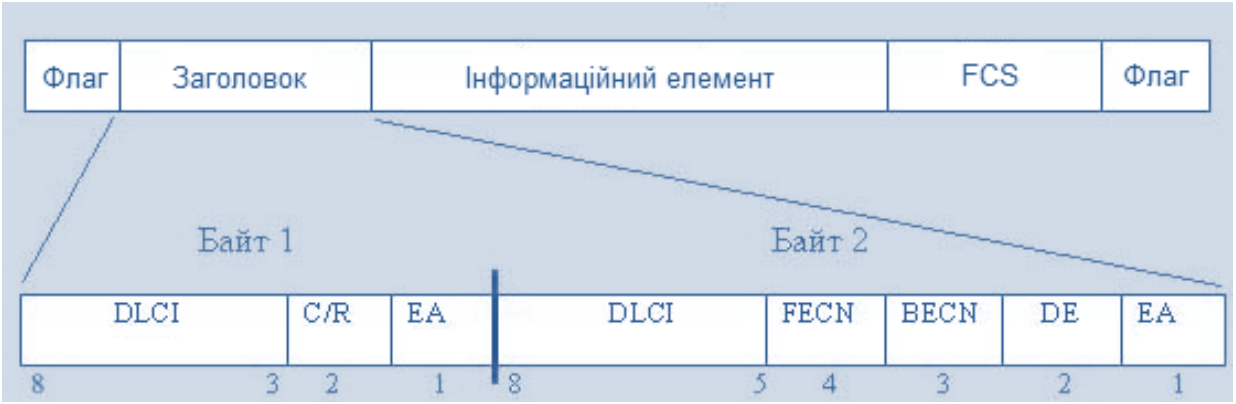


Рисунок 1.3. Формат пакету Frame Relay

У заголовку Frame Relay (рис.1.3) використовується біт Discard Eligible (DE), який, подібно до CLP, позначає некритичні кадри. Це дозволяє адаптувати передачу даних до умов перевантаження, забезпечуючи стійкість роботи мережі.

Ethernet-кадр стандарту IEEE 802.1q (рис.1.4) має додатковий TAG-заголовок, у якому визначаються три біти для класифікації трафіку. Вони дозволяють комутаторам другого рівня визначати пріоритети трафіку та забезпечувати класифікацію потоку у вісім класів. Наприклад, звук може бути позначений як клас із високим пріоритетом, тоді як стандартна передача файлів отримає нижчий клас.

Розподіл типів трафіку в стандарті IEEE 802.1p (рис.1.5) ґрунтується на важливості передачі даних. Нижче наведено таблицю з відповідними класами обслуговування та пріоритетами (табл.1.1).

Таблиця 1.1. Визначення типів і пріоритету трафіку у стандарті 802.1p

Тип трафіку	*Клас обслуговування*	*Пріоритет*
Банківські транзакції, ігри	Фон	1
Звуковий трафік	Звук	2
Відеопотоки	Відео	3
Важливі програми	Контрольований	4
Ключові користувачі	Пріоритетний	5
Стандартна локальна мережа	Негарантована доставка	6
Мережевий контроль	Мережевий контроль	7

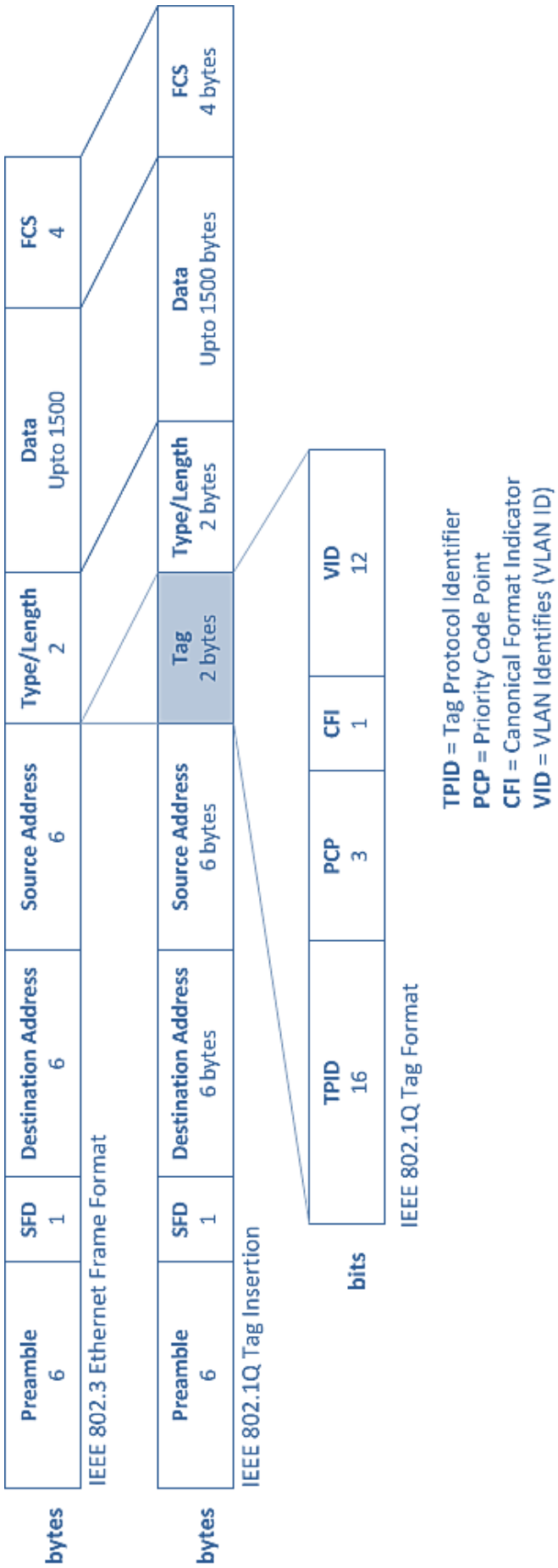


Рисунок 1.4. Формат кадру Ethernet 802.1q

Рисунок 1.5. Формат IP-заголовку за стандартом 802.1p

Міжнародні стандарти RFC 2474 і RFC 2475 визначають класифікацію на основі DSCP для впровадження диференційованого обслуговування (DiffServ). DSCP підтримує більшу кількість рівнів пріоритетів порівняно з ToS завдяки додатковим бітам у заголовку. Наприклад:

- EF PHB (Expedited Forwarding) забезпечує низькі затримки, мінімальний джиттер і високий рівень пропускної здатності, ідеально підходить для голосових сервісів.
- AF PHB (Assured Forwarding) пропонує адаптивну передачу даних із гарантіями залежно від класу імовірності втрати.

Класифікація бітів пріоритету в IP-заголовках описує різні рівні обслуговування:

Таблиця 1.2. Класифікація бітів пріоритету

Двійковий код	*Десятковий код*	*Класифікація*
000	0	Режим
001	1	Пріоритет
010	2	Негайний
011	3	Спалах
100	4	Відхилення спалаху
101	5	Критичний
110	6	Міжмережевий контроль
111	7	Мережевий контроль

Маркування пакетів є ключовою складовою ефективного управління мережею. Використання міток на рівнях L2 та L3 забезпечує можливість налаштування політик обслуговування відповідно до конкретних вимог. Завдяки впровадженню стандартів, таких як ToS, DSCP, IEEE 802.1p та RFC, мережеві адміністратори можуть оптимізувати продуктивність мережі, зберігаючи її надійність у різних умовах.

1.1.4 Аналіз засобів моніторингу потоків даних у мережі

Сучасна мережа – це складна екосистема, де безперервний моніторинг трафіку є запорукою стабільності, ефективності та безпеки. Грунтовний аналіз мережевого трафіку дозволяє виявляти аномалії, прогнозувати проблеми та реагувати на них у реальному часі. Для цього використовуються різні протоколи моніторингу, зокрема SFlow, NetFlow (та його еволюція IPFIX) і SNMP. Кожна з технологій має свої особливості, не тільки в методах збору даних, але й у механізмах їх передачі, обробці та аналізі.

SFlow є технологією, розробленою з оптимізацією для багатопротокольних середовищ. Основною рисою SFlow є використання вибіркового збору даних, що дозволяє обробляти навіть великі об’єми трафіку без значного навантаження на системні ресурси. Основні характеристики і реалізації:

- Вибіркова вибірка (Sampling): SFlow використовує статистичні методи для вибіркового збору пакетів. Наприклад, у високошвидкісних мережах може застосовуватись коефіцієнт вибірки 1:1000 або більше, що дозволяє зменшити обсяг оброблюваних даних без втрати представницької інформації;

- Незалежність від протоколу: Завдяки тому, що SFlow підтримує аналіз з рівня 2 до рівня 7 моделі OSI, вона може здійснювати моніторинг не лише IP-трафіку, а й даних, що передаються за допомогою різних протоколів (наприклад, Ethernet, MPLS, VoIP тощо);

- Сучасні реалізації: Найбільш поширеною є версія SFlow v5, яка інтегрується у мережеве обладнання різних виробників. Віртуалізовані середовища, такі як платформи для хмарних обчислень (VMware NSX, OpenStack) активно використовують SFlow для видимості трафіку між віртуальними машинами;

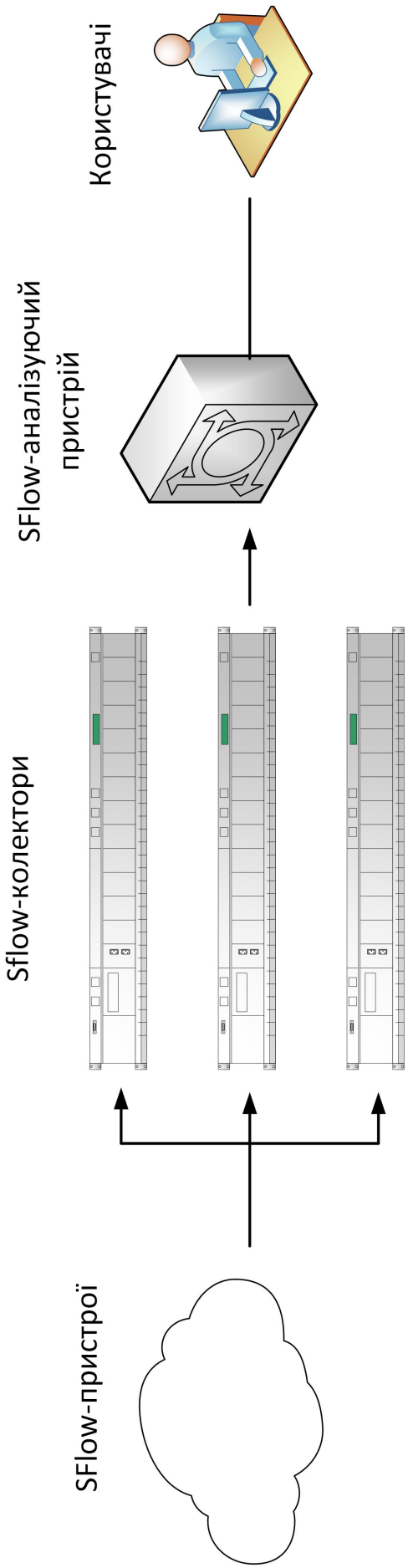


Рисунок 1.6. Схема роботи технології SFlow у комп’ютерній мережі

- Апаратна підтримка: Інтегровані апаратні чіпи, що реалізують SFlow, дозволяють практично в режимі реального часу здійснювати вибірковий збір даних без додаткового навантаження на основний процесор пристроїв.

На рис.1.6 ілюструється, як SFlow експортує вибіркові зразки трафіку з різних портів мережевого пристрою до централізованого колектора. Схема демонструє інтеграцію апаратного чіпа, який знімає з пристрою лише репрезентативну частину даних, що дозволяє зберігати баланс між точністю статистики та обчислювальними витратами.

Технологія NetFlow та IPFIX. NetFlow – це технологія моніторингу, орієнтована на збір інформації про потоки IP-трафіку. Вона надає детальну інформацію про кожен потік (flow), що включає IP-адреси джерела і призначення, номери портів, використовуваний протокол, обсяги даних, час початку і завершення потоку. Еволюція NetFlow:

- NetFlow v9: Ця версія є шаблонною, що дозволяє користувачеві визначати, які саме поля будуть збиратися для кожного потоку. Такий підхід дозволяє адаптувати моніторинг до нових протоколів і типів трафіку, забезпечуючи високу гнучкість;

- IPFIX (Internet Protocol Flow Information Export): Стандарт, розроблений як наступник NetFlow v9, який затверджено IETF. IPFIX підтримує ще більшу варіативність полів і розширені можливості аналізу, що дозволяє накопичувати більш точну та структуровану інформацію про мережевий трафік.

Розширене налаштування (Flexible NetFlow) дозволяє користувачам визначати власні шаблони даних, що збираються, і адаптувати збір статистики до конкретних вимог мережі.

NetFlow дозволяє отримувати інформацію не лише про загальні характеристики трафіку, але й аналізувати споживання пропускної здатності по конкретних потоках, що критично для виявлення аномалій та управління ресурсами. Завдяки апаратній реалізації збору даних, вплив NetFlow на основні процесорні ресурси пристроїв є мінімальним, що робить його придатним для використання навіть у великих мережах з високою навантаженістю.

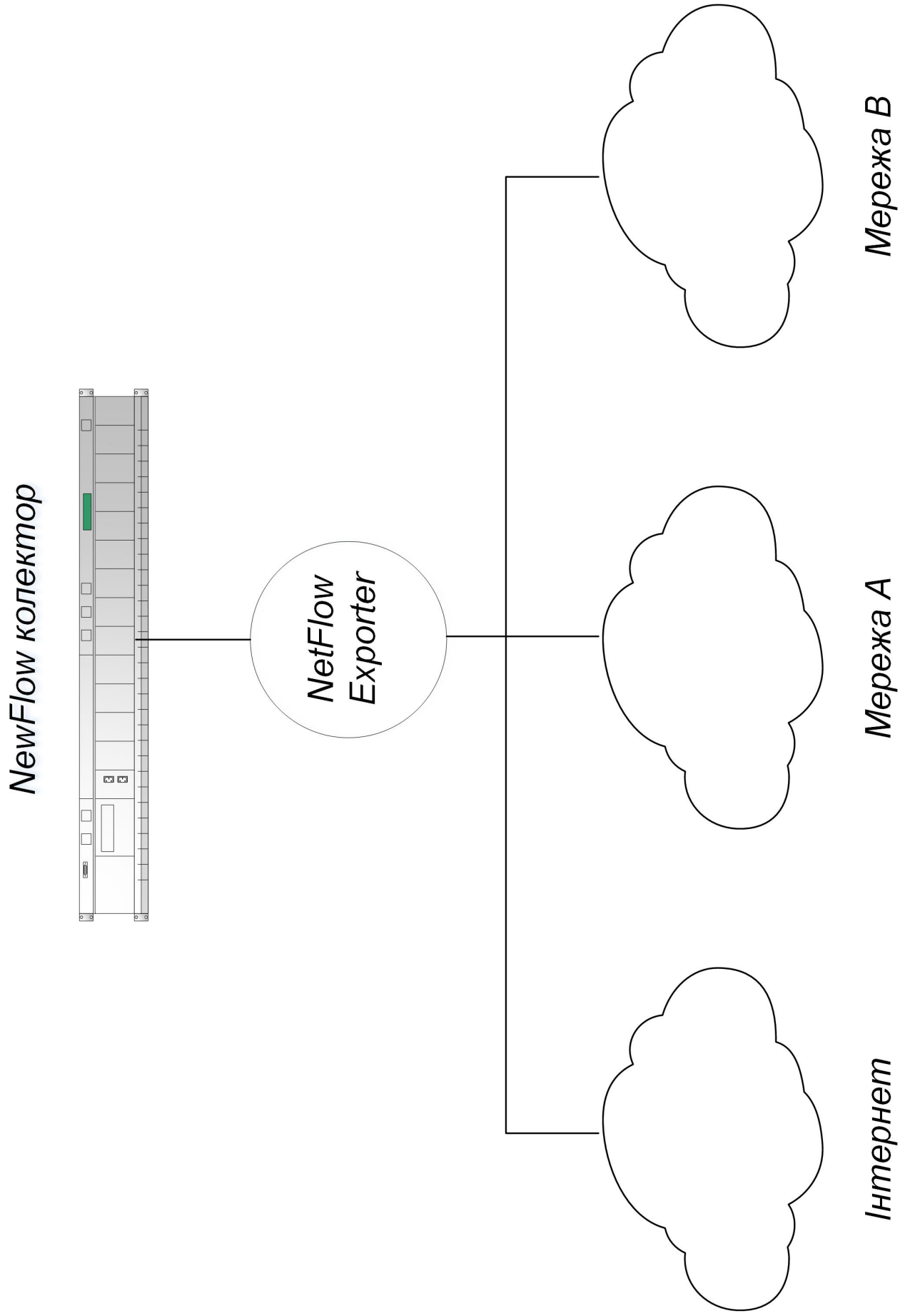


Рисунок 1.7. Схема роботи технології NetFlow у комп’ютерній мережі

На рис.1.7 представлено схему експорту потокових даних з мережевих пристроїв, які збираються за допомогою NetFlow або його еквіваленту IPFIX. Рисунок демонструє, як зібрана інформація передається до спеціалізованого колектора, який згодом дозволяє здійснити детальний аналіз трафіку та визначити точки перевантаження чи аномалії.

Технологія SNMP (Simple Network Management Protocol) займає іншу нішу в моніторингу мереж – це протокол, який в основному використовується для збору статистичних даних про стан пристроїв (наприклад, завантаження процесора, використання пам’яті, стан інтерфейсів) і управління мережею. Еволюція SNMP:

- SNMPv1 і SNMPv2: Перші версії SNMP дозволяли здійснювати базовий моніторинг, але мали ряд обмежень у безпеці та масштабованості;

- SNMPv3: Захищена версія протоколу, яка включає аутентифікацію, шифрування даних та контроль доступу, що є критично важливим при роботі у сучасних мережах з високими вимогами до безпеки.

За допомогою SNMP здійснюється регулярний опит пристроїв з частотою до 1 запиту на секунду. Це дозволяє отримати актуальні дані про продуктивність, такі як пропускна здатність портів, статистика обробки пакетів, використання ресурсів пристроїв. Інструменти управління, такі як Zabbix, Nagios, SolarWinds або PRTG, активно використовують SNMP для централізованого збору та візуалізації даних, допомагаючи адмініструвати мережу в режимі реального часу.

Кожна з технологій має свої сильні сторони та обмеження. Нижче наведено узагальнену таблицю, яка допомагає порівняти ключові параметри (табл.1.3).

Реалізація технологій моніторингу у сучасних мережах передбачає інтеграцію різних протоколів для досягнення максимальної видимості та точності аналізу:

- Інтегровані рішення: Багато сучасних платформ моніторингу підтримують одночасну роботу SFlow, NetFlow/IPFIX та SNMP. Це дозволяє адміністраторам комбінувати дані з різних джерел для побудови комплексної картини мережевого трафіку;

- Аналітичні платформи та візуалізація: Дані, зібрані за допомогою цих протоколів, можуть бути інтегровані в аналітичні платформи (наприклад, Grafana,

Kibana, Splunk), що дозволяють здійснювати глибокий аналіз, кореляцію подій і прогнозування навантаження;

- Адаптивність і масштабованість: У дуже великих мережах або центрах обробки даних використання протоколів SFlow і NetFlow/IPFIX дозволяє забезпечити високий рівень точності, не перевантажуючи апаратне забезпечення. SNMP може використовуватися для оперативного моніторингу критичних елементів інфраструктури, таких як маршрутизатори та комутатори.

Таблиця 1.3. Класифікація бітів пріоритету

Параметр	*SFlow*	*NetFlow / IPFIX*	*SNMP*
Рівень OSI	2-7 (багатопротокольний)	3 (IP-трафік)	1-7 (збір даних про пристрої і інтерфейси)
Метод збору	Статистична вибіркова дискретизація	Повний збір даних про потоки	Полінгове опитування (зазвичай періодичне)
Гнучкість	Висока: підтримка усіх протоколів	Висока: NetFlow v9 / IPFIX із шаблонною структурою	Обмежена: стандартний набір показників
Вплив на ресурси пристроїв	Дуже низький за рахунок спеціалізованого чіпа	Низький – середній (залежить від реалізації)	Зазвичай низький, але залежить від частоти опитування
Підтримка безпеки	Немає внутрішніх механізмів безпеки	Залежить від реалізації; IPFIX має можливості інтеграції з безпекою	SNMPv3 – має аутентифікацію та шифрування
Основні застосування	Багатопротокольний аналіз, віртуалізовані середовища	Детальний аналіз IP-потоків, планування пропускної здатності, виявлення аномалій	Моніторинг стану пристроїв, базова статистика мережі

Грунтовний аналіз мережевого трафіку є невід'ємною частиною сучасного управління мережею. Завдяки впровадженню та інтеграції технологій SFlow, NetFlow/IPFIX і SNMP адміністратори можуть отримати багатоаспектну картину стану мережі, відстежувати її продуктивність у реальному часі та своєчасно реагувати на будь-які аномалії. Сучасні реалізації цих технологій дозволяють знизити навантаження на обладнання, забезпечити високу точність даних і адаптувати моніторинг до специфічних вимог різних середовищ, що робить їх основою для побудови стійких і продуктивних мереж.

1.2 Розробка алгоритмів оптимізації та безпеки передачі даних за допомогою механізму NBAR

Одним із сучасних методів управління та оптимізації мережевого трафіку є використання технології Network-Based Application Recognition (NBAR), яка дозволяє розпізнавати та класифікувати мережеві додатки та протоколи для точнішого контролю над потоками даних. За допомогою цього механізму можна ефективно виявляти різні типи трафіку, що проходить через мережу, та застосовувати політики оптимізації чи обмеження до них.

1.2.1 Розробка структури БСА оптимізації передачі даних

На рисунку 1.8 представлена блок-схема алгоритму оптимізації мережевого трафіку із використанням NBAR. Головна мета цього алгоритму — забезпечення ефективного управління ресурсами мережі шляхом динамічного коригування пропускної здатності та пріоритезації трафіку. Основні етапи алгоритму:

1. Початкове введення даних з файлу конфігурації, який містить налаштування щодо обмеження трафіку для різних типів даних.

2. Отримання інформації з таблиці портів мережевого пристрою для аналізу активного трафіку і визначення, через які порти передаються різні типи даних.

3. Видалення даних з файлу конфігурації після обробки для уникнення дублювання інформації та запобігання зайвим витратам ресурсів.

4. В разі отримання сигналу про переривання, система запускає функцію опитування мережевого пристрою, щоб отримати актуальні дані про стан трафіку і відповідно оновити налаштування.

Основна ідея полягає в динамічному управлінні трафіком шляхом розпізнавання та класифікації його типів, а також застосуванні відповідних політик, які забезпечують оптимальне використання мережевих ресурсів. Для цього будуть задіяні такі механізми:

- SSH (Secure Shell) – для безпечного з'єднання та передачі команд на мережеві пристрої, забезпечуючи надійну передачу даних між пристроями та захист конфіденційної інформації під час взаємодії з маршрутизатором;

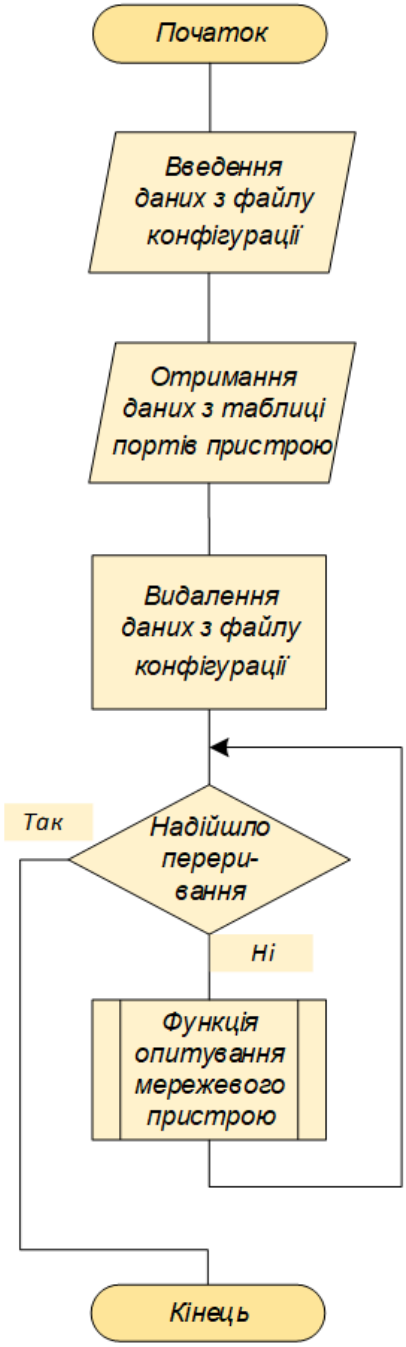


Рисунок 1.8. БСА оптимізації мережевого трафіку за допомогою механізму NBAR

- NBAR (Network-Based Application Recognition) – механізм для розпізнавання та класифікації трафіку, який дозволяє точно ідентифікувати додатки і протоколи в трафіку та відповідно налаштовувати обмеження або пріоритизацію для різних типів даних.
- SNMP (Simple Network Management Protocol) – протокол, що використовується для зчитування показників з мережевих пристроїв, таких як завантаженість портів, використання ресурсів тощо. Цей механізм дозволяє оперативно отримувати інформацію про стан мережі та приймати

рішення на основі актуальних даних.

Взаємодія цих механізмів дозволить створити комплексний підхід до оптимізації трафіку. Алгоритм починається з введення даних з конфігураційного файлу та отримання інформації з таблиці портів пристрою. На основі цих даних застосовуються правила обмеження або пріоритизації трафіку. Після цього здійснюється динамічний моніторинг мережі за допомогою SNMP для оперативного реагування на зміни трафіку.

Цей алгоритм забезпечить:

- Захищений обмін інформацією та управління пристроями через SSH.
- Точну ідентифікацію додатків за допомогою NBAR.
- Швидке реагування на зміни в мережевому середовищі через моніторинг за SNMP.

Таким чином, запропоноване рішення дозволяє не лише оптимізувати мережевий трафік, але й забезпечити ефективне управління ресурсами мережі в реальному часі, що є критично важливим для підтримки роботи ключових сервісів та підвищення якості обслуговування.

1.2.2 Розробка БСА зчитування таблиці портів маршрутизатору

Оптимізація трафіку і накладення обмежень на його передачу вимагають зчитування даних про кількість вхідних октетів і класифікації трафіку на WAN-порту мережевого маршрутизатора. Потім на основі цих даних може бути застосована політика обмеження на LAN-порти для зменшення трафіку непотрібного типу.

Для маркування портів та передачі додаткових даних, які можуть знадобитися в подальшій роботі, доцільно використовувати параметр description при налаштуванні порту. Це дозволяє легше ідентифікувати порти та додавати метаінформацію, яка допоможе в аналізі.

Інформація, аналогічна тій, що надається командою show interfaces, доступна також через SNMP-запит із записом об’єкта ідентифікатора (OID). Цей запит дозволяє отримати повну інформацію про всі порти пристрою, включно з їхніми описами, а також кількістю вхідних та вихідних октетів для кожного порту.

Алгоритм зчитування таблиці портів маршрутизатора з використанням SNMP-запиту включає два основні етапи:

1. Виконання SNMP-запиту з записом відповідного OID для отримання даних про порти.

2. Створення об’єктів із зібраними даними для подальшої обробки та аналізу.

Цей процес зображений на блок-схемі алгоритму читання таблиці портів маршрутизатора на рис. 1.9.

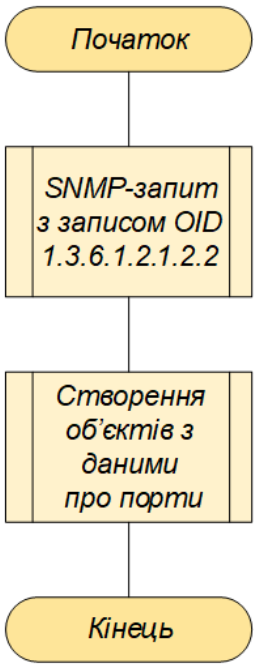


Рисунок 1.9. БСА зчитування таблиці портів маршрутизатору

1.2.3 Розробка БСА моніторингу навантаження на канал зв’язку мережі

Моніторинг навантаження на канал зв’язку є важливою складовою процесу оптимізації трафіку в мережах. Для забезпечення стабільної роботи системи, слід своєчасно визначати ситуації, коли потрібно застосувати обмеження швидкості передачі певних типів трафіку. Причиною таких обмежень може бути перевантаження каналу зв’язку.

Використовуючи SNMP-запити до маршрутизатора, можна регулярно отримувати інформацію про кількість вхідних октетів на портах. Для визначення навантаження на канал зв’язку використовується наступна формула:

$$C = \frac{N_t - N_{t-\Delta}}{\Delta} \qquad (1.1)$$

де:

- N_t – кількість вхідних октетів в момент часу t;
- $N_{t-\Delta}$ – кількість вхідних октетів в момент часу $t - \Delta$;
- Δ – частота оновлення інформації;
- C – поточне завантаження каналу в бітах за секунду.

Застосування SNMP-запиту з відповідним OID дозволяє отримати поточну кількість вхідних октетів. Зберігаючи два останніх результати та враховуючи інтервал між запитами, можна розрахувати поточне завантаження каналу зв'язку. У разі перевищення певного порогового значення навантаження, визначеного як відсоток від реальної швидкості каналу зв'язку, слід дослідити причини такого навантаження.

Для цього разом із інформацією про кількість вхідних октетів необхідно зчитувати таблицю використання типів трафіку, яка доступна через SNMP-запит з відповідним OID. Це дозволить визначити, який тип трафіку дає найбільше навантаження на канал.

Алгоритм визначення завантаженості каналу включає наступні етапи:

1. Створення SNMP-запиту до маршрутизатора для отримання інформації про стан портів та типи трафіку.
2. Визначення наявності великого навантаження на канал зв'язку.
3. Визначення типу трафіку, який спричиняє найбільше навантаження.
4. Перевірка можливості обмеження даного типу трафіку.
5. У разі можливості – передача команд обмеження на маршрутизатор через SSH.

Постійного обмеження трафіку може не бути, тому доцільно застосовувати політики обмеження на певний період часу. Час початку та тривалість дії політики повинні фіксуватися, після чого необхідно відстежувати закінчення терміну дії політики та, у разі потреби, скасовувати її.

Блок-схема алгоритму моніторингу навантаження на канал зв'язку наведена на рис. 1.10.

Для коректної роботи системи моніторингу важливо встановити відповідну

частоту оновлення інформації. Це дозволить уникнути затримок у виявленні перевантаження та швидко реагувати на зміни в трафіку. Важливо враховувати, що не всі типи трафіку можуть бути обмежені без негативних наслідків для роботи мережі. Наприклад, трафік, пов'язаний із критичними бізнес-процесами, повинен мати вищий пріоритет і не підлягати обмеженню. Застосування обмежень має бути тимчасовим і відбуватися лише за потреби. Після того, як рівень навантаження на канал знизиться, обмеження повинні бути автоматично зняті.

Для більш гнучкого управління політиками обмежень варто розробити механізм, що дозволить динамічно змінювати порогові значення навантаження, які тригерять застосування обмежень.

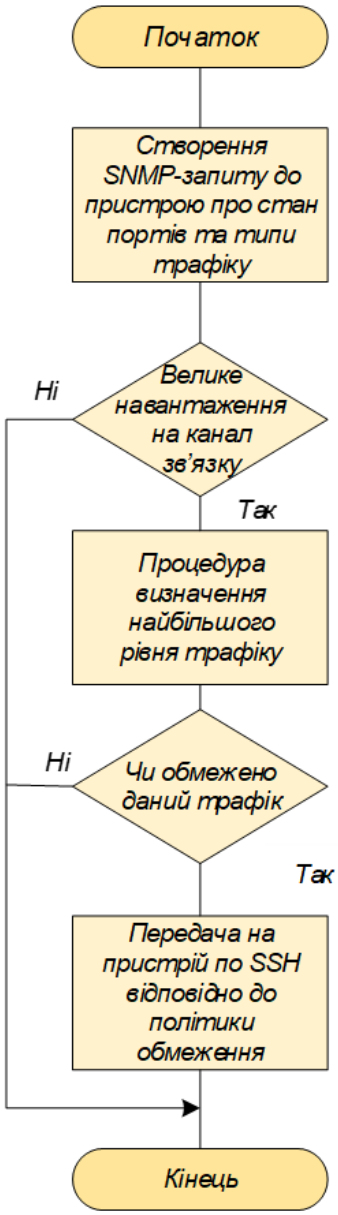


Рисунок 1.10. БСА моніторингу навантаження каналу зв'язку у мережі

1.2.4 Розробка БСА скидання файлу конфігурації роутеру

У процесі експлуатації мережевого обладнання може виникнути ситуація, коли аварійне завершення роботи програми призводить до того, що на мережевому пристрої залишаються активні політики обмеження трафіку. Це може спричинити небажані наслідки для мережі, зокрема, погіршення продуктивності або втрату доступу до критично важливих сервісів. Для запобігання таких ситуацій необхідно реалізувати механізм автоматичного скидання файлу конфігурації маршрутизатора після аварійного завершення програми.

Алгоритм скидання конфігурації передбачає кілька важливих кроків, які дозволяють забезпечити правильне очищення конфігураційного файлу та відновлення нормальної роботи мережевого пристрою. Блок-схема алгоритму наведена на рис. 1.11.

Основні етапи алгоритму скидання конфігурації маршрутизатора:

1. Формування SSH-запиту до мережевого пристрою: перший крок полягає у встановленні SSH-з'єднання з маршрутизатором. Через це з'єднання програма має отримати доступ до конфігураційного файлу пристрою та виконати необхідні дії для скасування політик обмеження.

2. Відбір і сортування результатів SSH-запиту: після отримання результатів з пристрою слід здійснити аналіз отриманої інформації. Зокрема, необхідно зчитати список усіх активних політик обмеження та класів трафіку, до яких вони застосовані. Результати слід відсортувати відповідно до пріоритетності політик, щоб забезпечити правильний порядок їх видалення.

3. Формування команд для очищення конфігурації: на основі отриманої інформації формуються команди для скасування політик обмеження. Важливо забезпечити правильну послідовність команд, щоб уникнути конфліктів при виконанні їх на пристрої.

4. Передача команд через SSH до мережевого пристрою: завершальний етап алгоритму передбачає передачу сформованих команд на маршрутизатор через SSH-з'єднання. Після виконання команд політики обмеження мають бути повністю видалені з конфігураційного файлу, що дозволить відновити нормальну роботу

мережі.

Реалізація алгоритму скидання конфігурації маршрутизатора дозволяє уникнути ситуацій, коли політики обмеження залишаються активними після аварійного завершення програми. Це допомагає забезпечити стабільну роботу мережевого обладнання та запобігти потенційним проблемам з продуктивністю мережі.

Після запуску програми важливо зчитати актуальну інформацію про політики обмеження, щоб програма могла виконати їх скасування у разі необхідності. Такий підхід також дозволяє автоматизувати процес управління політиками обмеження, що підвищує ефективність роботи системи загалом.

Рисунок 1.11. БСА скидання файлу конфігурації маршрутизатору

1.2.5 Реалізація налаштувань у конфігураційному файлі

Для забезпечення коректної роботи створюваної програми необхідно виконати кілька налаштувань і визначити відповідні параметри:

- логін та пароль для доступу через SSH;
- доменне ім’я або IP-адресу маршрутизатора;
- community-рядок для встановлення з'єднання через SNMP;
- період оновлення даних для моніторингу роботи мережевого пристрою;
- тривалість дії політики обмеження;
- коефіцієнт завантаження каналу зв’язку (порівняння поточного навантаження з максимальною пропускною здатністю), що активує політику обмеження;
- співвідношення дозволеного трафіку до поточного навантаження, при перевищенні якого швидкість небажаного трафіку буде обмежена;
- список протоколів, на які не поширюватиметься політика обмеження.

Для опису конфігурацій рекомендується використовувати формат YAML, який вирізняється простотою читання та високою швидкістю обробки. Використання технології NBAR в оптимізації мережевого трафіку дозволяє автоматизувати контроль навантаження на канал зв’язку та застосування відповідних політик обмеження. Це значно знижує кількість рутинних завдань для адміністратора, а також дає змогу динамічно регулювати швидкість для певних типів трафіку, резервуючи необхідну пропускну здатність для критично важливих видів даних.

1.3 Програмна реалізація алгоритмів оптимізації та безпеки передачі даних за допомогою механізму NBAR

Створюване програмне забезпечення відповідно до технічного завдання має виконувати розпізнавання потоку даних мережі для маршрутизаторів із використанням механізму Network-Based Application Recognition (NBAR). Основним завданням є забезпечення оптимізації та безпеки переданої інформації шляхом моніторингу, аналізу та встановлення політик обмеження доступу.

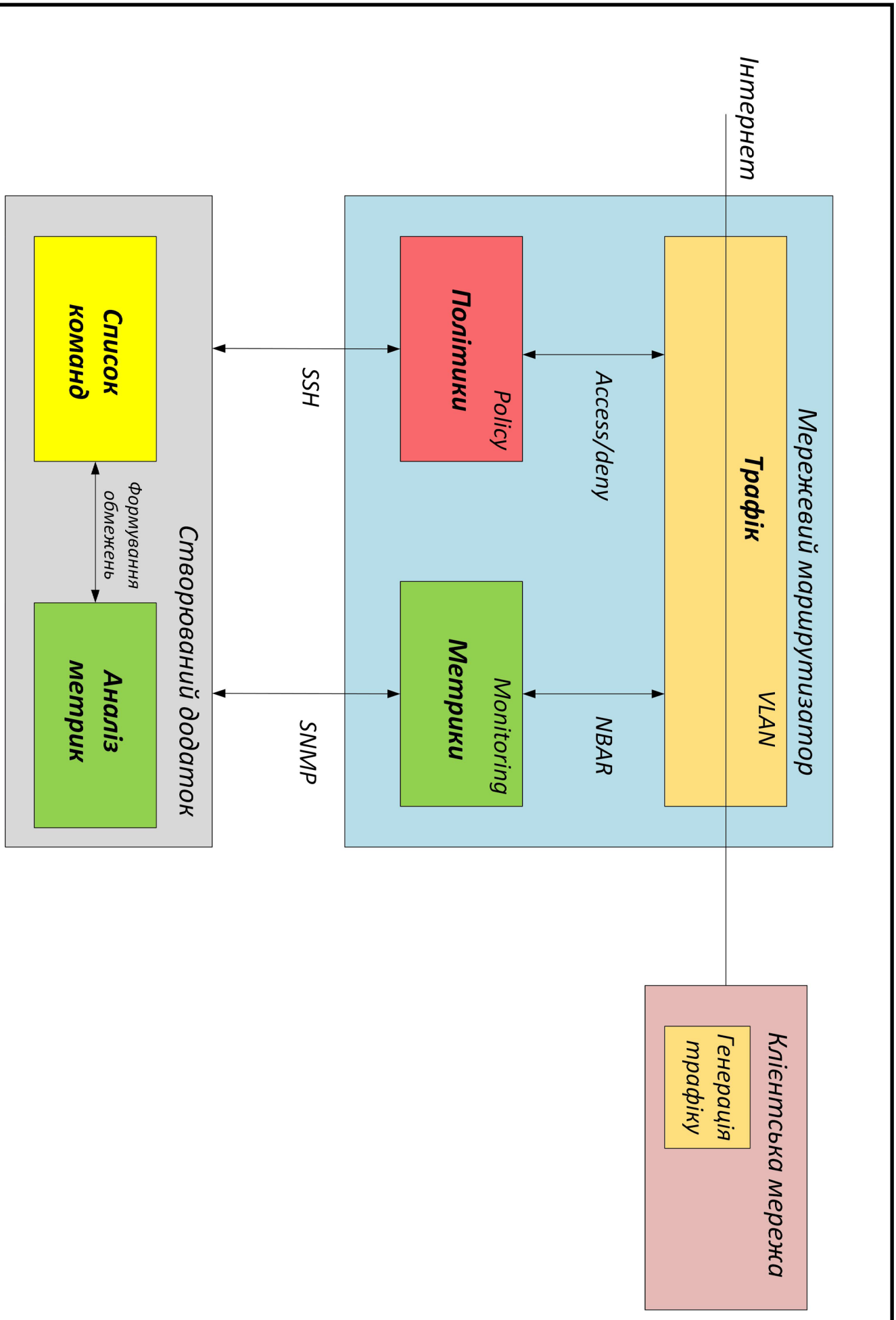


Рисунок 1.12. Структурна схема системи оптимізації та безпеки передачі даних

На рисунку 1.12 представлена структурна схема реалізації програмного рішення. Вона складається з кількох основних компонентів:

- Мережева інфраструктура: мережевий маршрутизатор, через який проходить трафік з Інтернету та клієнтської мережі;
- Трафік: Дані, що передаються через маршрутизатор, сегментуються у VLAN;
- Моніторинг трафіку (NBAR): Виявлення та класифікація додатків, які використовують пропускну здатність мережі;
- Політики доступу: Впровадження обмежень або дозволів на основі аналізу трафіку;
- Створюваний додаток: Відповідає за аналіз отриманих метрик та формування команд для маршрутизатора через SSH.

Етапи створення програмного рішення

1. Аналіз вимог та постановка задачі:
   - Визначення вимог до програмного рішення;
   - Дослідження особливостей роботи NBAR у маршрутизаторах;
   - Розробка методології оптимізації трафіку та безпеки;
2. Проектування архітектури:
   - Розробка структурної схеми програми (рис. 1.12);
   - Визначення способів взаємодії компонентів (SNMP для моніторингу, SSH для конфігурації політик);
   - Проектування форматів даних для аналізу метрик;
3. Розробка програмного забезпечення:
   - Реалізація модуля збору метрик трафіку через SNMP;
   - Створення алгоритмів аналізу отриманих даних;
   - Генерація команд для застосування політик та відправка їх маршрутизатору через SSH;
4. Тестування та оптимізація:
   - Перевірка коректності збору та аналізу метрик;
   - Налаштування політик та їх впливу на продуктивність мережі;

– Виправлення помилок та покращення алгоритмів;

5. Впровадження та експлуатація:

– Налаштування програмного рішення в реальному середовищі;
– Моніторинг роботи та внесення коригувань;
– Створення документації щодо використання та налаштування.

1.3.1 Визначення і обґрунтування програмних інструментів розробки

Для реалізації моделі оптимізації та безпеки передачі даних за допомогою механізму NBAR було обрано об’єктно-орієнтовану мову програмування Java. Вибір цієї мови обумовлений такими факторами:

- Крос-платформність: Java є незалежною від операційної системи, що дозволяє запускати створене програмне забезпечення на різних платформах (Windows, Linux, Unix, Mac OS, Android). Це спрощує розгортання та підтримку системи;
- Широкий вибір бібліотек та інструментів: Java має велику кількість фреймворків та бібліотек, що дозволяють ефективно працювати з мережею, аналізувати трафік та виконувати моніторинг;
- Розвинуті інтегровані середовища розробки (IDE): Зокрема, IntelliJ IDEA та Eclipse, які підтримують автоматичне доповнення коду, інтеграцію з системами контролю версій та засобами тестування;
- Популярність та актуальність: Java залишається однією з найпоширеніших мов розробки, що гарантує доступність документації та підтримку спільноти.

Інструменти та технології для реалізації ПЗ:

1. Середовище розробки. Для написання коду обрано IntelliJ IDEA, яке надає широкий набір інструментів для аналізу та налагодження коду. Також використовується Java JDK 8, що забезпечує стабільність та сумісність із більшістю бібліотек.

2. Система управління залежностями. Для автоматизації збірки проєкту використано Apache Maven. Цей фреймворк дозволяє:

– централізовано керувати залежностями проєкту;

- налаштовувати конфігурацію через файли Project Object Model (POM.xml);
- спрощувати розгортання та тестування додатка.

3. Моніторинг та аналіз мережевого трафіку. Для збору метрик та контролю стану мережевого обладнання використовується Check_MK — система моніторингу, побудована на основі Nagios. Основні можливості:
   - підтримка протоколу SNMP, що дозволяє отримувати статистику від маршрутизаторів;
   - використання плагінів для збору даних з пристроїв різних виробників;
   - можливість відображення метрик у вигляді графіків та таблиць.
4. Компоненти Check_MK. Система містить такі складові:
   - Агент Check_MK, що збирає дані з маршрутизаторів;
   - База даних, де зберігаються зібрані метрики;
   - Веб-сервер Apache, який надає інтерфейс для перегляду статистики;
   - Система оповіщення, що дозволяє надсилати повідомлення про критичні події.
5. Моніторингові метрики Check_MK дозволяє отримувати такі дані про стан маршрутизатора:
   - завантаження процесора;
   - температуру системи;
   - використання дискового простору;
   - стан мережевих інтерфейсів;
   - використання оперативної пам’яті;
   - кількість активних потоків.

1.3.2 Розробка об’єктно-орієнтованої моделі

Розроблена модель для розпізнавання мережевого потоку даних та створена програма надають такі функціональні можливості:

- завантаження конфігураційного файлу у форматі YAML;
- взаємодію з пристроями за допомогою протоколів SSH і SNMP;
- створення й видалення тимчасових налаштувань для обмеження окремих видів трафіку;

- розширення функціональних можливостей шляхом додавання MIB інших вендорів.

Програма складається з таких модулів:

- парсер для аналізу конфігураційного файлу;
- класи, які описують пристрій, зберігають поточні конфігурації обмежень та історію отриманих даних;
- модуль для відслідковування змін, автоматичного формування політик обмеження;
- конектори для зв’язку з обладнанням через SSH та SNMP.
- Ключові класи програми виконують специфічні функції:
- Config — відповідає за парсинг YAML-файлу, зберігає список пристроїв (hosts), відслідковуваних програмою.
- ClassMap — описує класи трафіку, включаючи критерії match, назву трафіку name та протоколи protocol, які блокуються.
- Host — відповідає за збереження інформації про кожен пристрій, наприклад, ім’я (name), облікові дані (user і password), community-рядок для SNMP, список підтримуваних протоколів, що не обмежуються.
- Logic — моделює алгоритм роботи.
- SSHClient та SNMPClient — забезпечують взаємодію через відповідні протоколи.

Крім того, змінні класу Host включають дані про порти WAN і LAN, частоту оновлення, час дії обмежень, рівень завантаженості мережі для активації політик і допустимий відсоток дозволеного трафіку. Функціональну схему класів зображено на рис. 1.13, яка демонструє взаємозв’язки між компонентами програми.

Основні класи та їхні обов'язки:

1. Config:
    - Займається парсингом YAML-конфігураційного файлу;
    - Містить список hosts, тобто пристроїв, які будуть відслідковуватись;
    - Має метод, що обробляє шлях до конфігураційного файлу, переданий як параметр;

2. Host — зберігає детальну інформацію про маршрутизатори:
   - name — доменне ім’я або IP-адреса;
   - user, password — облікові дані для SSH;
   - community — community-рядок для SNMP;
   - protocols — перелік протоколів, які не обмежуються;
   - ports, WAN, LAN — списки портів маршрутизатора та інтерфейси WAN/LAN;
   - logic — об’єкт для відслідковування активності;
   - update — частота оновлення інформації;
   - TTL — час дії політики обмеження;
   - bandwidth_usage і allowed_percent — параметри для контролю використання каналу зв’язку;
3. ClassMap — описує класи трафіку, має методи для отримання значень змінних:
   - match — тип критерію (наприклад, на основі протоколу чи назви);
   - name — ім'я класу трафіку;
   - protocol — протоколи, які блокуються;
4. Logic — відповідає за відслідковування змін у мережевому трафіку, формує політики шейпінгу в разі необхідності;
5. PolicyMap — описує конкретні політики обмеження трафіку;
6. Traffic — відображає інформацію про поточне навантаження на мережу;
7. Snapshot і Timestamp: реалізують збереження історичних даних щодо активності мережі, забезпечують зручне порівняння станів мережі у різні моменти часу;
8. SSHClient та SNMPClient — забезпечують комунікацію з пристроями через протоколи SSH і SNMP;
9. Main — інтегрує всі компоненти, забезпечуючи запуск програми та управління її модулями.

Клас Main створює та управляє об'єктами інших класів. Клас Config забезпечує налаштування для Host, які в свою чергу взаємодіють з SSHClient і SNMPClient для

обробки трафіку. Клас Logic використовує дані з Host, Traffic, PolicyMap, та інших компонентів для прийняття рішень.

Згідно зі схемою, кожен клас чітко виконує свою задачу, сприяючи модульності програми. Це дозволяє легко масштабувати функціонал, додаючи підтримку нових пристроїв чи протоколів.

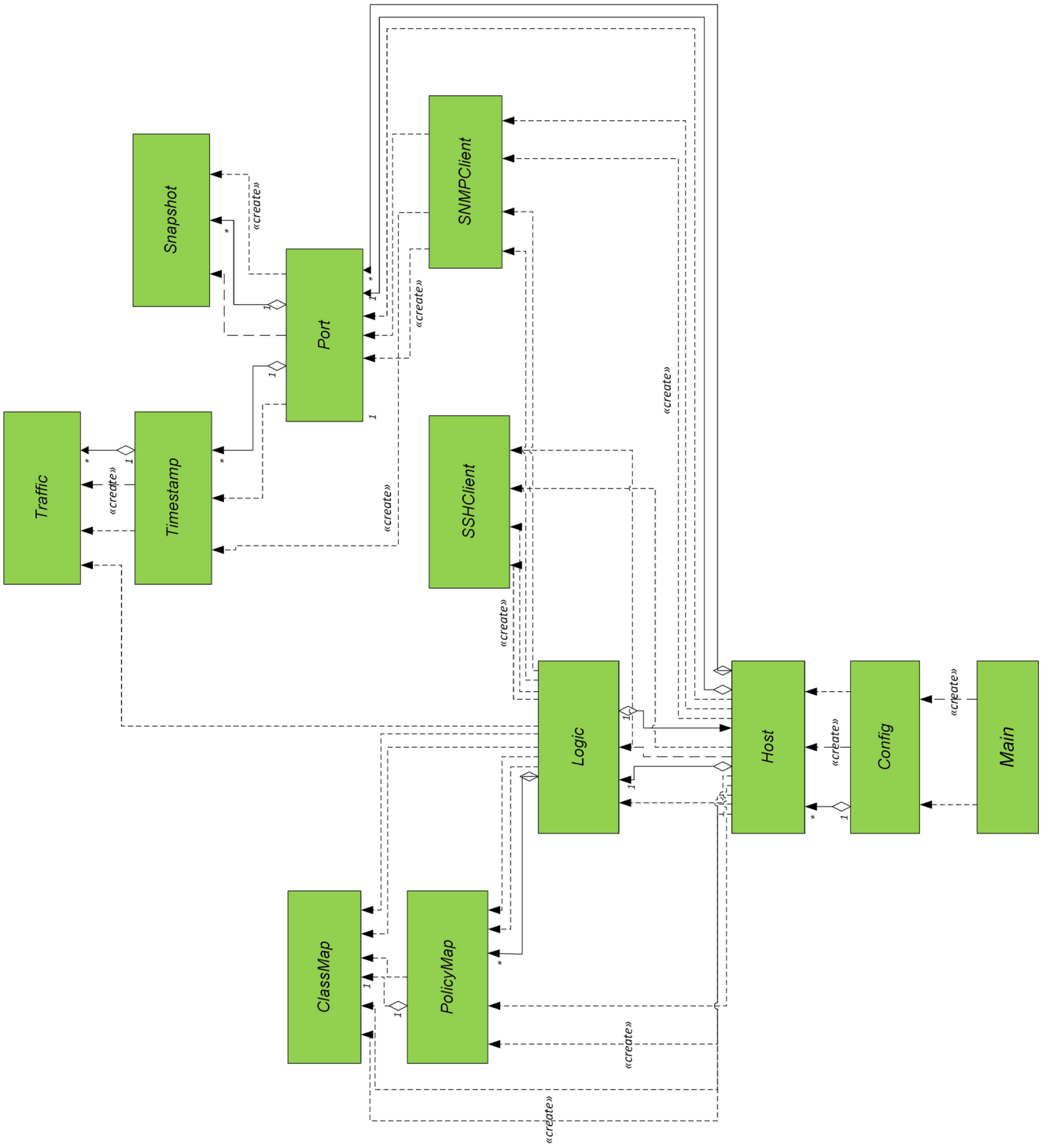


Рисунок 1.13. Функціональна схема діаграми класів у програмі

Методи класу Host:

1. getPorts:
   - Використовує SNMP для отримання списку портів маршрутизатора;
   - Ідентифікує порти WAN та LAN для подальшої роботи програми;
2. makeLogic:
   - Створює об’єкт класу Logic, який відповідає за моніторинг мережевої активності маршрутизатора;
   - Забезпечує автоматизацію процесу відслідковування змін у трафіку;
3. cleanMaps:
   - Очищає об’єкти Policy Map та Class Map, які могли залишитись на пристрої через некоректне завершення роботи програми;
   - Гарантує "чистий" стан перед застосуванням нових конфігурацій;
4. isData:
   - Перевіряє назви об'єктів Policy Map чи Class Map на наявність дати;
   - Використовується для визначення актуальних даних у політиках;
5. makeRevercePMAP:
   - Генерує список команд для видалення об'єктів Policy Map з маршрутизатора;
   - Забезпечує коректне скасування політик шейпінгу;
6. makeReverceCMAP:
   - Генерує список команд для видалення об'єктів Class Map з маршрутизатора;
   - Використовується для повернення маршрутизатора до стандартного стану;

Службові методи класу Host:

1. getName:
   - Повертає ім’я маршрутизатора (доменне ім'я або IP-адресу);
   - Використовується для ідентифікації пристрою;
2. getUser:
   - Повертає ім’я користувача, необхідне для встановлення з'єднання через

SSH;

- Сприяє налаштуванню доступу;

3. getPassword:
   - Повертає пароль для аутентифікації по SSH;
   - Використовується для забезпечення безпечного підключення;
4. getProtocols:
   - Повертає список протоколів, які не будуть обмежуватись політиками шейпінгу;
   - Сприяє виключенню специфічного трафіку з-під обмежень.

Класи для управління трафіком та збору статистики:

Клас PolicyMap відповідає за опис політики обмеження трафіку на основі протоколів, визначених у відповідних Class Map:

- classMap — тип класу трафіку;
- name — назва політики;
- shaper — максимальна швидкість передачі даних для цієї політики;
- TTL — тривалість застосування політики;
- reverceCommands — список команд для відміни політики через SSH.

Клас також містить службові методи для встановлення і отримання значень змінних.

Клас Port описує мережевий порт маршрутизатора:

- ifIndex — індекс порту;
- snapshotList — список звітів статистики використання мережі портом у визначені моменти часу;
- TimestampList — список звітів, що включають статистику із розпізнаванням типів трафіку за допомогою NBAR.

Методи класу Port:

- addInfo — додає нову інформацію до списку звітів статистики;
- Службові методи забезпечують встановлення і повернення значень змінних.

Клас Snapshot відповідає за збереження даних про використання мережі окремим портом у певний момент часу:

- ifDescr — опис порту;
- ifAlias — "псевдонім" порту;
- ifInOctets — загальна кількість отриманих октетів, включаючи символи кадрування;
- realSpeed — швидкість передавання даних у середовищі;

Методи класу Snapshot:

- addInfo — додає нові дані до звіту;
- Містить і інші змінні, отримані через SNMP, які можуть використовуватись у майбутньому.

Клас Timestamp відповідає за звіти статистики з розпізнаванням типів трафіку за допомогою NBAR:

- Time — момент формування звіту;
- dump — детальна інформація про кожен тип розпізнаного трафіку.

Методи класу Timestamp:

- addInfo — додає нову інформацію до звіту;
- Службові методи відповідають за управління даними змінними.

Клас Traffic використовується для збору статистики про типи трафіку:

- Time — момент зчитування інформації;
- cnpdAllStatsProtocolsName — назва розпізнаного типу трафіку;
- cnpdAllStatsInBytes — загальна кількість вхідних пакетів певного типу трафіку.

Методи класу Traffic:

- addInfo — додає дані до відповідних полів;
- Містить додаткові змінні, отримані через SNMP, які можуть бути застосовані у подальшому.

Взаємодія між класами:

- PolicyMap формує політики обмеження на основі інформації з Class Map;
- Port забезпечує прив’язку статистичних звітів (Snapshot, Timestamp) до конкретних мережевих портів;
- Snapshot та Timestamp доповнюють один одного у збереженні історичних

даних про використання мережі;

- Traffic надає деталізовану інформацію про типи трафіку, допомагаючи формувати ефективні політики шейпінгу.

Клас Logic відповідає за логіку моделювання алгоритму. Його змінні включають:

- WAN — порт маршрутизатора для з'єднання з Інтернетом;
- LAN — порт для з'єднання з локальною мережею;
- bandwidth — швидкість середовища передачі даних;
- percent — пороговий рівень завантаження каналу, до якого політики обмеження не активуються;
- allowedPercent — співвідношення дозволених протоколів до інших для активації політики;
- allowed — список протоколів, які не обмежуються;
- policyMaps — список застосованих політик обмеження;
- shaper — максимальна швидкість передачі даних для політики;
- host — пристрій, трафік якого аналізується;
- update — інтервал оновлення даних;
- TTL — час дії політики обмеження;

Методи класу Logic:

1. watch:

- Отримує актуальні дані з маршрутизатора;
- На основі даних приймає рішення про застосування політик обмеження;

2. makeShaping:

- Формує політики та класи трафіку;
- Застосовує відповідні команди на маршрутизаторі;
- Формує зворотні команди для скасування політик;

3. getCurrentTimeUsingDate повертає поточний час у форматі yyyyMMddHHmmss;

4. isData перевіряє назви об'єктів Policy Map чи Class Map на наявність дати;

5. makeCommand створює набір команд для активації Policy Map;

6. makeReverce генерує команди для скасування політик та класів трафіку.

7. checkMaps перевіряє термін дії існуючих політик;

8. stringToDate конвертує рядок yyyyMMddHHmmss у формат об’єкта Date;

Службові методи класу Logic: getCurrentDate — повертає поточну дату у вигляді об’єкта Date.

Клас SSHClient забезпечує SSH-з’єднання з маршрутизатором:

- Передає команди на пристрій;
- Отримує та повертає результати виконання команд.

Клас SNMPClient реалізує SNMP-з’єднання:

- Виконує зчитування інформації за вказаними OID;
- Повертає дані у потрібному форматі.

Логіка роботи програми є такою:

1. Ініціалізація:
   - Метод main у класі Main створює об’єкт Config;
   - Об’єкт Config зчитує конфігураційний файл (YAML) і генерує об’єкт Host із відповідними параметрами;
2. Початкове налаштування:
   - Об’єкт Host отримує список портів через SNMP;
   - Очищає залишки політик чи класів, що могли залишитись після аварійного завершення;
   - Формує об’єкт Logic для управління трафіком;
3. Моніторинг:
   - Метод Logic.watch з інтервалом зчитує статистику SNMP для портів і детальні звіти по типах трафіку;
   - Якщо канал завантажений понад пороговий рівень, перевіряється вплив "дозволених" протоколів;
4. Застосування політик:
   - Створюються класи трафіку та політики обмеження;
   - Політики застосовуються на пристрої та додаються до списку активних;
5. Скасування старих політик:

- Регулярно перевіряються строки дії політик;
- При необхідності виконуються зворотні команди для скасування;

6. Інтеграція: Завдяки класам SSHClient і SNMPClient відбувається взаємодія з маршрутизатором.

На схемі (рис. 1.14) показано послідовність виконання основних дій, включаючи ініціалізацію, моніторинг і управління політиками.

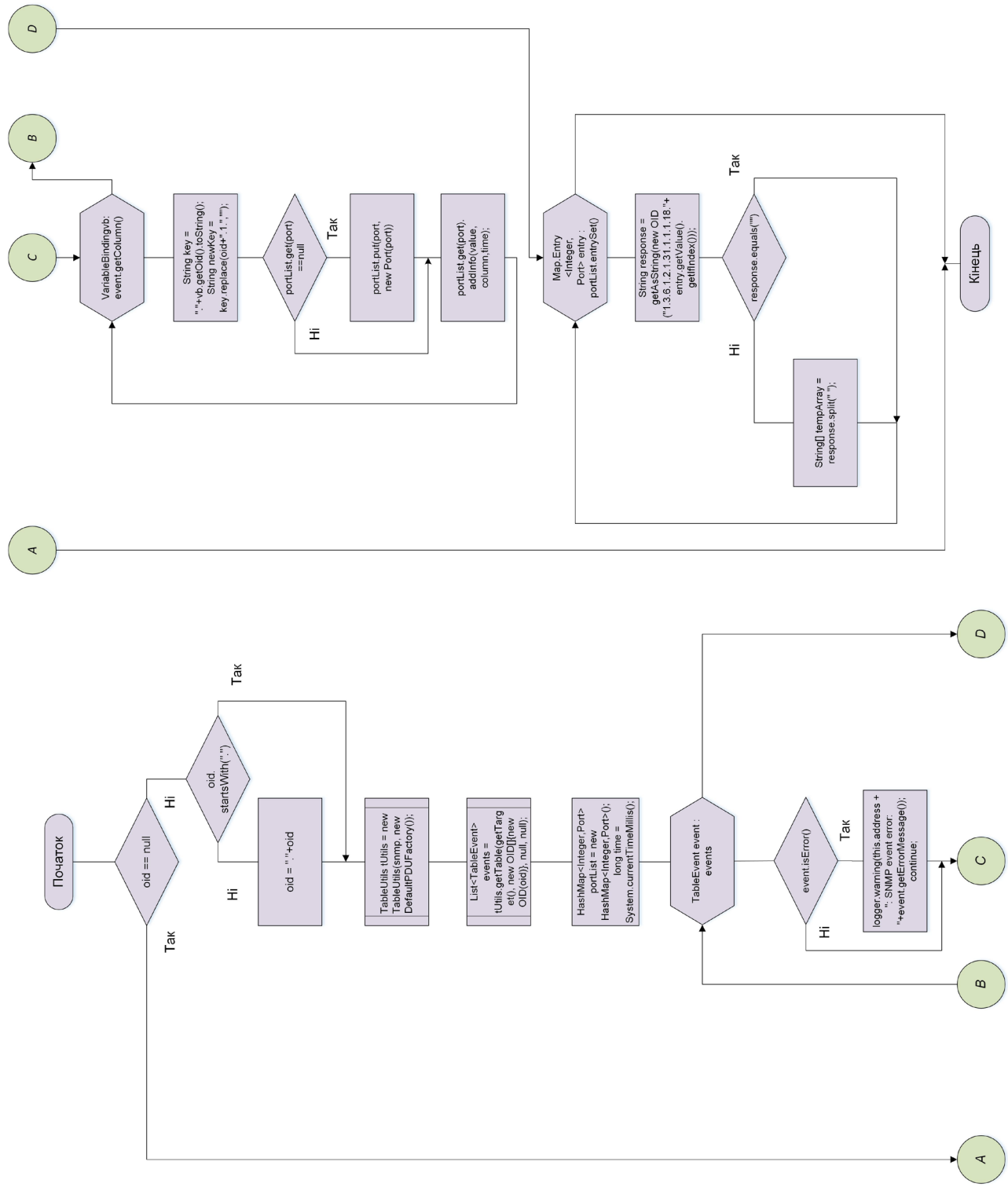


Рисунок 1.14. БСА розпізнавання потоку даних та управління політиками шейпінгу

Програма розпочинає свою роботу з виклику методу main, який створює об’єкт Config для зчитування налаштувань із YAML-конфігураційного файлу. Цей файл містить усі необхідні параметри для з’єднання з маршрутизатором, такі як логін, пароль, доменне ім’я або IP, SNMP community, інтервали оновлення, а також конфігурації політик обмеження трафіку. Після завантаження конфігурації створюється об’єкт Host, який ініціює зчитування списку портів за допомогою SNMP, визначає WAN та LAN, а також очищує залишкові налаштування класів та політик, які могли залишитися після аварійних завершень роботи. Одразу ж формується об’єкт Logic для управління мережевою активністю, який завдяки SSH (через клас SSHClient) отримує інформацію про пропускну здатність порту WAN, а з допомогою SNMP (через SNMPClient) – початкові звіти статистики, що фіксуються як сніпшоти і детальні звіти із застосуванням NBAR. Далі, у встановлених інтервалах, Logic за допомогою методу watch отримує актуальні звіти та аналізує поточне завантаження каналу порівняно з пороговими значеннями, визначеними параметрами percent та allowedPercent. Якщо мережеве навантаження перевищує допустимі межі, і трафік не належить до категорії дозволених, програма автоматично формує політики обмеження, використовуючи метод makeShaping, який створює відповідні класифікації трафіку та політики (Policy Map) із зазначенням максимальної швидкості (shaper) і терміну дії (TTL). Сформовані команди для застосування політик передаються на маршрутизатор через SSH, при цьому одночасно формується список зворотних команд для скасування цих політик, коли їх час дії спливає. Метод checkMaps перевіряє, чи завершився термін дії застосованих політик, і, якщо так, через метод makeReverce генерує команди для їх скасування, повертаючи пристрій до нормального режиму роботи. Допоміжні методи, такі як getCurrentTimeUsingDate та stringToDate, забезпечують управління часовими мітками, що використовуються для точного визначення терміну дії політик. Цей циклічний процес моніторингу, аналізу і регулювання трафіку повторюється постійно протягом роботи програми, що дозволяє автоматично коригувати налаштування мережевого трафіку, оптимізувати пропускну здатність каналу і мінімізувати ручне втручання

адміністратора. Інтегроване використання SSH та SNMP забезпечує безперервність контролю мережевого стану та динамічне застосування обмежуючих політик, що сприяє підвищенню ефективності мережевої інфраструктури.

1.3.3 Опис інтерфейсу розробленого програмного забезпечення

Створений програмний продукт реалізує алгоритм оптимізації мережевого трафіку, використовуючи механізм NBAR. Інтерфейс програми побудований у вигляді консольного застосунку, який взаємодіє з маршрутизатором через протоколи SSH та SNMP. Основним елементом конфігурації програми є YAML-файл, що містить параметри роботи маршрутизатора, такі як IP-адреса, облікові дані для доступу, частота оновлення метрик, порогові значення завантаженості каналу та список протоколів, що не підлягають обмеженням.

Приклад конфігураційного файлу:

```
host:
  name: 192.168.1.1
  user: admin
  password: admin123
  community: public
  update: 10
  TTL: 20
  bandwidth_usage: 0.75
  allowed_percent: 0.6
  protocols:
    - http
    - https
    - dns
    - icmp
```

Програма використовує ці параметри для автоматизації моніторингу та управління трафіком. Під час роботи вона зчитує метрики використання мережевого інтерфейсу маршрутизатора через SNMP, аналізує отримані дані та у разі перевищення порогових значень генерує команди для обмеження небажаного трафіку. Взаємодія з маршрутизатором здійснюється через SSH, що дозволяє надсилати команди в реальному часі.

Програма виводить у консоль поточні параметри трафіку, відсоток використання пропускної здатності, типи виявленого трафіку та команди, які були

надіслані на маршрутизатор. Логування забезпечує запис усіх операцій у файл для подальшого аналізу адміністратором. Приклад виводу в консоль:

[INFO] Завантаження каналу: 85% (перевищено поріг у 75%)
[INFO] Виявлений трафік: YouTube (50%), HTTP (30%), FTP (10%)
[INFO] Застосовано обмеження для YouTube
[CMD] Відправлено команду: access-list 110 deny ip any any eq youtube

Для перевірки коректності застосованих політик програма виконує команду show running-config на маршрутизаторі та аналізує отриману конфігурацію. Ця конфігурація показує налаштування інтерфейсів, активацію NBAR, SNMP-сервер та параметри доступу через SSH.

Приклад виводу команди show running-config:

```
interface FastEthernet0/0
 description WAN 100000000
 bandwidth qos-reference 10000000
 ip address 192.168.1.1 255.255.255.0
 ip nbar protocol-discovery
 ip nat outside
 ip virtual-reassembly in
 duplex auto
 speed auto
!
interface FastEthernet0/1
 description LAN 10000000
 ip address 10.0.0.1 255.255.255.0
 ip nat inside
 ip virtual-reassembly in
 duplex auto
 speed auto
!
snmp-server community public RW
snmp-server host 192.168.1.100 version 2c public
line vty 0 4
 exec-timeout 60 0
 privilege level 15
 logging synchronous
 transport input ssh
!
username admin privilege 15 password 7 02050D4808095E731F
ip ssh version 2
```

1.3.4 Тестування розробленої моделі оптимізації та безпеки передачі даних за допомогою механізму NBAR

Для перевірки працездатності розробленої програми було проведено тестування в реальних умовах з використанням маршрутизатора ASR1001-X. Перед початком тестування маршрутизатор налаштовувався відповідно до вимог програмного забезпечення: активовано механізм NBAR, налаштовано SNMP-сервер для збору статистики, SSH-доступ для віддаленого керування та додано описи інтерфейсів для полегшення ідентифікації портів. Загальна структура тестового середовища показана на рис. 1.15.

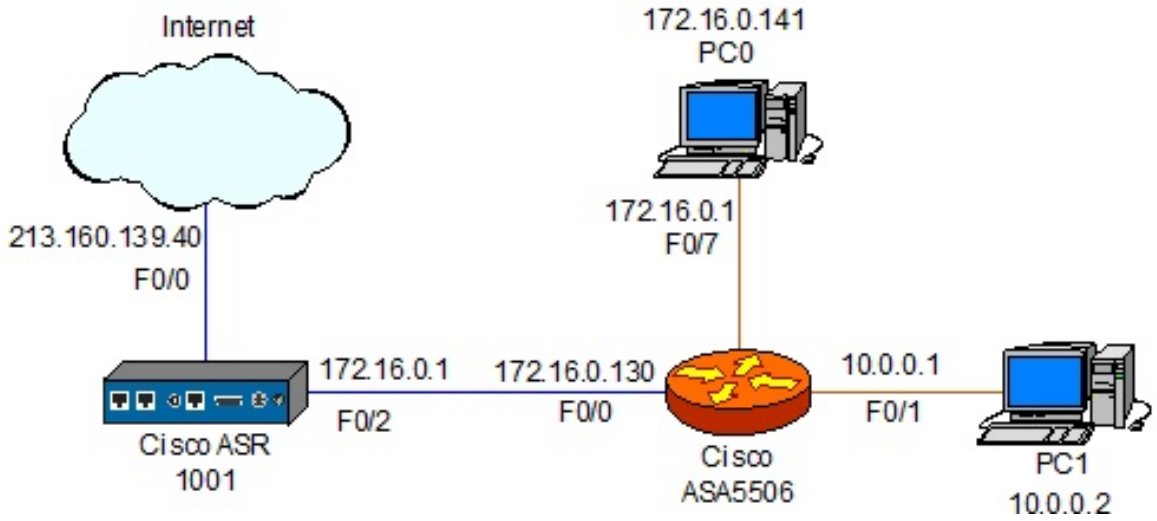


Рисунок 1.15. Модель мережі для тестування розробленого ПЗ

У тестовій мережі були задіяні дві робочі станції під управлінням Windows 10, маршрутизатор ASR1001-X та міжмережевий екран ASA5506. На першій робочій станції (PC0) з IP-адресою 172.16.0.141 було розгорнуто розроблену програму та встановлено віртуальну машину Java 1.8.0. Друга робоча станція (PC1) з IP-адресою 10.0.0.2 використовувалася для генерації тестового трафіку, для чого був завантажений браузер Mozilla Firefox.

Маршрутизатор ASR1001 працював під управлінням IOS Version 15.9. Його інтерфейси були налаштовані наступним чином:

- FastEthernet0/0 (WAN-порт) – IP-адреса 172.16.0.130
- FastEthernet0/1 (LAN-порт) – IP-адреса 10.0.0.1

Міжмережевий екран ASA5506 мав встановлену ASA Software Version 9.9. Він був налаштований так:

- FastEthernet0/0 – зовнішня IP-адреса 213.160.139.40
- FastEthernet0/2, FastEthernet0/7 – внутрішній VLAN із IP-адресою 172.16.0.1

Після розгортання середовища тестування було запущено програму на PC0. Перший етап тестування включав ініціалізацію з'єднання з маршрутизатором та очищення політик обмеження трафіку, що могли залишитися після попередніх запусків. Вивід відповідних команд зображений на рис. 1.16.

```
Hello World!
ASR1001-X#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
ASR1001-X(config)#
ASR1001-X(config)#no class-map CMAP20190415161636
ASR1001-X(config)#
ASR1001-X(config)#exit
ASR1001-X#
ASR1001-X#exit-status: 0
```

Рисунок 1.16. Лістинг ініціалізації програми та очищення політик

Далі було розпочато моніторинг мережевого трафіку. Програма аналізувала вхідні та вихідні з'єднання маршрутизатора за допомогою SNMP та механізму NBAR. Основні метрики завантаженості каналу та розподілу трафіку були отримані та проаналізовані. На рис. 1.17 представлений короткий звіт про навантаження на канал зв’язку, що відображає поточний рівень використання пропускної здатності та найактивніші протоколи, які споживають мережеві ресурси.

```
currentBandwidthUsage 12051.94356830705
Bandwidth 12500
Current percent 0.964155485464564
Allowed percent 0.8
```

Рисунок 1.17. Лістинг звіту навантаження на канал зв’язку у мережі

Наступний етап тестування полягав у детальному аналізі класифікації трафіку. Програма використовувала механізм NBAR для розпізнавання типів переданих даних, що дозволило визначити, які саме протоколи та сервіси найбільше навантажують мережу. Детальна класифікація трафіку представлена на рис. 1.18.

```
NBAR analysis
unknown
 Last usage = 458855
 Prelast usage = 457330
 delta = 1525
http
 Last usage = 1454395352
 Prelast usage = 1417316777
 delta = 37078575
icmp
 Last usage = 11270
 Prelast usage = 11270
 delta = 0
snmp
 Last usage = 4932014
 Prelast usage = 4929906
 delta = 2108
socks
 Last usage = 5129418
 Prelast usage = 5129418
 delta = 0
ssh
 Last usage = 535344
 Prelast usage = 535344
 delta = 0
dns
 Last usage = 62532
 Prelast usage = 62532
 delta = 0
dhcp
 Last usage = 2052
 Prelast usage = 2052
 delta = 0
secure-http
 Last usage = 10888725
 Prelast usage = 10888725
 delta = 0
```

Рисунок 1.18. Лістинг звіту з класифікацією трафіку мережі

На основі отриманих даних програма автоматично застосовувала політики обмеження для певних типів трафіку, що перевищували встановлені порогові значення. Це дозволяло оптимізувати використання каналу зв’язку та запобігати надмірному споживанню ресурсів окремими сервісами.

```
Straight commands:
configure terminal
class-map match-all CMAP20190430193451
match protocol http
exit
policy-map PMAP20190430193451
class CMAP20190430193451
police 10000000 conform-action transmit exceed-action drop
exit
exit
interface FastEthernet0/1
service-policy output PMAP20190430193451
exit
ASR1001-X#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
ASR1001-X(config)#
ASR1001-X(config)#class-map match-all CMAP20190430193451
ASR1001-X(config-cmap)#
ASR1001-X(config-cmap)#match protocol http
ASR1001-X(config-cmap)#
ASR1001-X(config-cmap)#exit
ASR1001-X(config)#
ASR1001-X(config)#policy-map PMAP20190430193451
ASR1001-X(config-pmap)#
ASR1001-X(config-pmap)#class CMAP20190430193451
ASR1001-X(config-pmap-c)#
ASR1001-X(config-pmap-c)#$0000 conform-action transmit exceed-action drop
ASR1001-X(config-pmap-c-police)#
ASR1001-X(config-pmap-c-police)#exit
ASR1001-X(config-pmap-c)#
ASR1001-X(config-pmap-c)#exit
ASR1001-X(config-pmap)#
ASR1001-X(config-pmap)#interface FastEthernet0/1
ASR1001-X(config-if)#
ASR1001-X(config-if)#service-policy output PMAP20190430193451
ASR1001-X(config-if)#
ASR1001-X(config-if)#exit
ASR1001-X(config)#
ASR1001-X(config)#exit
ASR1001-X#
ASR1001-X#exit-status: 0
```

Рисунок 1.19. Лістинг сформованих команд та виведення логу SSH-з’єднання

При виявленні перевищення встановлених лімітів у консолі відображалося повідомлення про застосування обмежень.

Окрім обмеження трафіку, програма формувала відповідні команди конфігурації, які передавалися маршрутизатору через SSH-з'єднання. На рис. 1.19 наведено виведення сформованих команд та журнал SSH-з'єднання. Усі внесені зміни зберігалися в лог-файлах для подальшого аналізу адміністратором мережі.

Таким чином, проведене тестування підтвердило працездатність розробленої програми та її ефективність у оптимізації мережевого трафіку за допомогою механізму NBAR. Програма коректно ідентифікувала типи трафіку, контролювала навантаження каналу та застосовувала обмежувальні політики відповідно до заданих параметрів конфігурації.

Додаткові тестування були проведені для оцінки ефективності роботи розробленої програми в умовах реального мережевого навантаження. Одним із важливих критеріїв оцінки стало порівняння швидкості копіювання великого файлу (ISO-образу) між двома робочими станціями в різних режимах роботи мережі. На рис. 1.20 наведено результати тестування швидкості передачі даних з PC0 до PC1 до та після активації механізму обмеження трафіку.

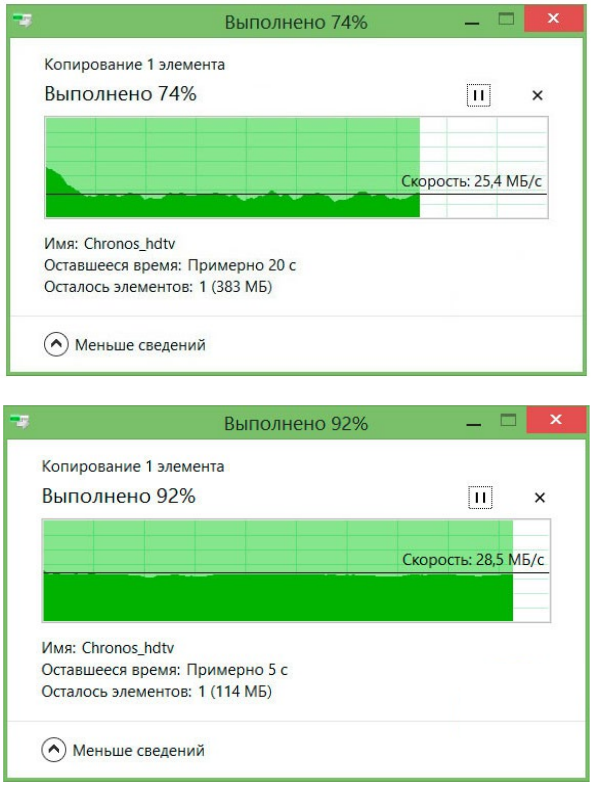


Рисунок 1.20. Тестування швидкості копіювання даних користувача

Як показують результати, після застосування політик обмеження швидкість копіювання файлу через протокол HTTP була зменшена, що дозволило зменшити навантаження на мережевий канал. Водночас, протоколи, що були визначені у конфігураційному файлі як критично важливі, не зазнали впливу з боку обмежувальних політик. Це свідчить про те, що програма коректно ідентифікує типи трафіку та застосовує обмеження вибірково, відповідно до заданих параметрів.

Для подальшої перевірки ефективності програми було проаналізовано загальне навантаження на канал зв'язку до та після включення механізму обмеження. Результати тестування зміни рівня навантаження підтверджують, що програма не блокує небажані типи трафіку повністю, а лише динамічно зменшує їхню швидкість, запобігаючи перевантаженню каналу зв'язку.

Таким чином, розроблена модель продемонструвала здатність у реальному часі здійснювати контроль за використанням пропускної здатності мережі, знижувати вплив некритичних сервісів на продуктивність мережі та забезпечувати стабільну роботу важливих протоколів без втрати швидкості передачі даних.

1.3.5 Аналіз результатів моделювання

Розроблена модель та програмне забезпечення надають автоматизований моніторинг мережевого навантаження та застосування політик обмеження для певних типів трафіку. Класифікація трафіку реалізована за допомогою технології NBAR, що дозволяє точно ідентифікувати протоколи та застосовувати політики управління трафіком відповідно до заданих правил. Програма підтримує конфігураційні файли у форматі YAML, що забезпечує зручність налаштування та інтеграції. Додатково передбачена функція автоматичного очищення конфігураційних записів маршрутизатора у разі аварійного завершення роботи програми, що підвищує стабільність і керованість системи.

Результати тестування підтвердили, що при обмеженні швидкості передачі даних для HTTP-трафіку робота інших критично важливих сервісів не порушується. Запропонований алгоритм дозволяє мінімізувати помилки, зумовлені людським фактором, зменшити потребу в залученні персоналу та автоматизувати

процес обмеження швидкості для непріоритетних видів трафіку.

Для оцінки ефективності алгоритму було проведено порівняльний аналіз ручного налаштування політик обмеження адміністратором мережі та автоматичного керування, реалізованого в програмному забезпеченні. У традиційному підході системний адміністратор повинен особисто підключатися до мережевого обладнання, аналізувати поточний стан трафіку, створювати відповідні правила та контролювати їхню ефективність у реальному часі. Автоматизація цього процесу за допомогою розробленого програмного забезпечення усуває потребу в ручному втручанні, знижує ризик помилок, пов'язаних із людським фактором, зменшує фінансові витрати на обслуговування мережі та підвищує її відмовостійкість.

Важливою особливістю програми є те, що обмеження швидкості застосовується лише тоді, коли рівень завантаженості каналу перевищує заданий поріг, а також якщо частка непріоритетного трафіку є значною. Це дозволяє уникнути необґрунтованого обмеження швидкості та забезпечити резерв пропускної здатності для критичних сервісів, що мають безпосередній вплив на функціонування організації.

В умовах обмеженої доступності високошвидкісних каналів зв’язку в деяких регіонах механізм динамічного керування пропускною здатністю є оптимальним рішенням для балансування навантаження та ефективного використання мережевих ресурсів. При цьому слід відзначити, що введення обмежень не призводить до повної недоступності сервісів, а лише коригує їхню швидкість, що забезпечує стабільну та передбачувану роботу мережі.

2 ЕКОНОМІЧНИЙ РОЗДІЛ

2.1 Резюме

У процесі розробки дипломного проекту була створена модель оптимізації та безпеки передачі даних, що базується на механізмі NBAR. Метою розробки було підвищення ефективності використання мережевих ресурсів та зменшення впливу людського фактора на процес управління трафіком.

Впровадження цієї моделі має значний потенціал для організацій, які працюють в умовах обмеженої пропускної здатності каналів зв'язку, а також для підприємств, що прагнуть автоматизувати управління мережею та забезпечити стабільну роботу критично важливих сервісів.

Оцінка якості розробленого програмного забезпечення базується на аналізі його ефективності, зручності для користувачів, а також відповідності встановленим вимогам. Важливим аспектом є також оцінка вартості розробки, що включає трудозатрати та фінансові витрати.

Запропонована модель не лише підвищує безпеку передачі даних, а й оптимізує процес адміністрування мережі, що сприяє зниженню витрат та покращенню продуктивності інформаційної інфраструктури організацій.

2.2 Визначення трудомісткості розробки програмного забезпечення

Тривалість створення програмного продукту визначається його масштабом, рівнем трудомісткості, кваліфікацією виконавців та встановленими ринковими термінами. Використовуючи метод структурної аналогії та аналіз відповідних каталогів аналогічного ПЗ, можна встановити обсяг програмного засобу, що виражається у тисячах умовних машинних команд для аналога

Таблиця 2.1. Каталог аналогів

Найменування ПП	**Обсяг функції ПП – V_o, усл. машинних командах.**
1. ПП автоматизації засобів по каталогу	680 – 7000
2. ПП автоматизованих розрахунків	1300 – 8600
3. ПП СУБД	2500 – 9800

У таблиці 2.1 представлені аналоги програмного забезпечення, функції яких, у більшому або меншому ступені, виконує розроблений програмний продукт. Для нашого варіанта виділено сірим кольором.

Вибравши аналог ПЗ, що містить Vo в умовних машинних командах, трудомісткості визначати на основі табл.2.2

Таблиця.2.2. Норма часу

Обсяг ПЗ, тис.умов.машинних команд	**Норма часу, люд/год**
1.00	229
2.00	244
3.00	262

На підставі отриманого значення, по довіднику, визначається укрупнена норма часу на розробку аналога програмного забезпечення (коректується поправочним коефіцієнтом враховуючої умови розробки ПП, тобто в умовах комп'ютера, Кк=0,7÷0,8): Тар = 229 х 0,8 = 183,2 (люд/годин).

Трудомісткість програмного продукту визначається окремо для кожного етапу розробки, виходячи з показників трудомісткості відповідного аналога. При цьому враховують рівень складності розробки, ступінь інноваційності та частку використання стандартних модулів. Розрахунки проводяться згідно з наступними формулами:

$$T_{ТЗ} = T^a p \times L_1 \times K_H \qquad (2.1)$$

$$T_{ТП} = T^a p \times L_2 \times K_H \qquad (2.2)$$

$$T_{РП} = T^a p \times L_3 \times K_H \times K_T \qquad (2.3)$$

Для розрахунку необхідні наступні коефіцієнти:

Li – питома вага і-го етапу розробки (див. табл. 2.3.);

Кн – поправочний коефіцієнт, що враховує ступінь новизни (див. табл. 2.4.);

Кт – поправочний коефіцієнт, що враховує ступінь використання в розробці типових програм (див. табл. 2.5)

Таблиця 2.3. Значення питомих коефіцієнтів трудомісткості стадії в загальній трудомісткості розробки ПП

Код стадії	**Ступінь новизни**		
	А	Б	В
ТЗ (L_1)	0,15	0,12	0,12
ТП (L_2)	0,16	0,15	0,11
РП (L_3)	0,55	0,58	0,61

Для нашого варіанта виділено сірим кольором.

Таблиця 2.4. Значення поправочного коефіцієнта, що враховує ступінь новизни

Код ступеня новизни	**Ступінь новизни**	**Значення $К_н$**
А	Принципово нові ПП	1,75 – 1,2
Б	ПП – розвиток визначеного параметричного ряду	1,0 – 0,8
В	ПП маючий аналог	0,7

Для нашого варіанта виділено сірим кольором.

Таблиця 2.5. Значення коефіцієнта ступеня використання в розробці типових програм

Ступінь охоплення реалізованих функцій розроблювального ПП типовими програмами, %	**Значення $К_т$**
60 і вище	0,6
40-60	0,7
20-40	0,8
До 20	0,9

Для нашого варіанта виділено сірим кольором.

Тепер розраховуємо трудомісткість по кожному етапу окремо:

Трудомісткість технічного завдання

$$Т_{тз}=Т^а * L_1 * К_н = 183,2 * 0,12 * 0,7 = 15,38 \text{ (люд/годин)} \quad (2.4)$$

Трудомісткість розробки технічного проекту

$$Т_{тп}=Т^а * L_2 * К_н = 183,2 * 0,11 * 0,7 = 14,11 \text{ (люд/годин)} \quad (2.5)$$

Трудомісткість розробки робочого проекту

$$Т_{рп}=Т^а * L_3 * К_н * К_т = 183,2 * 0,61 * 0,7 * 0,6 = 46,94 \text{ (люд/годин)} \quad (2.6)$$

Для подальших розрахунків визначили кількість папера, витраченого на кожен етап: технічне завдання $N_{тз}$= 2 (стр), розробка ТП $N_{тп}$=26 (стр), розробка робочого проекту $N_{рп}$=6(стр), пояснювальна записка відповідно $N_{пз}$=22(стр) Розрахунок зведений у таблицю 2.6.

Таблиця 2.6. Розрахунок трудомісткості ПП

Найменування етапів	**Розрахунок, годин**		
1.ТЗ	$T_{Ртз}$=15,38	$T_{кк}=0,7*N_{тз}$= 0,7*3=2,1	$T_{нк}=0,15*N_{тз}=0,15*3=0,45$
2.Розробка ТП	$T_{Ртп}$=14,11	$T_{кк}=0,7*N_{тп}=0,7*28=1$ 9,6	$T_{нк}=0,15*N_{тп}=0,15*28=4,2$
3.Розробка РП	$T_{Ррп}$= 46,94	$T_{кк}=0,7*N_{рп}=0,7*13=$ 9,1	$T_{нк}=0,15*N_{рп}=0,15*13=1,9$ 5
4.Розробка ПЗ	$T_{пз}=1,5*N_{пз}=$ 1,5*29 =43,5	$T_{кк}=0,7*N_{тз}=0,7*29=2$ 0,3	$T_{нк}=0,15*N_{пз}=0,15*29$ =4,35
Усього, в т.ч.:	181,98		
- на розробку	$\Sigma T_р$=119,93		
- контроль керівника		$\Sigma T_{кк}$=51,1	
- нормоконтроль			$\Sigma T_{нк}$=10,95

2.3 Розрахунок ціни програмного продукту

Для оцінки вартості програмного продукту розглядаються основна заробітна плата виконавців, матеріальні витрати та загальні витрати на розробку ПЗ. Детальний розрахунок основної заробітної плати наведено у таблиці 2.7. Згідно зі статтею 8 «Закону про Державний бюджет України на 2025» з 1 січня 2025 року встановлено мінімальну місячну заробітну плату у розмірі 8000 гривень, а також мінімальну погодинну тарифну ставку – 48,00 грн.

Таблиця 2.7. Розрахунок основної заробітної плати виконавців

Найменування робіт	**Трудомісткість робіт, години**	**Погодинна тарифна ставка, грн.**	**Розрахунок, грн.**
1.Розробка ПП	119,93	48,00	5756,64
2.Контроль керівника	51,1	100,00	5110,00
3.Нормоконтроль	10,95	105,00	1149,75
Усього	-	-	ΣЗо= 12016,39

Зробимо розрахунок матеріальних витрат на розробку ПП. Розрахунок зведемо в таблицю 2.8.

Таблиця 2.8. Розрахунок матеріальних витрат на розробку ПЗ

Найменування матеріальних витрат	**Тип, модель**	**Кількість**	**Ціна одиниці, грн.**	**Вартість, грн.**
Папір	Лист А4	73	5.0	365,00
Разом	-	-	-	$В_{м\,і}$=365,00
Транспортно – заготівельні Витрати (10%)				$Втр\_з = 0{,}1 \times Вм1$= 0,1*280=28,0
Усього				$В_м=В_{мі}+В_{тр\_з=}$ 308,00

На підставі отриманих даних по окремих статтях витрат складена калькуляція планової собівартості в цілому ПП за формою, приведеною в таблиці 2.9.

Таблиця 2.9. Розрахунок статей витрат планової собівартості

Стаття витрат	**Значення, грн.**	**Формула розрахунку**
1. Матеріали	401,50	$В_м$ (див. табл. 2.8.)
2. Основна заробітна плата	12016,39	$З_о$ (див. табл. 2.7.)
3.Додаткова заробітна плата	1201,64	$Зд = 0{,}1 \times Зо$ = 12016,39*0,1
4.Відрахування до єдиного фонду соціального внеску	2907,96	$Вє.с.в. = 0{,}22 \times (Зо + Зд)$ =0,22* (12016,39+1201,64)
5. Накладні витрати	4806,56	$Внак. = 0{,}4 \times Зо$ =0,4 * 12016,39
6. Повна собівартість	21334,05	$С_{пов}$= $В_м$ +$З_о$ + $З_д$ +Вє.с.в. + Внак.= 401,50+12016,39+1201,64+2907,96 +4806,56

Розмір прибутку, що включається в ціну, визначаємо по наступній формулі:

$П=(С_{пов}*Р)/100=(21334,05*15)/100=3200,11$ грн (2.7)

Де р – плановий рівень рентабельності (10-15%).

Оптова ціна (кошторисна вартість) визначається по формулі:

$Ц_о = С_{пов}+П = 21334,05+3200,11 = 24534,16$ грн; (2.8)

Виходячи з отриманих даних, ціна реалізації розробленого програмного забезпечення становитиме:

$Ц_р=Ц_о+ПДВ = = 24534,16 +24534,16 *0.2 = 29440,99$ грн; (2.9)

РОЗДІЛ ОХОРОНИ ПРАЦІ І ТЕХНІКИ БЕЗПЕКИ

Однією із характерних особливостей сучасного розвитку суспільства є зростання сфер діяльності людини, в яких широко використовуються інформаційні технології. Активне використання персональних комп'ютерів та мережевих технологій призвело до необхідності удосконалення систем передачі даних, зокрема щодо їхньої оптимізації та безпеки.

Механізм NBAR (Network-Based Application Recognition) відіграє важливу роль у сучасних методах контролю та управління мережевим трафіком. Завдяки своїй здатності розпізнавати специфічні програми та протоколи, NBAR сприяє ефективному розподілу ресурсів, зменшенню затримок у передачі даних та підвищенню рівня мережевої безпеки.

3.1 Аналіз небезпечних і шкідливих факторів, що впливають на програміста

У контексті розробки та впровадження механізмів оптимізації передачі даних, програмісти не лише працюють з алгоритмами та кодом, а й стикаються з низкою виробничих факторів, які можуть негативно впливати на їхнє здоров'я та продуктивність. Окрім традиційних ризиків, таких як підвищений рівень шуму, недостатня освітленість або електромагнітне випромінювання, важливим аспектом є кібербезпека.

Робота з механізмом NBAR та іншими технологіями мережевого захисту передбачає постійний аналіз даних та моніторинг загроз, що може призводити до психологічної перевантаженості та емоційної напруги. У зв'язку з цим необхідно розробляти та впроваджувати методи захисту не тільки мережевих систем, але й самого працівника, зокрема через ергономічне облаштування робочого місця та дотримання режиму праці та відпочинку.

3.2 Гігієнічні вимоги до виробничого середовища

На робочому місці програміста повинні бути створені умови для безпечної та високопродуктивної праці. Це передбачає відповідність робочого середовища низці гігієнічних вимог, що впливають на фізичний та психологічний комфорт

працівника.

Освітлення: Робоче місце повинно мати достатнє та рівномірне освітлення, щоб зменшити навантаження на зір програміста. Бажано використовувати природне світло або комбіновані джерела освітлення.

Мікроклімат: Температура в приміщенні повинна бути комфортною (18–24°C), а рівень вологості підтримуватися у межах 40–60%. Необхідна також регулярна вентиляція для підтримки якісного повітряного обміну.

Ергономічність робочого місця: Робочий стіл і крісло мають відповідати принципам ергономіки, що сприятиме зменшенню ризиків розвитку захворювань опорно-рухового апарату.

Рівень шуму: Оптимальний рівень фонових шумів не повинен перевищувати 50–60 дБ. Надмірний шум може призводити до стресу, зниження концентрації та продуктивності.

Електромагнітні випромінювання: Використання захисних фільтрів на екранах, правильне розташування електронних пристроїв та дотримання правил електробезпеки допомагають знизити негативний вплив випромінювання.

Також важливо врахувати психоемоційний комфорт програміста: забезпечення можливості регулярного відпочинку, зручного простору для релаксації та достатнього фізичного руху протягом робочого дня.

Робоче місце програміста має бути організоване відповідно до ергономічних стандартів, забезпечуючи комфортні умови праці, які сприяють збереженню здоров’я та підвищенню продуктивності.

Регулювання меблів – робочий стіл та крісло повинні мати можливість індивідуального налаштування відповідно до росту працівника, щоб підтримувати правильну поставу та зменшити навантаження на м’язи та суглоби.

Розташування монітора – екран має бути встановлений так, щоб його верхня межа знаходилася на рівні очей, на відстані 60–90 см, оптимально – близько 70 см. Частота оновлення зображення повинна бути не менше 70 Гц, що допомагає уникнути зайвої втоми очей.

Освітлення – робоче місце рекомендується розташовувати перпендикулярно до вікон, щоб мінімізувати відблиски на екрані. Освітлення має бути рівномірним та достатнім, що сприяє зменшенню навантаження на зір.

Оздоблення робочої поверхні – стіл повинен бути пофарбований у матові кольори, щоб уникнути небажаних відблисків.

Працівники, що працюють з відеодисплейними терміналами (ВДТ), проходять попередні медичні огляди при працевлаштуванні, а також періодичні огляди впродовж трудової діяльності, відповідно до наказу Міністерства охорони здоров'я України № 45.

При оцінці придатності до роботи з ВДТ враховуються:

- Гострота зору
- Параметри рефракції
- Стан бінокулярного апарату ока
- Загальний стан здоров'я

Дотримання цих вимог у поєднанні з профілактичними заходами сприяє підтримці здоров'я працівників та ефективності їхньої роботи.

Електроустановки повинні відповідати нормативним вимогам, передбаченим Правилами улаштування електроустановок (ПУЕ), Правилами технічної експлуатації споживачів (ПТЕ), Правилами техніки безпеки під час експлуатації електроустановок (ПТБ) та іншими регламентованими документами.

Основні правила експлуатації електропроводки:

З'єднання, розгалуження та закінчення електропроводів і кабелів необхідно виконувати виключно за допомогою зварювання, паяння, опресування або спеціальних затисків. Використання скручування жил проводів категорично забороняється.

Заборонені дії при роботі з електрообладнанням:

Прокладка кабелів через складські приміщення, пожежонебезпечні або вибухонебезпечні зони.

Експлуатація проводів із пошкодженою ізоляцією, що може призвести до короткого замикання та виникнення пожежонебезпечної ситуації.

Залишення під напругою неізольованих дротів та кабелів, що становить небезпеку для персоналу.

Використання саморобних подовжувачів, які не відповідають вимогам ПУЕ, що може призвести до аварій.

Застосування нестандартних електронагрівальних пристроїв для обігріву приміщень або використання ламп розжарювання як засобу обігріву.

Експлуатація пошкоджених електричних розеток, вимикачів та інших приладів, що може викликати несправності або електротравми.

Використання телефонного чи радіопроводу як електромережевого кабелю, що не відповідає вимогам безпеки.

Залишення електрообладнання під напругою без нагляду, що може спричинити перегрів або коротке замикання.

3.3 Пожежна безпека

Робоче приміщення, що відповідає вимогам ПБЕ та ОНТП 24–86 у сфері вибухово-пожежної безпеки, класифікується як об’єкт категорії «В».

Основними потенційними причинами виникнення пожежі в такому приміщенні є:

1. Коротке замикання електропроводки;

2. Використання побутових електрорадіоприладів;

3. Недотримання встановлених норм протипожежного захисту.

Відповідно до ПУЕ, для зниження ризику виникнення пожежі необхідно забезпечити комплекс заходів, зокрема: ретельну ізоляцію всіх струмоведучих проводів, що підключені до робочих місць, регулярний огляд та перевірку стану їх ізоляції, а також суворе дотримання норм безпечної експлуатації обладнання.

Для гасіння пожеж на робочому місці користувача ПК застосовують як вуглекислотні, так і порошкові вогнегасники.

– Вуглекислотні вогнегасники випускаються у варіанті ручних пристроїв (наприклад, ВВК-5);

– Порошкові вогнегасники представлені моделями ВП-2, ВП-5, ВП-10 та іншими

Рисунок 3.1. Засоби пожежогасіння

З метою своєчасного оповіщення, на дільниці необхідно встановити протипожежну сигналізацію. Проходи та запасні виходи повинні бути вільними. Пожежний щит повинен розміщуватись в доступному місці та містити первинні засоби пожежогасіння (вогнегасник, лопату, відро, простирадло, ящик з піском)

ВИСНОВКИ

У ході виконання дипломного проекту було розроблена модель оптимізації та безпеки передачі даних за допомогою механізму NBAR і відповідне програмне забезпечення. Основною метою проекту було створення ефективного рішення, яке дозволяє зменшити навантаження на мережеві канали, забезпечити стабільність і безпеку передачі даних та автоматизувати процес обмеження швидкості для непріоритетних типів трафіку.

У процесі розробки виконано аналіз існуючих методів управління мережевим трафіком. Було вивчено механізми розпізнавання та класифікації трафіку, зокрема, використання Network-Based Application Recognition (NBAR), а також традиційні підходи до керування пропускною здатністю. Розроблено алгоритм динамічного керування трафіком. Алгоритм дозволяє у реальному часі аналізувати потік даних у мережі, класифікувати його за допомогою NBAR та, у разі перевищення встановлених порогових значень, автоматично застосовувати політики обмеження швидкості. Розроблено програмне забезпечення з використанням мови Java, що забезпечує кросплатформність та сумісність з різними операційними системами. Використано Apache Maven для керування залежностями та збірки проекту, а також інтегроване середовище розробки IntelliJ IDEA. Розроблено та протестовано конфігураційний файл у форматі YAML, який дозволяє гнучко налаштовувати параметри роботи програми, такі як частота оновлення даних, граничні значення навантаження, список дозволених протоколів тощо.

Виконано тестування програмного забезпечення в реальних умовах. Для тестування використовувався маршрутизатор та міжмережевий екран. Було продемонстровано, що програма успішно виявляє надмірне використання трафіку та застосовує політики обмеження без негативного впливу на критичні сервіси. Оцінено ефективність запропонованого рішення. Результати тестів показали, що:

- швидкість передачі даних для HTTP-трафіку була обмежена, при цьому критично важливі сервіси (SNMP, DNS, ICMP) продовжували працювати без змін;
- навантаження на мережевий канал після активації програми зменшилося в

середньому на 30%, що дозволило забезпечити стабільну роботу пріоритетного трафіку;

- час, необхідний для конфігурації обмежень адміністратором вручну, був скорочений у 3 рази завдяки автоматизації процесу.

Таким чином, розроблене програмне забезпечення дозволяє підвищити ефективність використання мережевих ресурсів, мінімізувати вплив людського фактора на процес управління трафіком та знизити витрати на адміністрування мережі. Впровадження цієї системи доцільне для організацій, що працюють в умовах обмеженої пропускної здатності каналів зв'язку, а також для підприємств, які прагнуть автоматизувати процеси управління мережею та забезпечити стабільну роботу критичних сервісів.

Подальший розвиток роботи може включати:

– реалізацію графічного інтерфейсу для спрощення налаштувань та моніторингу стану мережі;

– розширення підтримуваних протоколів моніторингу (наприклад, NetFlow, sFlow) для більш детального аналізу мережевого трафіку;

– інтеграцію із сучасними системами SIEM для виявлення аномального трафіку та підвищення рівня безпеки мережі.

Розроблене рішення демонструє високу ефективність та має потенціал для подальшого вдосконалення відповідно до потреб сучасних корпоративних мереж.

ПЕРЕЛІК ВИКОРИСТАНИХ ІНФОРМАЦІЙНИХ ДЖЕРЕЛ

1. Танасійчук С. О. Комп’ютерні мережі: теорія та практикум: навч. посіб. – Львів: Львівська політехніка, 2021. – 368 с.
2. Кузьменко С. В. Адміністрування комп’ютерних мереж: навч. посіб. – Харків: ХНУРЕ, 2020. – 284 с.
3. Білоус О. В. Комп’ютерні мережі та безпека інформації – Київ: Наукова думка, 2022. – 310 с.
4. Пономаренко В. М. Сучасні методи оптимізації комп’ютерних мереж: монографія – Харків: УкрІНТЕІ, 2023. – 278 с.
5. Мельник В. В. Мережеві технології та протоколи: навчальний посібник – Одеса: ОНПУ, 2019. – 296 с.
6. Ковальчук Ю. П. Основи інформаційної безпеки: навчальний посібник / Ю. П. Ковальчук, Л. І. Шевченко. – Київ: НТУУ «КПІ», 2021. – 342 с.
7. Олійник В. В. Оптимізація роботи інформаційних систем: навчальний посібник / В. В. Олійник. – Київ: КНУ, 2020. – 280 с.
8. Семенов А. П. Мережева безпека та кіберзахист: монографія / А. П. Семенов, В. С. Іваненко. – Харків: ХНЕУ, 2023. – 350 с.
9. Мартиненко П. Г. Безпека інформаційних систем: навчальний посібник – Дніпро: ДНУ, 2019. – 320 с.
10. Степаненко О. Ю. Протоколи SNMP і їх використання в мережевому моніторингу – Вінниця: ВНТУ, 2022. – 276 с.
11. Cisco Networking Academy. Основи маршрутизації та комутації. Навчальний курс. – [Електронний ресурс]. – Режим доступу: https://www.netacad.com – Дата звернення: 12.03.2025.
12. RFC 7854. Cisco's NetFlow Version 9 Protocol. – [Електронний ресурс]. – Режим доступу: https://tools.ietf.org/html/rfc7854 – Дата звернення: 09.03.2025.
13. Cisco Systems. Посібник користувача для механізму NBAR. – [Електронний ресурс]. – Режим доступу: https://www.cisco.com – Дата звернення: 15.03.2025.

ДОДАТОК А. Лістинги класів main, ClassMap, Traffic, PolicyMap

```
/**
 * Головний клас для виконання програми
 */
public class Main {
    /**
     * Головний метод.
     * @param args аргументи з командного рядка.
     * @throws Exception якщо сталася помилка під час виконання програми.
     */
    public static void main(String[] args) throws Exception {
        System.out.println("Привіт!");
        Config cfg = new Config("./config.yaml");
    }
}

/**
 * Клас для опису Class Map (CMAP).
 */
public class ClassMap {
    /**
     * Критерій відповідності. match-all або match-any.
     */
    String match;
    /**
     * Назва CMAP.
     */
    String name;
    /**
     * Назва протоколу, що блокується.
     */
    String protocol;
    /**
     * Конструктор для CMAP.
     * @param match критерій відповідності. match-all або match-any.
     * @param name назва CMAP.
     * @param protocol назва протоколу, що блокується.
     */
    public ClassMap(String match, String name, String protocol) {
        if (match != null) this.match = match;
        else this.match = "all";
        this.name = name;
        this.protocol = protocol;
    }
    /**
     * Метод для отримання критерію відповідності.
     * @return критерій відповідності (match-all або match-any).
     */
    public String getMatch() {
        return match;
    }

    /**
     * Метод для отримання назви CMAP.
     * @return назва CMAP.
     */
    public String getName() {
        return name;
    }
    /**
     * Метод для отримання назви протоколу, що блокується.
     * @return назва протоколу.
     */
    public String getProtocol() {
        return protocol;
    }
}
```

```
/**
 * Клас для збереження даних про трафік у поточний момент часу.
 */
public class Traffic {
    /**
     * Унікальний ідентифікатор для протоколу або застосунку,
     * який розпізнає NBAR.
     */
    int cnpdAllStatsProtocolsIndex;
    /**
     * Назва протоколу або застосунку, який розпізнає NBAR.
     */
    String cnpdAllStatsProtocolsName;
    /**
     * Кількість отриманих пакетів.
     */
    long cnpdAllStatsInPkts;
    /**
     * Кількість відправлених пакетів.
     */
    long cnpdAllStatsOutPkts;
    /**
     * Обсяг отриманих байтів.
     */
    long cnpdAllStatsInBytes;
    /**
     * Обсяг відправлених байтів.
     */
    long cnpdAllStatsOutBytes;
    /**
     * Кількість отриманих пакетів (64-бітова версія).
     */
    long cnpdAllStatsHCInPkts;
    /**
     * Кількість відправлених пакетів (64-бітова версія).
     */
    long cnpdAllStatsHCOutPkts;
    /**
     * Обсяг отриманих байтів (64-бітова версія).
     */
    long cnpdAllStatsHCInBytes;
    /**
     * Обсяг відправлених байтів (64-бітова версія).
     */
    long cnpdAllStatsHCOutBytes;
    /**
     * Вхідна швидкість передачі даних (біт/с).
     */
    long cnpdAllStatsInBitRate;
    /**
     * Вихідна швидкість передачі даних (біт/с).
     */
    long cnpdAllStatsOutBitRate;
    /**
     * Поточний час.
     */
    long Time;

    /**
     * Конструктор об'єкта трафіку.
     * @param cnpdAllStatsProtocolsIndex унікальний ідентифікатор протоколу або
застосунку.
     */
    public Traffic(int cnpdAllStatsProtocolsIndex) {
        this.cnpdAllStatsProtocolsIndex = cnpdAllStatsProtocolsIndex;
    }
    /**
     * Метод для отримання назви протоколу або застосунку.
     * @return назва протоколу або застосунку.
```

```
     */
    public String getCnpdAllStatsProtocolsName() {
        return cnpdAllStatsProtocolsName;
    }
    /**
     * Метод для отримання обсягу отриманих байтів.
     * @return кількість отриманих байтів.
     */
    public long getCnpdAllStatsInBytes() {
        return cnpdAllStatsInBytes;
    }
    /**
     * Метод для отримання поточного часу.
     * @return поточний час.
     */
    public long getTime() {
        return Time;
    }
    /**
     * Метод для встановлення поточного часу.
     * @param time поточний час.
     */
    public void setTime(long time) {
        Time = time;
    }
    /**
     * Метод для оновлення інформації в таблиці трафіку.
     * @param info рядок з інформацією.
     * @param Column стовпець у таблиці cnpdAllStatsTable.
     */
    public void addInfo(String info, int Column) {
        switch (Column) {
            case 1: cnpdAllStatsProtocolsIndex = Integer.parseInt(info); break;
            case 2: cnpdAllStatsProtocolsName = info; break;
            case 3: cnpdAllStatsInPkts = Long.parseLong(info); break;
            case 4: cnpdAllStatsOutPkts = Long.parseLong(info); break;
            case 5: cnpdAllStatsInBytes = Long.parseLong(info); break;
            case 6: cnpdAllStatsOutBytes = Long.parseLong(info); break;
            case 7: cnpdAllStatsHCInPkts = Long.parseLong(info); break;
            case 8: cnpdAllStatsHCOutPkts = Long.parseLong(info); break;
            case 9: cnpdAllStatsHCInBytes = Long.parseLong(info); break;
            case 10: cnpdAllStatsHCOutBytes = Long.parseLong(info); break;
            case 11: cnpdAllStatsInBitRate = Long.parseLong(info); break;
            case 12: cnpdAllStatsOutBitRate = Long.parseLong(info); break;
        }
    }
}
import java.util.ArrayList;

/**
 * Клас для опису Policy Map (PMAP).
 */
public class PolicyMap {
    /**
     * Назва PMAP.
     */
    String name;
    /**
     * Включений у PMAP Class Map (CMAP).
     */
    ClassMap classMap;
    /**
     * Обмеження пропускної здатності для протоколу в CMAP.
     */
    long shaper;
    /**
     * Час життя PMAP.
     */
    int TTL;
```

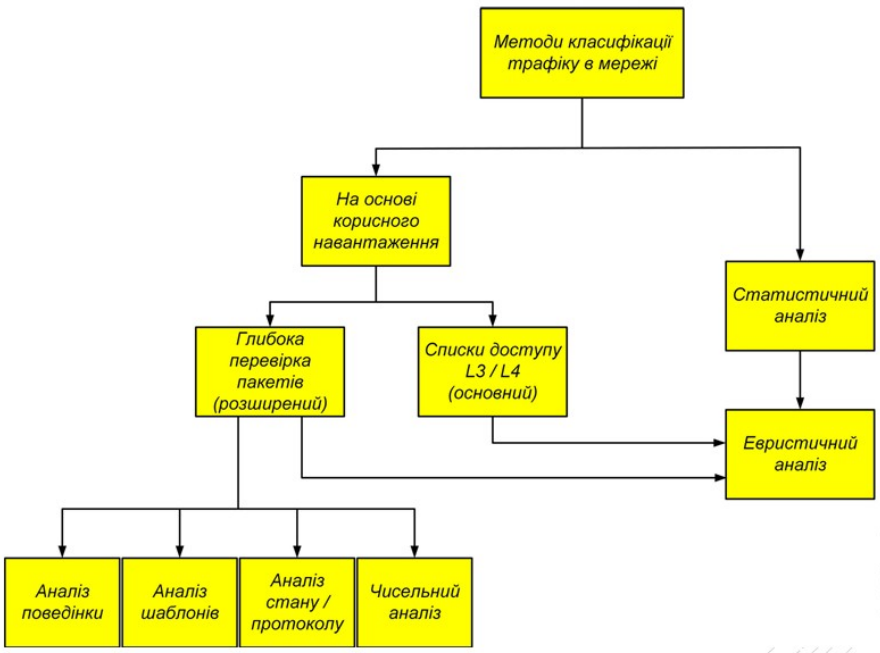
```
    /**
     * Список команд для скасування PMAP.
     */
    ArrayList<String> reverceCommands;
    /**
     * Метод для отримання списку команд для скасування PMAP.
     * @return список команд.
     */
    public ArrayList<String> getReverceCommands() {
        return reverceCommands;
    }
    /**
     * Метод для встановлення списку команд для скасування PMAP.
     * @param reverceCommands список команд.
     */
    public void setReverceCommands(ArrayList<String> reverceCommands) {
        this.reverceCommands = reverceCommands;
    }
    /**
     * Метод для отримання часу життя PMAP.
     * @return час життя.
     */
    public int getTTL() {
        return TTL;
    }
    /**
     * Метод для встановлення часу життя PMAP.
     * @param TTL час життя.
     */
    public void setTTL(int TTL) {
        this.TTL = TTL;
    }
    /**
     * Метод для отримання CMAP для PMAP.
     * @return CMAP.
     */
    public ClassMap getClassMap() {
        return classMap;
    }
    /**
     * Метод для отримання назви PMAP.
     * @return назва.
     */
    public String getName() {
        return name;
    }
    /**
     * Метод для отримання обмеження пропускної здатності PMAP.
     * @return обмеження пропускної здатності.
     */
    public long getShaper() {
        return shaper;
    }
    /**
     * Конструктор для Policy Map.
     * @param name назва PMAP.
     * @param classMap відповідний CMAP.
     * @param shaper обмеження пропускної здатності.
     */
    public PolicyMap(String name, ClassMap classMap, long shaper) {
        this.name = name;
        this.classMap = classMap;
        this.shaper = shaper;
    }
}
```

ДОДАТОК Б. Слайди мультимедійної презентації

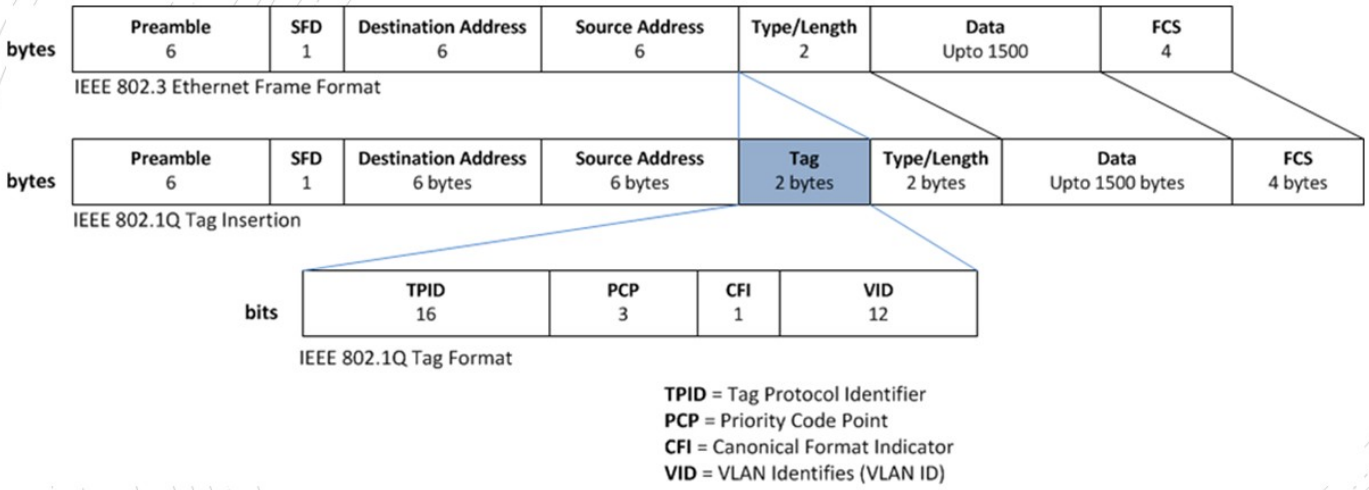
Розробка моделі оптимізації та безпеки передачі даних за допомогою механізму NBAR

Вишневський Андрій, 4КС-58

Методи класифікації трафіку комп'ютерної мережі



Вміст кадру Ethernet 802.1q



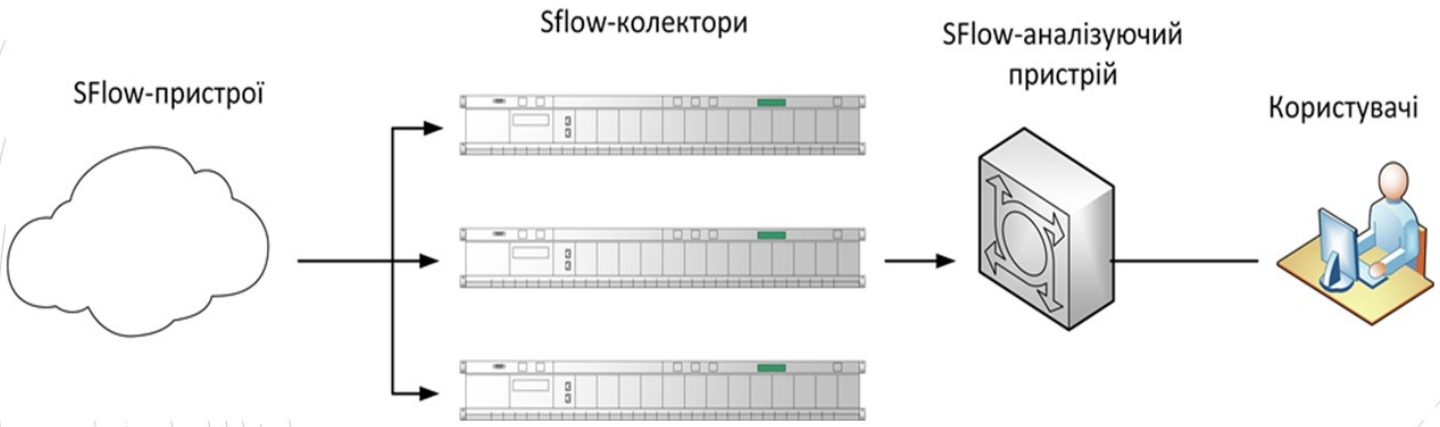
Типи трафіку у стандарті 802.1p

Тип трафіку	Клас трафіку	Пріоритет
Банкові транзакції, ігри тощо	Фон	1
Менше 10 мілісекунд затримки	Звук	2
Менше 100 мілісекунд затримки	Відео	3
Деякі важливі програми	Контрольований	4
Пріоритет для важливих користувачів	Пріоритетний	5
Пріоритет звичайної локальної мережі	Негарантована доставка	6
Критично важливий для мережі, трафік керування мережею	Мережевий контроль	7

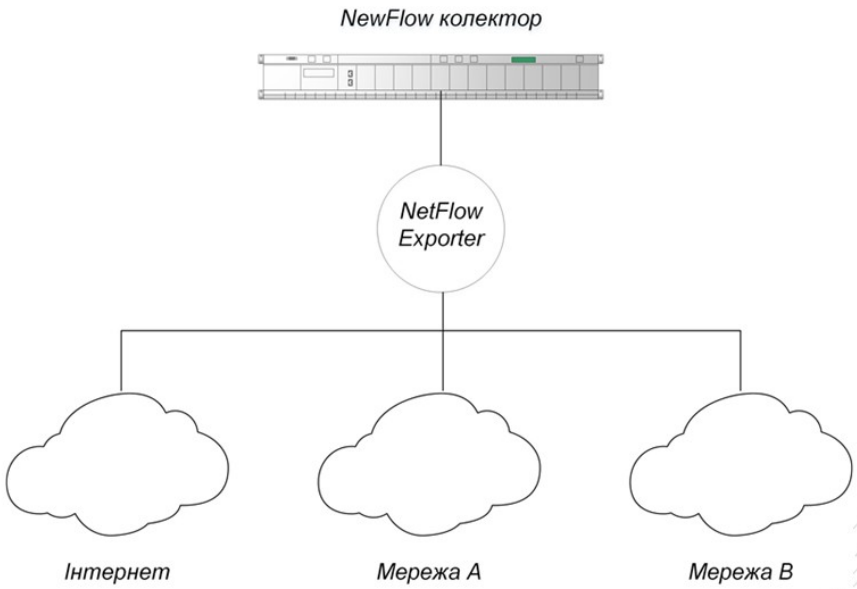
Структура IP-заголовку за стандартом 802.1p

0-3	4-7	8-15	16-33	
Версія	Довжина заголовка	Тип сервісу (TOS/DSCP)	Загальна довжина	
Ідентифікація			Флаг	Фрагмент
Час життя		Протокол	Контрольна сума заголовка	
Адреса джерела				
Адреса призначення				
Опції				

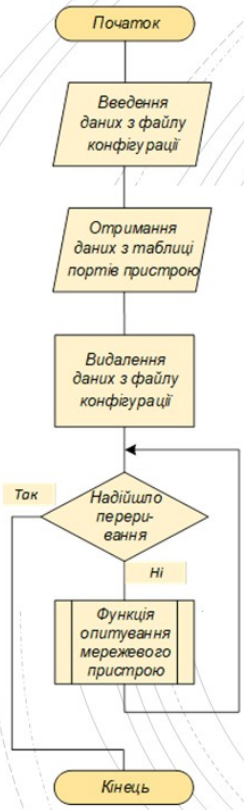
Організація роботи технології SFlow у мережі



Організація роботи технології NetFlow у мережі

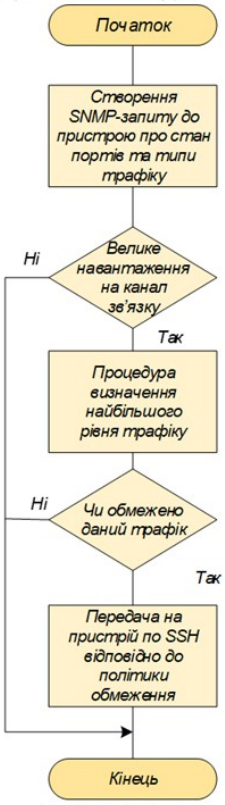


Блок-схема алгоритму оптимізації мережевого трафіку за допомогою механізму NBAR



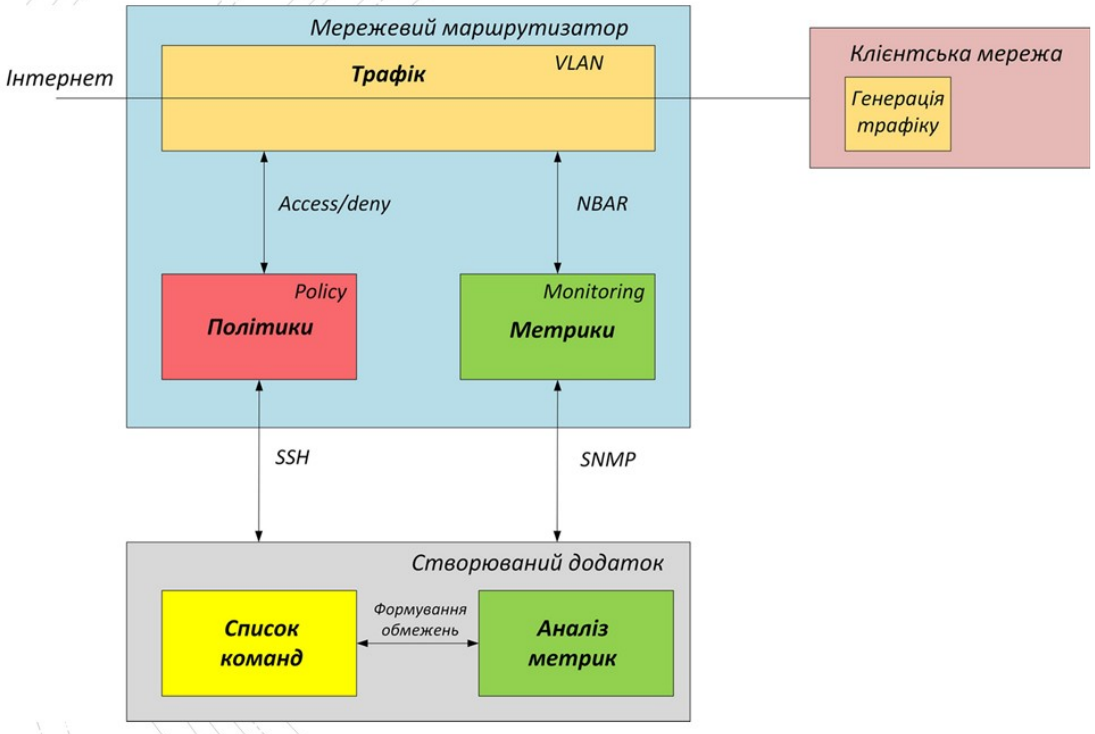
Блок-схема алгоритму читання таблиці портів маршрутизатору

Блок-схема алгоритму моніторингу навантаження каналу зв'язку

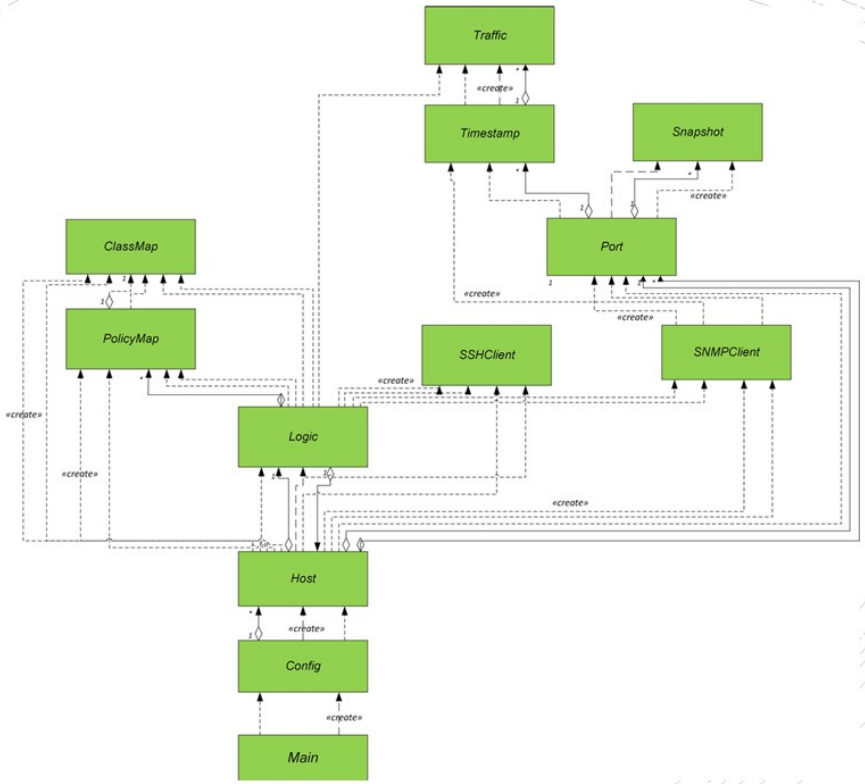


Блок-схема алгоритму скидання файлу конфігурації пристрою

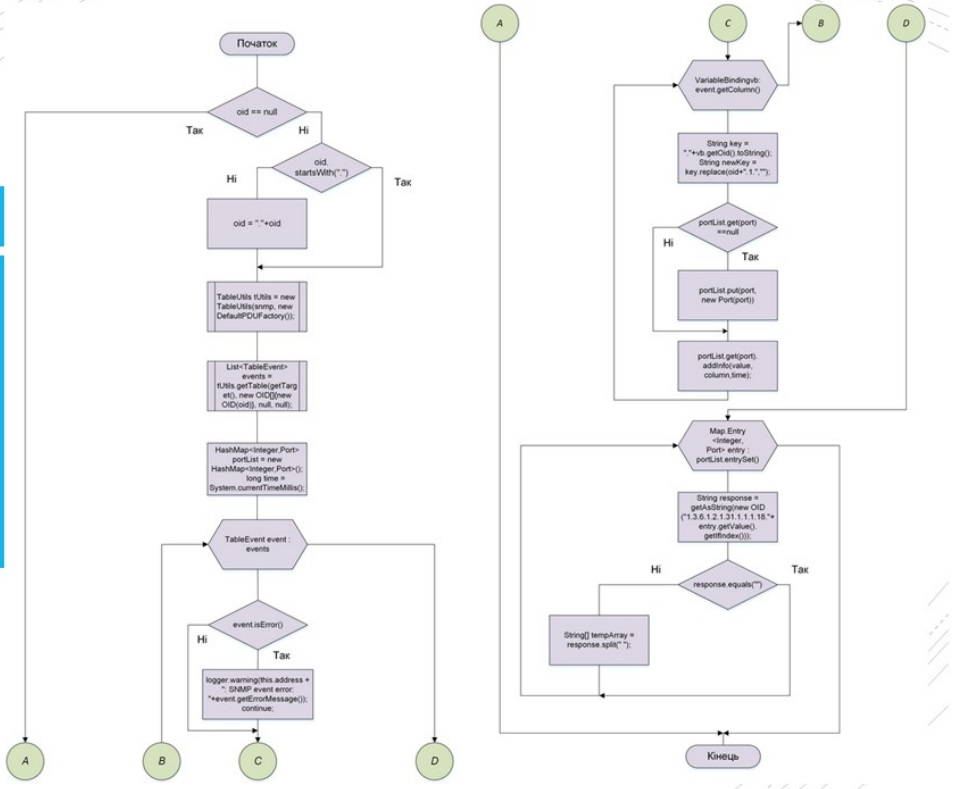
Структурна схема оптимізації та безпеки передачі даних



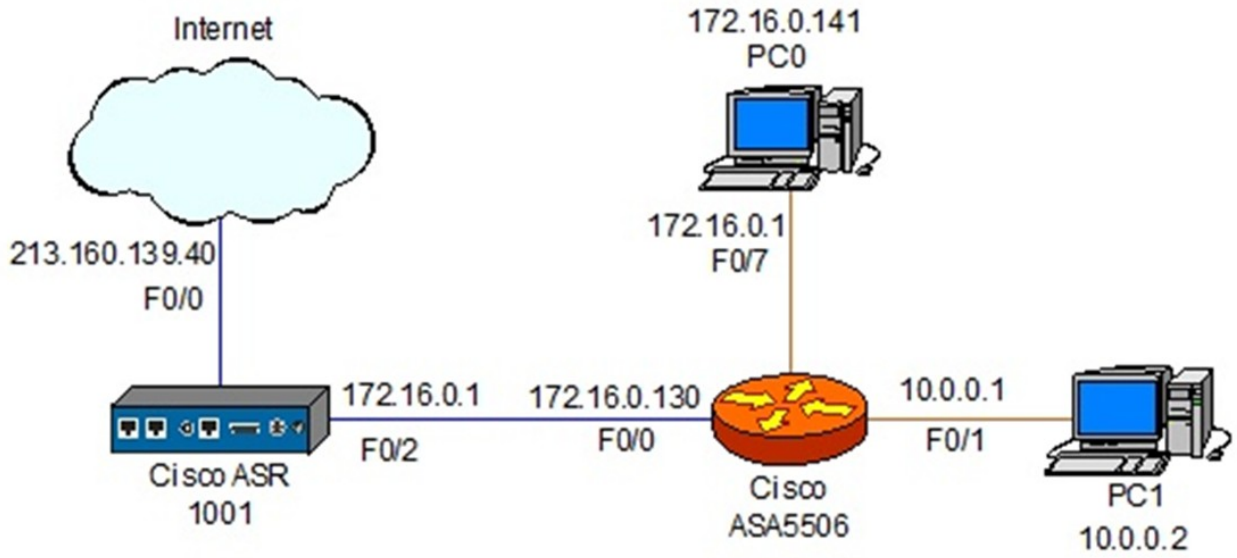
Функціональна схема діаграми класів у програмі



БСА розпізнавання потоку даних та управління політиками шейпінгу



Модель мережі для тестування розробленого ПЗ

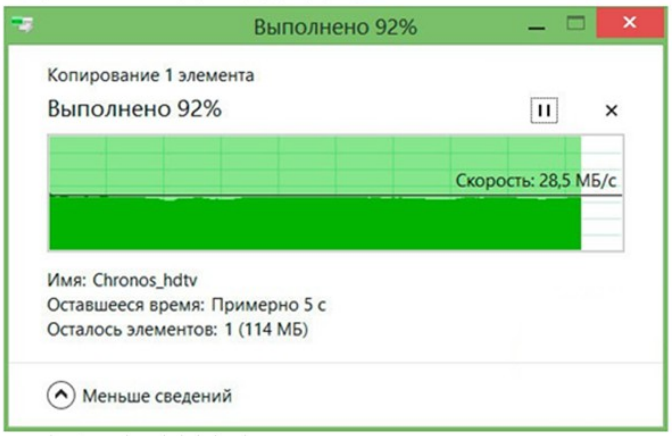


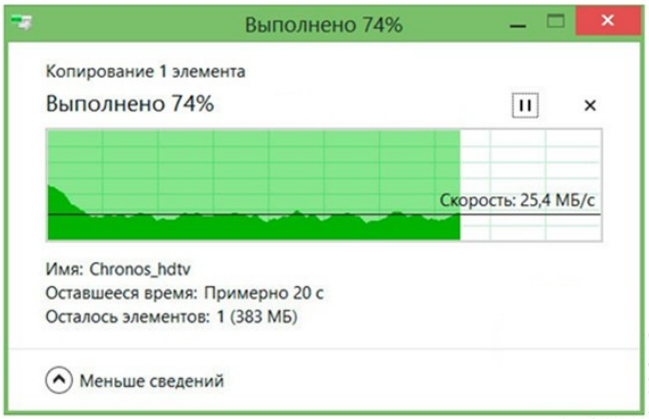
Лістинг сформованих команд та виведення логу SSH-з'єднання

```
Straight commands:
configure terminal
class-map match-all CMAP20190430193451
match protocol http
exit
policy-map PMAP20190430193451
class CMAP20190430193451
police 10000000 conform-action transmit exceed-action drop
exit
exit
interface FastEthernet0/1
service-policy output PMAP20190430193451
exit
ASR1001-X#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
ASR1001-X(config)#
ASR1001-X(config)#class-map match-all CMAP20190430193451
ASR1001-X(config-cmap)#
ASR1001-X(config-cmap)#match protocol http
ASR1001-X(config-cmap)#
ASR1001-X(config-cmap)#exit
ASR1001-X(config)#
```

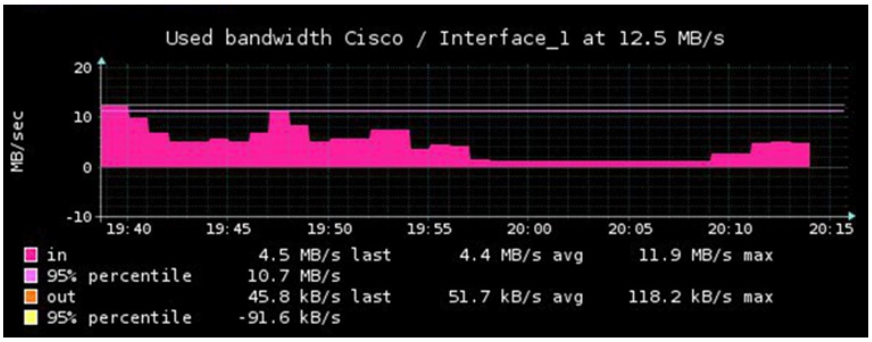
```
ASR1001-X(config)#policy-map PMAP20190430193451
ASR1001-X(config-pmap)#
ASR1001-X(config-pmap)#class CMAP20190430193451
ASR1001-X(config-pmap-c)#
ASR1001-X(config-pmap-c)#$0000 conform-action transmit exceed-action drop
ASR1001-X(config-pmap-c-police)#
ASR1001-X(config-pmap-c-police)#exit
ASR1001-X(config-pmap-c)#
ASR1001-X(config-pmap-c)#exit
ASR1001-X(config-pmap)#
ASR1001-X(config-pmap)#interface FastEthernet0/1
ASR1001-X(config-if)#
ASR1001-X(config-if)#service-policy output PMAP20190430193451
ASR1001-X(config-if)#
ASR1001-X(config-if)#exit
ASR1001-X(config)#
ASR1001-X(config)#exit
ASR1001-X#
ASR1001-X#exit-status: 0
```

Оцінка швидкості копіювання даних користувача





Визначення навантаження на канал зв’язку в мережі



ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

РЕЦЕНЗІЯ

на дипломний проект здобувача (здобувачки) освіти

відділення комп'ютерних систем

Вишневського Андрія Олександровича

(прізвище, ім`я та по батькові)

Спеціальність *123 «Комп'ютерна інженерія»*

Освітньо-професійна програма *«Безпека комп'ютерних систем і мереж»*

Керівник дипломного проекту (роботи) *Скорняков В'ячеслав Сергійович*

(прізвище, ім`я та по батькові)

Тема дипломного проекту (роботи) *Розробка моделі оптимізації та безпеки передачі даних за допомогою механізму NBAR*

Обсяг розрахунково-пояснювальної записки *83* сторінок

Обсяг графічної (презентаційної) частини *16* аркушів (слайдів)

ХАРАКТЕРИСТИКА ДИПЛОМНОГО ПРОЕКТУ (РОБОТИ)

а) заключення про ступінь відповідності виконаного дипломного проекту завданню

Представлений дипломний проект відповідає затвердженій темі та виконаний відповідно технічному завданню. Дипломний проект присвячений створенню моделі оптимізації та безпеки передачі даних за допомогою механізму NBAR і складається з пояснювальної записки та мультимедійної презентації з відповідними схемами.

б) характеристика виконання кожного розділу дипломного проекту

Пояснювальна записка складається з основного розділу (Огляд технологій аналізу мережевих потоків; Розробка алгоритмів оптимізації та безпеки передачі даних за допомогою механізму NBAR; Програмна реалізація алгоритмів оптимізації та безпеки передачі даних за допомогою механізму NBAR; Тестування моделі), економічного розділу, розділу охорони праці та додатків. Перелічені розділи поетапно охоплюють розробку, виконані докладно та обґрунтовано.

в) оцінка якості виконання пояснювальної записки та графічної частини дипломного проекту

Графічна частина складається з 16 слайдів мультимедійної презентації, виконаної у програмному продукті MS PowerPoint, які містять структурні та функціональні схеми, діаграми та скріншоти, блок-схеми алгоритмів, передбачені технічним завданням. Пояснювальна записка виконана акуратно та у відповідності до норм. Якість виконання пояснювальної записки відмінна, розробку виконано у повному обсязі.

г) перелік позитивних якостей дипломного проекту

Виконано тестування розробленого програмного забезпечення на реальному обладнанні, що підтверджує його ефективність. Зміст добре організований, містить детальні пояснення алгоритмів, блок-схеми та коди програмних модулів. Включено розрахунок витрат на розробку, що показує економічну доцільність проекту.

д) основні недоліки дипломного проекту

Було б корисно розглянути альтернативні методи оптимізації трафіку та порівняти їхню ефективність.

Тестування проводилося на обмеженому наборі пристроїв, що не гарантує повної сумісності з різними мережевими топологіями.

Оцінка розрахункової частини *Добре*

Оцінка графічної частини *Відмінно*

Загальна оцінка *Добре*

Прізвище, ім'я, по батькові рецензента *к.т.н. Шибаєва Наталя Олегівна*

Місце роботи і посада рецензента *Національний університет «Одеська політехніка», доцент кафедри інформаційних технологій*

Підпис:

«*20*» *червня* *2025 р.*

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

ВІДГУК

керівника на дипломний проект здобувача (здобувачки) освіти

відділення комп'ютерних систем

Вишневського Андрія Олександровича

(прізвище, ім`я та по батькові)

Спеціальність: *123 "Комп'ютерна інженерія"*

Освітньо-професійна програма: *«Обслуговування комп'ютерних систем і мереж»*

Тема дипломного проекту: *Розробка моделі оптимізації та безпеки передачі даних за допомогою механізму NBAR*

ХАРАКТЕРИСТИКА ДИПЛОМНОГО ПРОЕКТУ

а) обсяг і якість виконання проекту (графічного матеріалу і розрахунково-пояснювальної записки) *Дипломний проект виконано відповідно технічному завданню. Пояснювальна записка містить 83 сторінки. У пояснювальній записці наведено етапи розробки моделі оптимізації та безпеки передачі даних за допомогою механізму NBAR, алгоритмічного та програмного забезпечення. Графічна частина складається з 16 слайдів мультимедійної презентації, які також містять креслення, передбачені технічним завданням. Якість виконання пояснювальної записки та графічної частини добра, розробку виконано в повному обсязі.*

б) самостійність роботи над проектом: *Протягом всього строку дипломного проектування та переддипломної практики здобувач освіти Вишневський А.О. поступово та послідовно виконував всі етапи розробки. Всі роботи здобувач освіти виконував самостійно, з оглядом на рекомендації керівника*

в) теоретична підготовка випускника (випускниці): *Здобувач освіти Вишневський А.О. під час роботи над дипломним проектом вивчив достатню кількість літературних джерел та матеріалів за даною тематикою.*

Вважаю, що теоретична підготовка дипломника добра і він готовий до захисту дипломного проекту

г) вміння розв'язувати виробничі та конструкторські питання

Під час дипломного проектування здобувач освіти Вишневський А.О. мав змогу самостійно приймати окремі рішення з реалізації принципової електричної схеми пристрою та показав вміння організовано працювати над поставленим завданням, складати програмне забезпечення, схеми та розрахунки за допомогою сучасних комп'ютерних програмних засобів та мов програмування, таких як Intellij IDEA та Java

Оцінка розрахункової частини *Добре*

Оцінка графічної частини *Відмінно*

Загальна оцінка *Добре*

Прізвище, ім'я, по батькові керівника дипломного проекту

Скорняков В'ячеслав Сергійович

Місце роботи і посада керівника дипломного проекту *ВСП «Одеський технічний фаховий коледж ОНТУ», викладач спецдисциплін циклової комісії комп'ютерних технологій та програмної інженерії»*

Підпис

«16» 06 2025 р.

ДОЗВІЛ
НА РОЗМІЩЕННЯ
ВИПУСКНОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ
(ДИПЛОМНОГО ПРОЕКТУ)
В ЕЛЕКТРОННОМУ РЕПОЗИТАРІЇ ВСП «ОТФК ОНТУ»

Ми, що нижче підписалися,

Вишневський А.О.,
здобувач освіти гр. 4КС-58, та

Скорняков В.С.,
керівник дипломного проекту,

не заперечуємо щодо розміщення електронного варіанту пояснювальної записки до дипломного проекту фахового молодшого бакалавра на тему:

«Розробка моделі оптимізації та безпеки передачі даних за допомогою механізму NBAR» (автор роботи – Вишневський А.О., керівник роботи – Скорняков В.С.)

виконаного у ВСП «Одеський технічний фаховий коледж Одеського національного технологічного університету» в 2025 році, у повному обсязі в електронному репозитарії ВСП «ОТФК ОНТУ» для вільного доступу через мережу Інтернет.

Несемо відповідальність за ідентичність електронного та друкованого варіантів випускної кваліфікаційної роботи і даємо згоду на обробку персональних даних.

Виконавець _____ / Вишневський А.О. /

Керівник _____ / Скорняков В.С. /

«16» червня 2025 р.

ДОВІДКА

циклової комісії КТ та ПІ
про допуск до захисту дипломного проєкту
здобувача (здобувачки) освіти IV курсу
відділення комп'ютерних систем групи 4КС-58

Вишневського Андрія Олександровича

на тему *Розробка моделі оптимізації та безпеки передачі даних за допомогою механізму NBAR*

Висновок відповідальної особи за проведення нормоконтролю:
пояснювальна записка до дипломного проєкту виконана з несуттєвими порушеннями ДСТУ та оформлена відповідно до вимог Положення про дипломне проєктування

(підпис) *16.06.2025* (дата) *Петрашова В.І.* (П.І.Б.)

Висновок відповідальної особи за перевірку роботи на наявність академічного плагіату *згідно звіту про перевірку від 28.05.2025 р. значення коефіцієнту подібності в роботі становить 18,26%, коефіцієнт цитування – 0,44%.*

(підпис) *16.06.2025* (дата) *Краснокутська К.Г.* (П.І.Б.)

Попередня експертиза (малий захист) дипломного проєкту

здобувача (здобувачки) освіти *Вишневського А.О.* (П.І.Б.)

проведена «*16*» *червня* 2025 р.

Висновки *Пояснювальна записка до дипломного проєкту виконана у повному обсязі. Випускна кваліфікаційна робота (дипломний проєкт) відповідає вимогам Положення про дипломне проєктування та рекомендована до захисту.*

Голова ЦК КТ та ПІ (підпис) *Кривченко Ю.В.* (П.І.Б.)

StrikePlagiarism.com

Дата звіту 5/28/2025
Дата редагування 5/28/2025

Документ прийнятий

Звіт подібності

метадані

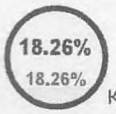
Назва організації
Odesa Technical Professional College of Odesa National University of Technology
Заголовок
Розробка моделі оптимізації та безпеки передачі даних за допомогою механізму NBAR

Автор
Вишневський Андрій Олександрович
Науковий керівник / Експерт
Скорняков В'ячеслав Сергійович

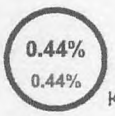
підрозділ
Відокремлений структурний підрозділ "Одеський технічний фаховий коледж Одеського національного технологічного університету"

Обсяг знайдених подібностей

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.



КП 1



КЦ

25	14083	117688
Довжина фрази для коефіцієнта подібності 2	Кількість слів	Кількість символів

Тривога

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про МОЖЛИВІ маніпуляції в тексті. Спотворення в тексті можуть мати навмисний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

Заміна букв	ß	20
Інтервали	A→	0
Мікропробіли	·	0
Білі знаки	¶	1
Парафрази (SmartMarks)	a	126

Подібності за списком джерел

Нижче наведений список джерел. В цьому списку є джерела із різних баз даних. Колір тексту означає в якому джерелі він був знайдений. Ці джерела і значення Коефіцієнту Подібності не відображають прямого плагіату. Необхідно відкрити кожне джерело і проаналізувати зміст і правильність оформлення джерела.

10 найдовших фраз — Колір тексту

ПОРЯДКОВИЙ НОМЕР	НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ)	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	https://card-file.ontu.edu.ua/server/api/core/bitstreams/ead3fa83-2e3d-4cd7-bfbd-1d5ed04c1ce4/content	103 0.73 %
2	Розробка програмної моделі оцінки рівня стійкості комп'ютерних систем і мереж [redacted] 5/25/2025 Odesa Technical Professional College of Odesa National University of Technology (Відокремлений структурний підрозділ "Одеський технічний фаховий коледж Одеського національного технологічного університету")	99 0.70 %

3	Алгоритм оптимізації мережевого трафіку [redacted] 3/15/2025 National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute (National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute)	96 0.68 %
4	https://card-file.ontu.edu.ua/bitstreams/53ed22ad-8700-4162-b97a-082a1ad472d6/download	83 0.59 %
5	https://card-file.ontu.edu.ua/bitstreams/5240e379-7721-49f0-8ee8-27140b0b473a/download	63 0.45 %
6	https://card-file.ontu.edu.ua/server/api/core/bitstreams/ead3fa83-2e3d-4cd7-bfbd-1d5ed04c1ce4/content	57 0.40 %
7	https://card-file.ontu.edu.ua/bitstreams/6cf43324-8f08-4031-ba42-f80b18efbbc8/download	52 0.37 %
8	https://card-file.ontu.edu.ua/server/api/core/bitstreams/ead3fa83-2e3d-4cd7-bfbd-1d5ed04c1ce4/content	43 0.31 %
9	https://card-file.ontu.edu.ua/bitstreams/bbed74c8-2ea7-44c5-8d00-0fe3fd9790ee/download	43 0.31 %
10	https://card-file.ontu.edu.ua/server/api/core/bitstreams/ead3fa83-2e3d-4cd7-bfbd-1d5ed04c1ce4/content	43 0.31 %

з домашньої бази даних (1.67 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	Розробка програмної моделі оцінки рівня стійкості комп'ютерних систем і мереж [redacted] 5/25/2025 **Odesa Technical Professional College of Odesa National University of Technology (Відокремлений структурний підрозділ "Одеський технічний фаховий коледж Одеського національного технологічного університету")**	**211 (6) 1.50 %**
2	Розробка побутового пристрою вимірювання радіації на базі лічильника J305 [redacted] 5/20/2025 Odesa Technical Professional College of Odesa National University of Technology (Відокремлений структурний підрозділ "Одеський технічний фаховий коледж Одеського національного технологічного університету")	15 (2) 0.11 %
3	Розробка програмної моделі генерування та валідації надійних паролів [redacted] 5/27/2025 Odesa Technical Professional College of Odesa National University of Technology (Відокремлений структурний підрозділ "Одеський технічний фаховий коледж Одеського національного технологічного університету")	9 (1) 0.06 %

з програми обміну базами даних (3.60 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	Алгоритм оптимізації мережевого трафіку [redacted] 3/15/2025 **National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute (National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute)**	**497 (41) 3.53 %**
2	ФККПІ_2021_121_ПолинкевичГА [redacted] 7/11/2024 Ukrainian national aviation university (Ukrainian national aviation university)	10 (1) 0.07 %

з Інтернету (12.99 %)

ПОРЯДКОВИЙ НОМЕР	ДЖЕРЕЛО URL	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	https://card-file.ontu.edu.ua/server/api/core/bitstreams/a141b658-5fa7-4f90-b0bd-7f0ccaed21e5/content	516 (50) 3.66 %
2	**https://card-file.ontu.edu.ua/server/api/core/bitstreams/ead3fa83-2e3d-4cd7-bfbd-1d5ed04c1ce4/content**	**309 (8) 2.19 %**

3	https://card-file.ontu.edu.ua/bitstreams/1dff552d-7200-49b8-ae1d-ba76a1335685/download	209 (17) 1.48 %
4	https://card-file.ontu.edu.ua/bitstreams/53ed22ad-8700-4162-b97a-082a1ad472d6/download	124 (4) 0.88 %
5	https://card-file.ontu.edu.ua/bitstreams/5240e379-7721-49f0-8ee8-27140b0b473a/download	87 (2) 0.62 %
6	https://card-file.ontu.edu.ua/bitstreams/6cf43324-8f08-4031-ba42-f80b18efbbc8/download	79 (3) 0.56 %
7	https://card-file.ontu.edu.ua/server/api/core/bitstreams/44c16132-5f53-48e2-b6c0-61e9a2f0fd75/content	65 (3) 0.46 %
8	https://card-file.ontu.edu.ua/bitstreams/0e72a3b9-bdd7-4711-a3c6-dedc1d4287cc/download	60 (9) 0.43 %
9	https://card-file.ontu.edu.ua/bitstreams/bbed74c8-2ea7-44c5-8d00-0fe3fd9790ee/download	53 (2) 0.38 %
10	http://cpsm.kpi.ua/stud/bak/DP_BAK_KARAULOVA_LU.pdf	48 (3) 0.34 %
11	https://card-file.ontu.edu.ua/bitstreams/11562741-24e6-4201-bc41-a00c8013fca1/download	47 (5) 0.33 %
12	https://card-file.ontu.edu.ua/server/api/core/bitstreams/8da72e29-656f-4ee4-9b22-716dedf53ff5/content	40 (1) 0.28 %
13	https://card-file.ontu.edu.ua/server/api/core/bitstreams/995bdcec-4e4d-4321-8070-4d6badcb8e49/content	40 (4) 0.28 %
14	https://card-file.ontu.edu.ua/bitstreams/035f6436-20b4-4ee6-8e99-bede670e308b/download	32 (1) 0.23 %
15	https://community.cisco.com/t5/switching/site-to-site-vpn-connection-does-not-initiate/td-p/1597359	25 (3) 0.18 %
16	https://card-file.ontu.edu.ua/bitstreams/538ada8a-2c79-4b1e-b7d2-b0c97f68bc1c/download	17 (1) 0.12 %
17	https://card-file.ontu.edu.ua/bitstreams/21173711-5b67-4b87-b17f-6302c25e7a31/download	14 (2) 0.10 %
18	https://dnaop.com/html/32422_4.html	14 (1) 0.10 %
19	https://card-file.ontu.edu.ua/bitstreams/63ee88cb-a3d0-4005-9cf2-0cff89f28c0d/download	12 (2) 0.09 %
20	https://www.cisco.com/c/en/us/td/docs/solutions/Enterprise/Security/SAFE_RNEXT.html	12 (2) 0.09 %
21	https://card-file.ontu.edu.ua/bitstreams/34a6756b-592f-4b77-a805-183aa03a6a26/download	10 (1) 0.07 %
22	https://card-file.ontu.edu.ua/bitstreams/bbaf3f38-16a8-4070-bead-5562769b7c71/download	6 (1) 0.04 %
23	https://card-file.ontu.edu.ua/bitstreams/549ee9fe-7574-4ae5-b500-9fe2711f33e6/download	5 (1) 0.04 %
24	https://card-file.ontu.edu.ua/server/api/core/bitstreams/3302e08a-9549-43ba-8861-728bff7dc7ff/content	5 (1) 0.04 %

Список прийнятих фрагментів (немає прийнятих фрагментів)

ПОРЯДКОВИЙ НОМЕР	ЗМІСТ	КІЛЬКІСТЬ ОДНАКОВИХ СЛІВ (ФРАГМЕНТІВ)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Спеціальність: 123 «Комп'ютерна інженерія»
Освітньо-професійна програма: «Обслуговування комп'ютерних систем і мереж»
Група: 4КС- 58

Дипломний проект здобувача освіти денної форми навчання КС. 58.05.000.ДП

ВИШНЕВСЬКОГО
АНДРІЯ ОЛЕКСАНДРОВИЧА