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

Спеціальність: 123 «Комп'ютерна інженерія»

Освітня програма: «Комп'ютерна графіка і Web-дизайн»

Група: 4КГ-06

Дипломний проект

здобувача освіти денної форми навчання
КГ.06.03.000.ДП

БУХТЄЄВА
МАКСИМА ІГОРОВИЧА

м. Одеса
2023 р.

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

Спеціальність: **123 «Комп'ютерна інженерія»**
Освітня програма: «**Комп'ютерна графіка і Web-дизайн»**
Група: **4КГ-06**

ПОЯСНЮВАЛЬНА ЗАПИСКА

до дипломного проекту (роботи) на тему:

Розробка моделі розпізнавання потоку даних мережі для маршрутизаторів Cisco

Проектний матеріал складається з пояснювальної записки на 68 сторінках та графічного (презентаційного) матеріалу на 17 аркушах (слайдах).

Дипломник _____ (Бухтєєв М.І.)
Керівник _____ (Кривченко А.А.)

Консультанти:
з економічної частини _____ (Копайгородська Т.Г.)
з охорони праці _____ (Чорновол Н.І.)
з дотримання вимог ЄСКД _____ (Петрашова В.І.)
старший консультант _____ (Кривченко А.А.)

До захисту допущений
Голова циклової комісії _____ (Кривченко Ю.В.)
Завідувач відділення _____ (Скорнякова О.В.)

Захист «22» червня 2023 р. Протокол ДКК № 4
Оцінка ДКК 5 (відмінно)

Секретар ДКК _____

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

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

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

Освітня програма **«Комп'ютерна графіка і Web-дизайн»**

ЗАТВЕРДЖУЮ:

Заст. дир. з НВР _____

Беркань І.В.

"_____" _____2023 р.

ЗАВДАННЯ

на дипломний проект (роботу)

Здобувачеві (здобувачці) освіти Бухтєєву Максиму Ігоровичу

(прізвище, ім'я, по батькові)

1. Тема проекту (роботи) Розробка моделі розпізнавання потоку даних мережі для маршрутизаторів Cisco

затверджена наказом по коледжу від "17" жовтня 202 2 р. № 235-А2-ОД

2. Термін здачі закінченого проекту (роботи) 12.06.2023

3. Вихідні данні до проекту (роботи) 1. У якості мережевих пристроїв використовувати маршрутизатори та міжмережеві екрани Cisco; 2. Програмне забезпечення має: зчитувати файл 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. Розробке програмного забезпечення виконувати у ICP Intellij Idea мовою Java.

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

Аналіз моделей розпізнавання мережевих потоків даних

Розробка алгоритмів розпізнавання потоку даних мережі

Розробка програмного забезпечення для розпізнавання потоку даних мережі

Визначення програмних засобів розробки

Тестування моделі розпізнавання потоку даних

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

Методи класифікації трафіку комп'ютерної мережі; Блок-схема алгоритму оптимізації мережевого трафіку; Блок-схема алгоритму читання таблиці портів маршрутизатору Cisco; Блок-схема алгоритму моніторингу навантаження каналу зв'язку; Блок-схема алгоритму скидання файлу конфігурації пристрою; Структурна схема моделі розпізнавання потоку даних; Функціональна схема діаграми класів у програмі; Блок-схема алгоритму основного коду програми; Структура мережі для тестування розробленої моделі; 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
1.Технологічний	Кривченко А.А.		
розділ			
2.Екон. частина	Копайгородська Т.Г.		
3.Охорона праці	Чорновол Н.І.		
Нормоконтроль	Петрашова В.І.		

7. Дата видачі завдання 01.05.2023

Керівник Кривченко А.А. _____
(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.	Вступ. Постановка задачі проектування	23.05.2023	
2.	Аналіз технічного завдання та пошук літератури	25.05.2023	
3.	Аналіз моделей розпізнавання мережевих потоків даних	26.05.2023	
4.	Аналіз засобів моніторингу мережевого трафіку	29.05.2023	
5.	Розробка структури алгоритму оптимізації трафіку	30.05.2023	
6.	Розробка структури алгоритму зчитування таблиці портів	31.05.2023	
7.	Розробка алгоритму моніторингу навантаження на канал	1.06.2023	
8.	Розробка алгоритму скидання файлу конфігурації	2.06.2023	
9.	Визначення програмних засобів розробки	5.06.2023	
10.	Розробка об'єктно-орієнтованої моделі програми	6.06.2023	
11.	Реалізація інтерфейсу програми розпізнавання потоку	7.06.2023	
12.	Тестування моделі розпізнавання потоку даних	8.06.2023	
13.	Аналіз результатів моделювання та ефективності	9.06.2023	
	розробленого алгоритму та моделі		
14.	Економічні розрахунки і розробка питань охорони праці	10.06.2023	
15.	Виконання графічної частини проекту	11.06.2023	

Дипломник _____
(підпис)

Керівник _____
(підпис)

Формат	Зона	Поз.	Позначення	Назва	Кіл.	Примітка
				Документація		
			КГ 06. 03 000. 00 ДП	Дипломний проект		
А4			КГ 06. 03 000. 00 ДП ПЗ	Пояснювальна записка	1	

Зм.	Арк.	№ докум.	Підпис	Дата	КГ 06. 03 000. 00 ДП		
Розробив		Бухтєєв М.І.			Розробка моделі розпізнавання потоку даних мережі для маршрутизаторів Cisco	Літ.	Аркуш 4 / Аркушів 68
Перевірив		Кривченко А.А.				Н Д П	
Н. Контр.		Петрашова В.І.				ВСП "ОТФК ОНТУ" гр.4КГ-06	
Затверд.		Кривченко Ю.В.					

ЗМІСТ

Вступ.....6
1 Технологічний розділ.....7
1.1 Аналіз моделей розпізнавання мережевих потоків даних.....7
1.1.1 Загальні принципи класифікації мережевого трафіку.....7
1.1.2 Принципи класифікації мережевого трафіку від Cisco.....11
1.1.3 Позначення пакетів для застосовування політик служби.....13
1.1.4 Аналіз засобів моніторингу мережевого трафіку.....16
1.2 Розробка алгоритмів розпізнавання потоку даних мережі 19
1.2.1 Розробка структури алгоритму оптимізації трафіку.....20
1.2.2 Розробка структури алгоритму зчитування таблиці портів.....21
1.2.3 Розробка алгоритму моніторингу навантаження на канал зв'язку.....21
1.2.4 Розробка алгоритму скидання файлу конфігурації маршрутизатору 23
1.2.5 Виконання налаштувань за допомогою файлу конфігурації.....24
1.3 Розробка програмного забезпечення для розпізнавання потоку даних мережі.....25
1.3.1 Визначення програмних засобів розробки.....25
1.3.2 Розробка об'єктно-орієнтованої моделі програми.....27
1.3.3 Реалізація інтерфейсу програми розпізнавання потоку даних.....36
1.3.4 Тестування моделі розпізнавання потоку даних.....38
1.3.5 Аналіз результатів моделювання та ефективності алгоритму.....43
2 Економічна частина.....44
3 Охорона праці.....49
Висновки.....54
Перелік використаних джерел.....55
Додаток А. Лістинг класу logic для опису логіки спостереження мережевої активності (мова Java).....56
Додаток Б. Слайди мультимедійної презентації 57

ВСТУП

Через високу складність структур сучасних мереж задача маршрутизації вирішується не в повному обсязі та має високу часову складність. Сьогодні одним з найбільш поширених рішень для забезпечення необхідного сервісу для заданого трафіку в певних технологічних рамках є впровадження технології QoS. Проте ця технологія не є гнучкою. Актуальним є питання підвищення продуктивності використання мережі з зовнішнім каналом, який має низьку пропускну спроможність. На віддалених стратегічних об'єктах, де серед доступних середовищ передачі даних є лише мобільний чи супутниковий зв'язок, відсутність можливості передачі важливих даних через системні оновлення чи перегляд відео одним з працівників може зупинити роботу об'єкта. Постає питання: як можна відрізнити трафік та здійснити автоматичне регулювання пропускної можливості для певних його видів. Рішенням може стати використання протоколу SNMP для збору даних про використаний трафік та протоколу NBAR для його класифікації [1].

Метою даної роботи є застосування SNMP для збору даних про використаний трафік та NBAR для класифікації всього трафіку по типам. Для відправки змін в конфігурації маршрутизаторів Cisco для обмеження певних типів трафіку буде використовуватися протокол SSH. У даній роботі буде виконано розробку моделі розпізнавання потоку даних мережі для маршрутизаторів Cisco на основі програмного продукту, що буде здатен в динамічному режимі розпізнавати трафік. Запропонований продукт буде ділити трафік на важливий і неважливий згідно з заданою конфігурацією. Для забезпечення стабільної роботи критично важливих ресурсів має бути програмно реалізоване обмеження пропускної здатності для неважливого трафіку. Для виконання поставленої задачі необхідно також провести порівняльний аналіз запропонованого алгоритму із ручним налаштуванням мережевого пристрою. Використання розробленого програмного забезпечення має дозволити зменшити кількість ситуацій, коли мережевий адміністратор допускає помилки в налаштуванні мережевих пристроїв. За відсутності високошвидкісного каналу зв'язку маж бути забезпечено безперебійну роботу критично важливих сервісів, без яких робота об'єкта може бути зупинена.

1 ТЕХНОЛОГІЧНИЙ РОЗДІЛ

1.1 Аналіз моделей розпізнавання мережевих потоків даних

Трафік у комп'ютерній мережі має неоднорідну структуру, містить в собі потоки різних додатків. Ці програми потребують дотримання індивідуальних вимог до параметрів мережі. В іншому випадку якість і зручність використання цих додатків не буде надана належним чином.

1.1.1 Загальні принципи класифікації мережевого трафіку

Хоча в локальній обчислювальній мережі, з її величезною пропускною спроможністю, може бути нескладно виконати необхідні вимоги, зазвичай складно задовольнити їх в глобальних мережах, які мають обмеження пропускної здатності. Таким чином, керування трафіком в глобальних мережах необхідне, щоб правильно розставляти пріоритети для різних додатків в обмеженій смузі пропускання і забезпечувати дотримання їх вимог. Крім того, розуміння додатків і протоколів в мережевому трафіку важливо для реалізації відповідних політик безпеки. Існують різні методи і техніки класифікації трафіку мережі (рис.1.1). Класифікація трафіку допомагає ідентифікувати додатки та протоколи, які існують в мережі [2].

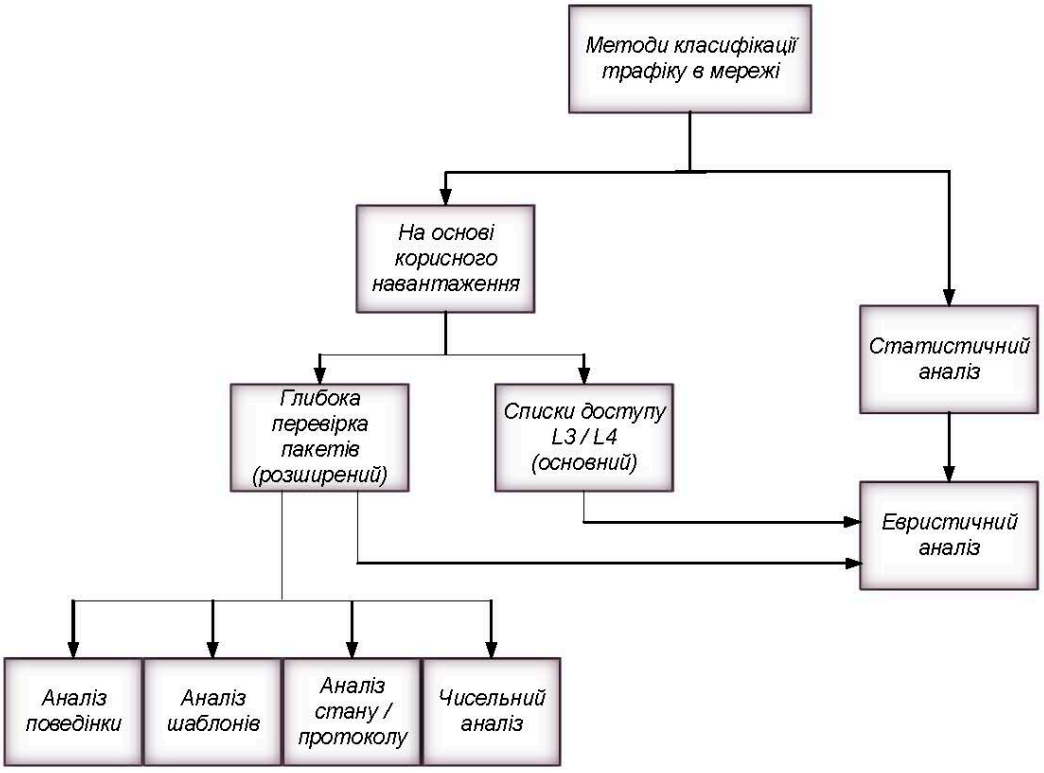


Рисунок 1.1. Методи класифікації трафіку комп'ютерної мережі

Різні дії, такі як: моніторинг, виявлення, управління та оптимізація, можуть виконуватися на класифікованому трафіку з кінцевою метою підвищення продуктивності мережі.

Для класифікації мережевого трафіку існує два підходи:

- класифікація пакета на основі корисного навантаження. У цьому методі пакети класифікуються на основі полів корисного навантаження, таких як порти рівня 4 (джерело або пункт призначення);
- класифікація пакета на основі статистичного методу, який використовує статистичний аналіз поведінки трафіку, такий як міжпакетна затримка, час сеансу і тощо.

Найбільш поширеним є метод на основі корисного навантаження. Його можна розділити на загальний (базовий) аналіз корисного навантаження та розширений аналіз корисного навантаження. Загальний підхід до класифікації трафіку заснований на інформації в заголовку IP. Проглядається наступна інформація IP-заголовку:

- адреса рівня 3 (IP-адреса);
- адреса 2-го рівня (MAC);
- протоколи.

Перевагою цього методу є те, що він дуже простий, але він не забезпечує класифікацію для більшості додатків. Метод класифікації, заснований на розміщенні трафіку (вхідний інтерфейс) також існує, але не використовується широко.

Всі загальні методи класифікації, засновані на IP-адресі одержувача, IP-адресі джерела або IP-протоколі, обмежені в можливостях, оскільки перевірка відбувається тільки за рахунок заголовку IP. Аналогічно, класифікація на основі тільки портів рівня 4 також обмежена. Проблема цього підходу полягає в тому, що не всі сучасні програми використовують стандартні порти. Передовими методами класифікації залишаються ті, що засновані на глибокій перевірці пакетів (DPI). Вони набагато надійніші, ніж загальна методика класифікації. Існують різні методи DPI, такі як аналіз шаблонів або аналіз поведінки.

Існуючі методи класифікації трафіку на основі корисного навантаження також можуть бути розділені на основі методу обробки, який використовується для класифікації трафіку. Незалежно від методу, всі вони використовують одну або кілька методів перевірки корисного навантаження, такі як Deep Packet Inspection для перевірки та класифікації трафіку. Найбільш простий і не пов'язаний з пакетами стан (PBNS) включає перевірку корисного навантаження для певних параметрів, таких як номери портів. Це менше навантажує процесор. Цей метод зазвичай використовує основну методику класифікації на основі корисного навантаження. Однак він не завжди є точним, оскільки класифікація здійснюється на основі пакету без урахування сеансу програми, а також обмежується тим, наскільки глибоко перевіряється потік всередині пакету [2].

У методі PBFS, заснованому на потоках, потік визначається як послідовність пакетів від програми-відправника до програми-отримувача. У цьому методі для кожного потоку підтримується таблиця для відстеження кожного сеансу на основі 5 кортежів (адреса джерела, адреса призначення, вихідний порт, порт призначення та транспортний протокол). Оскільки потік має безліч пакетів, як тільки пакет позначений як приналежність до додатку, всі наступні пакети в потоці повинні бути позначені як такі. Наприклад, у типовому виклику VoIP, H.323 використовується для налаштування виклику, а потім RTP / RTCP використовується для перенесення фактичного голосового трафіку. Як тільки потік H.323 ідентифікований і позначений, наступні потоки RTP / RTCP позначаються тими ж параметрами до однієї IP-адреси джерела / IP-адреси призначення.

Метод MBFS, заснований на повідомленні, аналогічний методу PBFS за винятком того, що він діє на повідомлення замість пакетів. Повідомлення є залежним від протоколу і є інформаційним елементом, який може охоплювати кілька пакетів, або один пакет може містити кілька повідомлень. Оскільки він працює на повідомленнях, необхідно мати певний тип нормалізатора TCP, щоб піклуватися про фрагменти IP і сегменти TCP. Проте вимагається збільшення вимог до пам'яті, оскільки необхідно враховувати повідомлення цілком.

У методі MBPS, заснованому на протоколі, не тільки відстежується програма,

але й здійснюється передача програми. Іншими словами, для реалізації цього методу необхідне повне знання автомата протоколу. Цей метод вимагає більше ресурсів процесору і має серйозніші вимоги до пам'яті.

Методи класифікації PBFS, MBFS і MBPS використовують передові методи класифікації, що базуються на глибокій перевірці пакетів [3].

Не дивлячись на те, що більшість загальних додатків можна визначити або принаймні припустити на основі інформації L3 і L4, необхідні додаткові підкласи в програмах (наприклад, URL-адреси) або конкретні види повідомлень у межах програми. Для належної класифікації необхідно провести глибоку перевірку пакетів (DPI) і перевірити, що це за додаток. Більшість механізмів DPI використовують аналіз підписів для розуміння та перевірки різних додатків. Підписи є унікальними шаблонами, які пов'язані з кожним додатком. Іншими словами, кожний додаток вивчається з урахуванням його унікальних характеристик і створюється довідкова база даних. Механізм класифікації потім порівнює трафік з цим посиланням для визначення точних додатків. Це означає, що посилання необхідно періодично оновлювати, щоб підтримувати інформацію про додатки, а також нові розробки в існуючих протоколах. Існують різні методи аналізу підпису:

- аналіз шаблонів;
- чисельний аналіз;
- поведінковий аналіз;
- статистичний аналіз;
- аналіз стану / протоколу.

В корисне навантаження пакетів деякі програми вбудовують певні шаблони (байти / символи / рядки), які можуть використовуватися механізмом класифікації для ідентифікації таких протоколів. Залежно від програми ці структури не обов'язково завжди мають бути розташовані на конкретному детермінованому зміщенні.

Шаблони можуть бути присутніми в будь-якому положенні пакета. Тим не менш, механізм класифікації може ідентифікувати ці пакети. Однак не всі протоколи вбудовують в пакети особливий шаблон, рядок або символи, і тому цей

підхід в таких випадках працювати не буде.

Вивчення числових характеристик пакетів, таких як розмір корисного навантаження, кількість пакетів відповідей і зміщення передбачає чисельний аналіз. Запит від клієнта є 18-байтовим повідомленням, і відповідь, яку він отримує, зазвичай становить 11 байт. Рішення про класифікацію може зайняти більше часу, оскільки аналіз може поширюватися на множинні пакети.

Аналіз поведінки трафіку іноді дає змогу краще зрозуміти програми, що можуть бути запущені. Такий метод може бути використаний для класифікації таких додатків. Аналогічним чином, здійснюючи статистичний (евристичний) аналіз контрольованих пакетів, основний протокол може бути класифікований. Поведінковий і евристичний аналіз зазвичай йдуть поряд, і для виявлення вірусів багато антивірусних програм використовують саме ці методи. У деяких додатках протокол відповідає певній послідовності кроків або дій. Наприклад, типовий запит FTP GET від клієнта супроводжується дійсним відгуком від сервера. Оскільки все більше додатків починають шифрувати трафік, для будь-якого механізму класифікації стає проблематично класифікувати програми. При шифруванні вся інформація верхнього рівня стає невидимою для механізмів DPI. Поведінкові та статистичні методи аналізу можуть допомогти визначити деякі програми. Нові механізми класифікації, які використовують ці методи аналізу (разом з інтелектуальними алгоритмами, такими як алгоритми кластеризації), можуть допомогти ідентифікувати зашифрований трафік [3].

В типовому розгортанні ці методи використовуються разом, адже жоден з цих методів самостійно не може забезпечити задовільну класифікацію всіх застосувань.

1.1.2 Принципи класифікації мережевого трафіку від Cisco

Технології класифікаційні Cisco включають списки доступу QoS і технологію DPI. Списки доступу L3 / L4 на основі програмного забезпечення QoS надають можливість налаштувати списки доступу на основі рівня 3 або 4, які можна використовувати з QoS для класифікації різних типів трафіку.

Конкретні класи QoS можуть бути налаштовані на використання різних списків доступу для відповідності трафіку і на основі відповідності можуть бути

позначені пакети. Узгодження може бути засноване на адресах рівня 3 (IP джерела / призначення), протоколі рівня 4 або портах або їх комбінації. На додаток до програмного забезпечення на основі ACL, платформи Cisco, такі як 6500 і GSR, дають можливість виконувати пошук ACL в апаратному забезпеченні. Наприклад, в платформі 6500 ці ACL можуть бути запрограмовані в пам'яті, що адресуються Ternary Content Addressable Memory (TCAM), і пошуки, виконані проти цих записів.

TCAM мають обмежену пам'ять і без ретельного планування ресурси можуть бути вичерпані. Пошук TCAM набагато швидший, ніж традиційні пошуки програмного забезпечення, оскільки він виконується апаратно.

DPI може бути спільним резидентом в програмному забезпеченні або може бути виділеним апаратними засобами. Хоча спеціальне апаратне забезпечення забезпечує швидкість і універсальність, його вартість розгортання обмежує використання середовищами з великим обсягом трафіку, наприклад центрами обробки даних або великими відділеннями підприємств. Інструмент керування службами Cisco (SCE) є гарним прикладом спеціального обладнання DPI. Двигуни на основі програмного забезпечення на основі програмного забезпечення є економічно ефективними, але вони споживають ресурси процесору і, отже, можуть бути розгорнуті лише в низьких або середніх обсягах трафіку. SCE представляє собою пристрій DPI, який може виконувати функцію DPI і визначати шаблони трафіку на лінії. SCE включає в себе безліч технологій DPI, таких як аналіз протоколу/стану, аналіз структури, поведінковий та евристичний аналіз. SCE також може здійснювати класифікацію на рівні абонентів. Cisco SCE може бути розгорнуто в діапазоні або поза діапазоном. Зазвичай він розгортається в центрі даних. Якщо він розгорнутий в смузі, весь трафік в мережі проходить через SCE. Якщо він розгорнутий поза смугою, то копія всього трафіку передається на SCE за допомогою перемикача постійного струму [4].

Мережеве розпізнавання додатків (NBAR) є функцією класифікації в IOS. NBAR може знаходитись глибоко всередині пакету і здійснювати аналіз стану інформації в пакеті. Він може розпізнавати ряд додатків, включаючи ті, що

використовують ефемерні порти. Навіть з даним протоколом, NBAR може виглядати настільки глибоко всередині пакетів, що він може класифікувати пакети, які мають один і той же протокол, але з різними параметрами, специфічними для протоколу. Наприклад, NBAR може класифікуватися на основі URL-адрес для HTTP-пакетів і заснований на трафіку ICA для CITRIX ICA. Як правило, QoS і NBAR використовуються спільно. NBAR використовується для розпізнавання конкретних додатків, а QoS – для їх позначення і забезпечення відповідної обробки на основі маркування.

1.1.3 Позначення пакетів для застосовування політик служби

Після того, як потік і пакети були ідентифіковані, вони повинні бути позначені так, щоб на них могли застосовуватися відповідні політики служби. Маркування або прапори можна встановити кількома способами: для IP, типу послуги (ToS) або точки диференційованого обслуговування (DSCP); для пакетів Ethernet, пріоритет VLAN тощо. Найбільш широко використовуваним методом є маркування L3.

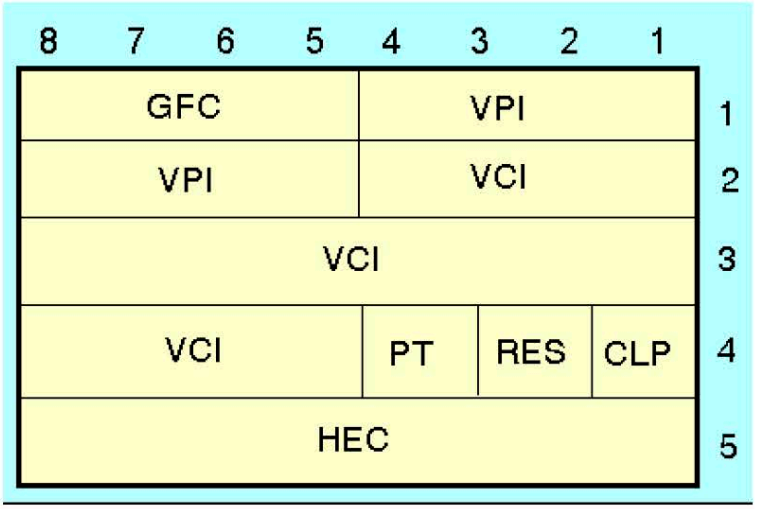


Рисунок 1.2. Вміст ATM-комірки і розташування біта CLP

Популярні технології другого рівня (L2), такі як режим асинхронного перенесення (ATM), Frame Relay (FR) і Ethernet, надають варіанти маркування пакетів, щоб допомогти забезпечити диференційоване зчитування.Мережі ATM використовують просте маркування клітинки біта пріоритету втрати клітини (CLP) на заголовку комірки, щоб вказати, чи може комірка бути скинутою під час перевантаження. Типова ATM Cell складається з 5-байтового заголовка і 48-

байтового корисного навантаження (рис. 1.2). Якщо біт CLP встановлений в "1", комірка може бути скинута в часи перевантаження.

Для позначення біта CLP використовується комутатор ATM. Ця функціональність була додана до CISCO IOS як частина розширеного набору функцій QoS. Користувач може вибрати маркування некритичного трафіку, що проходить через комутатори ATM, з бітом CLP. Під час перевантажень це забезпечить доступність пропускної здатності для критичного трафіку [5].

Заголовок Frame Relay також має біт, що називається Discard Eligible (DE), щоб вказати, чи може кадр бути скинутим під час перевантаження (рис.1.3). Біт DE може бути встановлений для некритичного трафіку, щоб допомогти полегшити перевантаження.

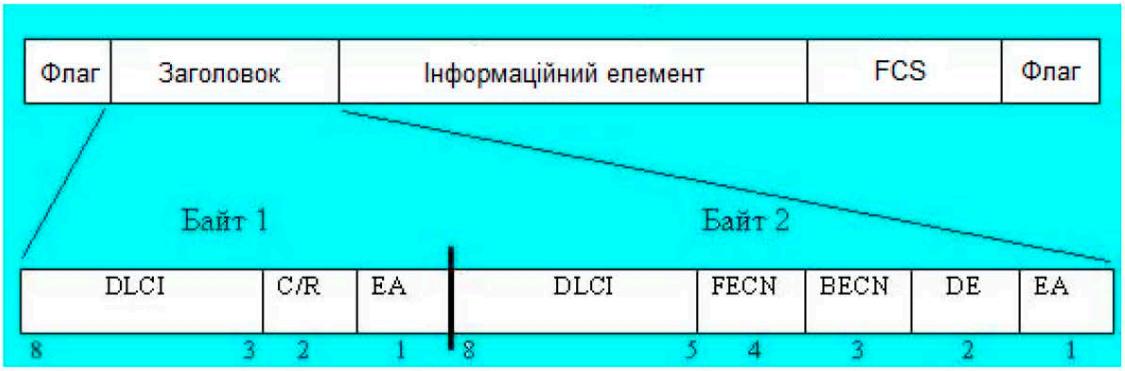


Рисунок 1.3. Вміст кадру Frame Relay

За стандартом IEEE 802.1p забезпечується прискорення класу трафіку та динамічна фільтрація багатоадресної передачі. Вона дозволяє комутаторам 2-го рівня визначати пріоритети трафіку.

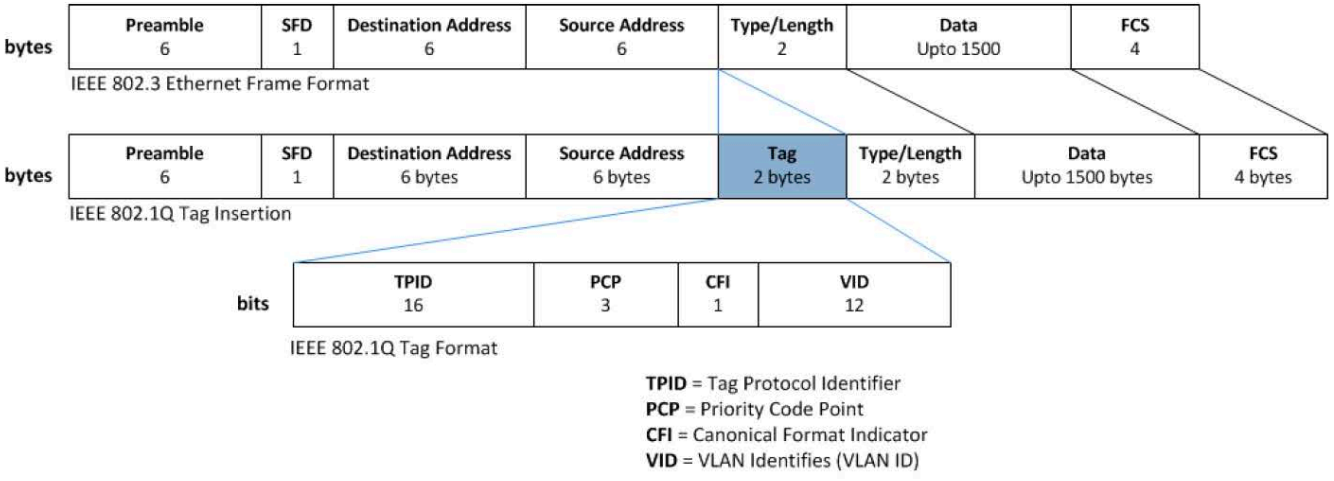


Рисунок 1.4. Вміст кадру Ethernet 802.1q

Специфікація 802.1p визначає 3 біти в заголовку для класифікації, що допомагає класифікувати трафік у вісім різних класів трафіку. Слід зазначити, що 802.1p є розширенням стандарту 802.1q і вони працюють разом.

Кадр Ethernet 802.1q і байт TAG, де розташовані біти пріоритету, показаний на рис. 1.4. Організація IEEE висунула рекомендації щодо різних типів трафіку (табл. 1.1), відповідних класів трафіку та пріоритетів, які будуть використовуватися зі стандартом 802.1p.

Таблиця 1.1. 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

Заголовок IP (рис.1.5) має поля, які можна використовувати для класифікації трафіку в групи обробки. Найбільш широко використовуваними методами маркування L3 є тип послуги (ToS) і DSCP.

Рисунок 1.5. Структура IP-заголовку за стандартом 802.1p

У таблиці 1.2 деталізуються біти пріоритету та їхні можливі значення:

– затримка – коли встановлено значення 1, пакет запитує низьку затримку.

– пропускна здатність – при встановленні в 1 пакет вимагає високої

пропускної здатності.

– надійність – при встановленні на «1», пакет вимагає високої надійності.

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

У RFC 2474 і RFC 2475 була визначена кодова точка диференційованої служби (DSCP). DiffServ (DS) має більше рівнів пріоритету, ніж у ToS, оскільки DS має більше бітів пріоритету. Поля DS використовуються для визначення поведінки на хоп (PHB) пакета. Біти ECN не були в оригінальних RFC DSCP. Пізніше вони були додані у RFC 3168 щоб дозволити повідомлення про перевантаження на шляху. RFC 2597 для DiffServ визначає PHB з гарантованою переадресацією (AF), який може бути використаний постачальником послуг для надання різних гарантій переадресації на основі різних класів автофокусування. Існує чотири різні класи AF з трьома різними ймовірностями падіння. RFC 2598 для DiffServ визначає прискорене пересилання (EF) PHB. "EF PHB може використовуватися для побудови низьких втрат, низьких затримок, низького джиттера, гарантованої пропускної здатності та наскрізного обслуговування за допомогою доменів DS (Diffserv). Для EF PHB рекомендується використовувати код 101110. Всі PHB вимагають підтримки постачальників для реалізації, і не всі постачальники підтримують їх повністю [6].

1.1.4 Аналіз засобів моніторингу мережевого трафіку

Для забезпечення максимальної продуктивності мережі життєво важливими є ефективний моніторинг мережі та керування трафіком. Хоча технології SFlow, NetFlow і SNMP пропонують різні засоби моніторингу мережевого трафіку, необхідно визначити відмінності між цими технологіями.

Найкраще рішення завжди залежить від стану мережі та наявних ресурсів. Технологія SFlow розроблена корпорацією InMon. Вона призначена для сумісності на багатьох різних платформах комутаторів і мережевих маршрутизаторів, що дозволяє SFlow зростати в популярності. Технологія SFlow (рис. 1.6) використовує спеціальний чіп, який вбудований в апаратне забезпечення, що знімає навантаження з процесора і пам'яті [5].

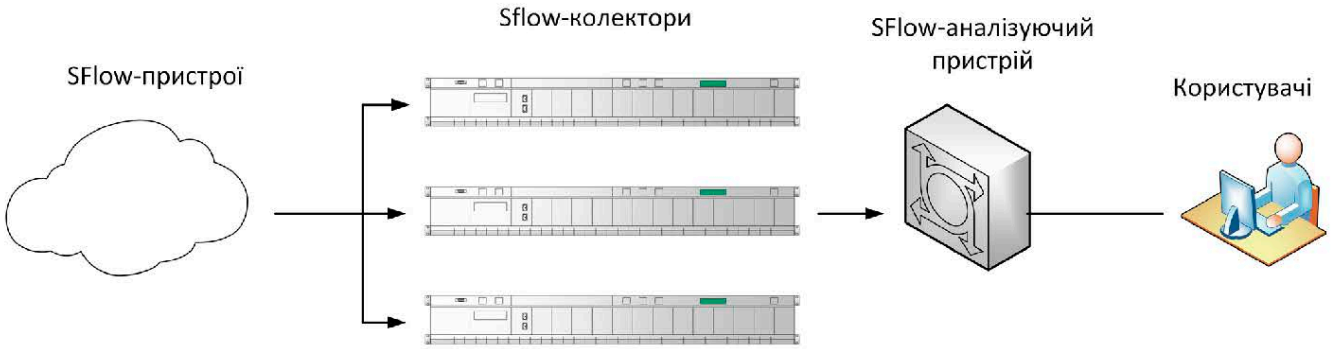


Рисунок 1.6. Організація роботи технології SFlow у мережі

Технологія NetFlow (рис. 1.7) розроблена компанією Cisco і представлена в комутаторах і маршрутизаторах Cisco, що дає змогу мережевим пристроям експортувати дані IP-потоку в колектор NetFlow / аналізатор NetFlow, який збирається, обробляється і далі розкривається. Технологія NetFlow має дуже малий вплив на процесор.

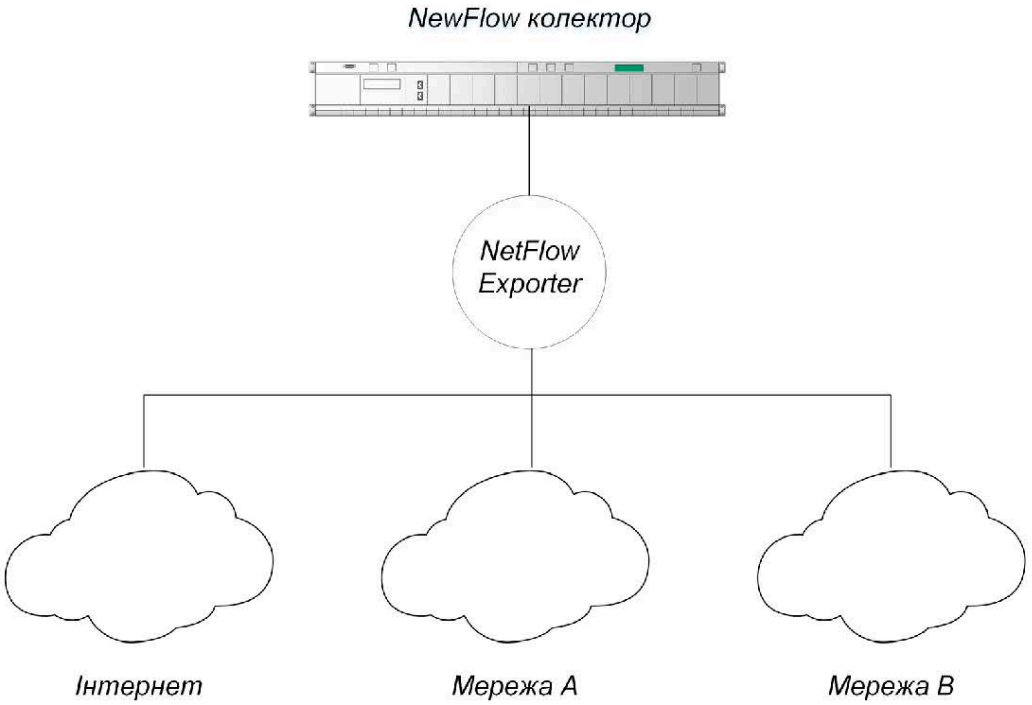


Рисунок 1.7. Організація роботи технології NetFlow у мережі

Важливою відмінністю між технологіями SFlow і NetFlow є те, що SFlow є незалежним мережевим рівнем і має можливість отримати доступ до трафіку з рівнів 2-7 моделі OSI, тоді як NetFlow обмежується тільки трафіком IP. При виборі технології SFlow або NetFlow треба врахувати наступні аспекти:

- якщо мережа підтримує багатопротокольне середовище, можна застосувати технологію SFlow і комутатори;
- якщо мережа підтримує лише трафік, заснований на протоколі IP, можна застосувати технологію NetFlow;
- якщо потрібна 100% точність мережевого трафіку та підзвітності, краще застосувати технологію NetFlow.

Відомо, що як SFlow, так і NetFlow можна використовувати для отримання видимості мережі та вимірювання використання пропускної здатності. Вони також є найпотужнішим варіантом моніторингу для мереж великого трафіку та просунутих користувачів. З іншого боку, SNMP (Simple Network Management Protocol, простий протокол керування мережею) є основним засобом збору даних про пропускну спроможність та використання мережі. Моніторинг використання пропускної здатності маршрутизаторів і комутаторів по портам є найпоширенішим використанням SNMP, а також моніторингу показань пристроїв, таких як пам'ять, завантаження процесора тощо.

Протокол SNMP виявився дуже популярним протоколом керування мережею, який використовується в основному для моніторингу мережі. Що стосується керування продуктивністю на маршрутизаторах / комутаторах, особливо у випадку з багатопротокольною передачею, то незалежний шар SFlow повинен бути вибором для збору, моніторингу та аналізу трафіку даних.

Протокол NetFlow виступає як більш компактний протокол, ніж SNMP, який краще масштабується для збору продуктивності та керування мережевим трафіком. Відмінності між протоколами SNMP та NetFlow є такими:

- протокол SNMP може використовуватися для реального часу (тобто кожну секунду), і хоча NetFlow надає початковий і кінцевий час для кожного потоку, це робиться не так, як у SNMP-протоколі;

- протокол NetFlow визначає джерела споживання пропускної здатності мережі. Він набагато більш детальний, ніж SNMP і тому експорт NetFlow споживає набагато більше дискового простору для статистичної інформації;
- протокол SNMP може використовуватися для збору статистики по споживанню ресурсів процесора та пам'яті, що недоступне за допомогою протоколу NetFlow.

Протокол SNMP призначений для стандартного моніторингу мережі, в той час як протоколи SFlow / NetFlow призначені для аналізу, збору, і моніторингу мережевого трафіку. Протокол SFlow призначений для комутаторів багатопротокольної мережі, а протокол NetFlow для трафіку на основі IP, що вимагає підвищення точності та масштабованості. Комутатори, що підтримують одночасно протоколи SFlow, NetFlow і SNMP теж існують [5].

У якості попередніх висновків можна зазначити, що класифікація трафіку допомагає ідентифікувати додатки та протоколи, які існують в мережі. Різні дії, такі як: моніторинг, виявлення, керування та оптимізація можуть виконуватися на класифікованому трафіку з кінцевою метою підвищення продуктивності мережі. Описані вище механізми QoS, такі як керування перевантаженнями, запобігання перевантаженням, поліпшення трафіку, формування та ефективність зв'язку, можна використовувати для керування пропускною спроможністю глобальної мережі. Керування трафіком в глобальних мережах в обмеженій смузі пропускання необхідне для правильної розстановки пріоритетів для різних видів додатків.

1.2 Розробка алгоритмів розпізнавання потоку даних мережі

Для обмеження швидкості трафіку певних типів даних використовується розпізнавання цих типів трафіку певним чином та внесення політик обмеження на мережевий маршрутизатор Cisco. При цьому політики обмеження будуть працювати завжди, в не залежності від потреби. Також є можливість резервування певної полоси пропускання для певного виду трафіку проте це також може негативно вплинути на передачу інших видів трафіку, або взагалі при великій кількості типів трафіку, які повинні гарантовано доставлятись швидкість може

бути обмежена настільки, що коректна робота критично важливих сервісів буде неможлива. Вирішити цю проблему дозволяє зменшення пропускної здатності певних типів трафіку, від яких не залежить робота критично важливих сервісів.

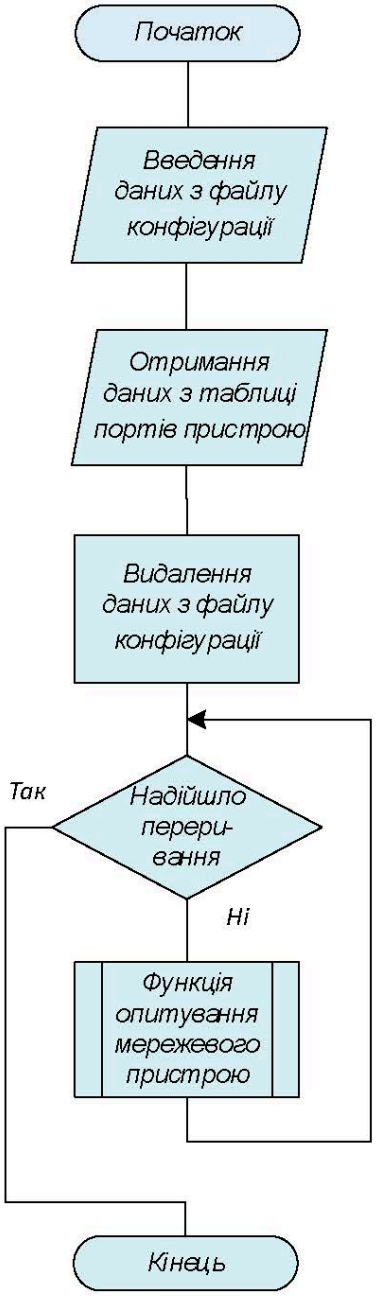


Рисунок 1.8. Блок-схема алгоритму оптимізації мережевого трафіку

1.2.1 Розробка структури алгоритму оптимізації трафіку

Пропонований алгоритм з використанням декількох мережевих механізмів та протоколів дозволить вирішити проблему оптимізації трафіку. Загальна структура запропонованого алгоритму представлена на рис.1.8. При цьому будуть використані наступні механізми:

- SSH – механізм для безпечного з'єднання та передачі команд на мережевий маршрутизатор Cisco;
- NBAR – механізм для розпізнавання та класифікації трафіку;
- SNMP – механізм для зчитування показників з пристроїв.

1.2.2 Розробка структури алгоритму зчитування таблиці портів

Оптимізація трафіку та накладання обмежень на нього передбачає зчитування значень кількості вхідних октетів та класифікації трафіку на порті WAN мережевого маршрутизатору Cisco та застосовування політики обмеження на LAN-порт. Для маркування портів та передачі додаткових даних, які можуть знадобитись в майбутньому пропонується використовувати опцію description при налаштуванні порту. Дані, аналогічні команді «show interfaces», доступні за допомогою SNMP-запиту з записом OID 1.3.6.1.2.1.2.2. За допомогою цього запиту можна отримати повну інформацію про всі порти, включаючи блок description та кількість вхідних, вихідних октетів для кожного з портів. Алгоритм зчитування таблиці портів наведено на рис. 1.9.

Рисунок 1.9. Блок-схема алгоритму читання таблиці портів маршрутизатору Cisco

1.2.3 Розробка алгоритму моніторингу навантаження на канал зв'язку

Необхідно визначити ситуації, коли обмеження швидкості передачі певних протоколів мають бути застосовані. Причиною певних обмежень може бути

завантаження каналу зв'язку. Для цього слід з певною частотою зчитувати з мережевого маршрутизатору Cisco інформацію про кількість вхідних октетів і на основі цих даних робити висновки про навантаження на канал зв'язку. Для визначення завантаженості каналу зв'язку можна скористатися формулою (1.1).

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

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, біт за секунду.

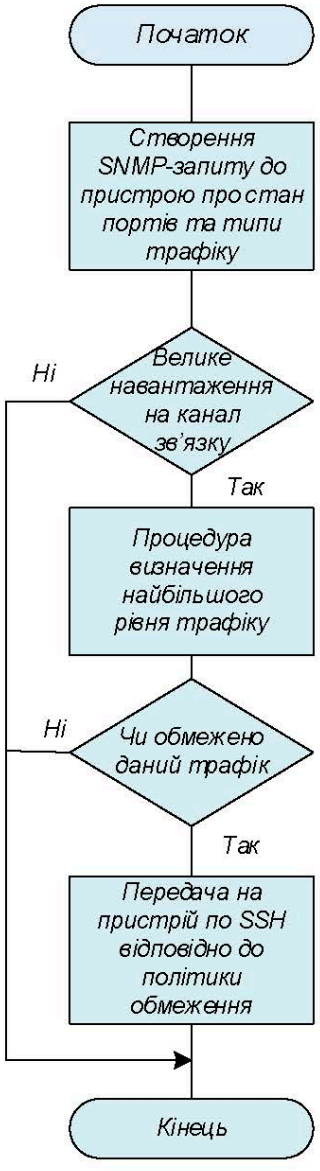


Рисунок 1.10. Блок-схема алгоритму моніторингу навантаження каналу зв'язку

За допомогою SNMP-запиту з OID 1.3.6.1.2.1.2.2 може бути отримана інформація про кількість вхідних октетів. Завжди тримаючи в пам'яті два останніх результати запиту та інтервал між запитами можна визначити поточне навантаження на канал зв'язку. При досяжності певного проценту, що визначається, як відношення поточного навантаження до реальної швидкості каналу зв'язку слід визначити чим зумовлене таке навантаження на канал зв'язку. Для цього слід разом з інформацією про кількість вхідних октетів зчитувати та зберігати таблицю використання типів трафіку. Вона може бути отримана за допомогою SNMP запиту з OID 1.3.6.1.4.1.9.9.244.1.2.1. Далі слід визначити за формулою вказаною вище тип трафіку, який дає найбільше навантаження. Також слід перевірити, чи не є цей тип трафіку таким, від якого залежить робота критично важливих сервісів. Якщо швидкість трафіку можна обмежити, то слід сформувати список команд, які будуть надіслані мережевому маршрутизатору Cisco по SSH для обмеження швидкості передачі обраного типу трафіку, список команд для скасування політики обмежень. Структуру алгоритму роботи моніторингу навантаження на канал зв'язку наведено на рис. 1.10.

Постійної потреби в обмеженні трафіку може не бути, тому слід застосовувати політики обмеження на певний час. Для цього слід фіксувати час, коли політика обмеження була застосована, та час її дії. Час застосування пропонується використовувати в назві. В подальшому слід відслідковувати, чи не закінчився термін дії політики. У випадку, коли термін дії політики закінчився, слід застосовувати список команд для скасування цієї політики обмеження [7].

1.2.4 Розробка алгоритму скидання файлу конфігурації маршрутизатору

Можлива ситуація, коли відбувається аварійне завершення програми і застосовані політики залишаються на мережевому пристрої. Алгоритм очищення конфігураційного пристрою при аварійному завершенні програми наведено на рис.1.11. Для того, щоб програма могла виконувати очищення конфігураційного файлу при запуску програми, слід передбачити генерування списку команд для скасування політик обмеження, а також зчитування списку класів трафіку та політик обмеження.

Рисунок 1.11. Блок-схема алгоритму скидання файлу конфігурації пристрою

1.2.5 Виконання налаштувань за допомогою файлу конфігурації

Мають бути виконані наступні налаштування та визначені такі параметри для коректної роботи створюваної програми:

- логін та пароль для з'єднання за допомогою SSH;
- доменне ім'я або IP-адреса мережевого маршрутизатору Cisco;
- community-рядок для з'єднання за допомогою SNMP;
- період оновлення для моніторингу активності мережевого пристрою;
- час, на який застосовується політика обмеження;
- відношення поточного навантаження до максимальної пропускної здатності каналу зв'язку, після якого застосовується політика обмеження;
- відношення кількості дозволеного трафіку, до поточного навантаження.

При перевищенні цього показника, трафік, що не є дозволеним, буде обмежений у швидкості;

– список протоколів, які не будуть обмежуватись.

Для опису конфігураційних файлів у програмі пропонується використовувати популярний формат YAML. Він більш зручний для читання та має високу швидкість обробки. Запропонований спосіб оптимізації мережевого трафіку з використанням технології NBAR дозволить на основі розпізнавання трафіку відслідковувати навантаження на канал зв'язку та застосовувати політики обмеження в автоматичному режимі. Цей алгоритм суттєво зменшить кількість задач, які має виконувати адміністратор. Також це дозволить обмежити швидкість для певних видів трафіку на певний проміжок часу, а не на постійній основі, що дасть можливість для критично важливих типів трафіку резервувати певну смугу пропускання [8].

1.3 Розробка програмного забезпечення для розпізнавання потоку даних мережі

Створюване програмне забезпечення відповідно до технічного завдання має виконувати розпізнавання потоку даних мережі для маршрутизаторів Cisco. Структурну схему програми наведено на рис.1.12.

1.3.1 Визначення програмних засобів розробки

Для моделювання запропонованого алгоритму було обрано об'єктно-орієнтовану мову програмування Java. Такий вибір обґрунтовано наступними факторами:

– мова Java є крос-платформною, тобто є можливим виконання вже зібраних програм, написаних на Java, незалежно від платформи/операційної системи (зокрема, Linux, Unix, Mac OS, Windows, Android);

– для мови Java передбачено великий вибір інструментів та бібліотек, що можна застосувати для програмування необхідного рішення;

– для мови Java створені зручні програмні рішення інтегрованих середовищ розробки (зокрема, Intelij IDEA, Eclipse);

– мова Java є сучасною та популярною, її застосування є актуальним та корисним для покращення навичок програмування та розробки додатків.

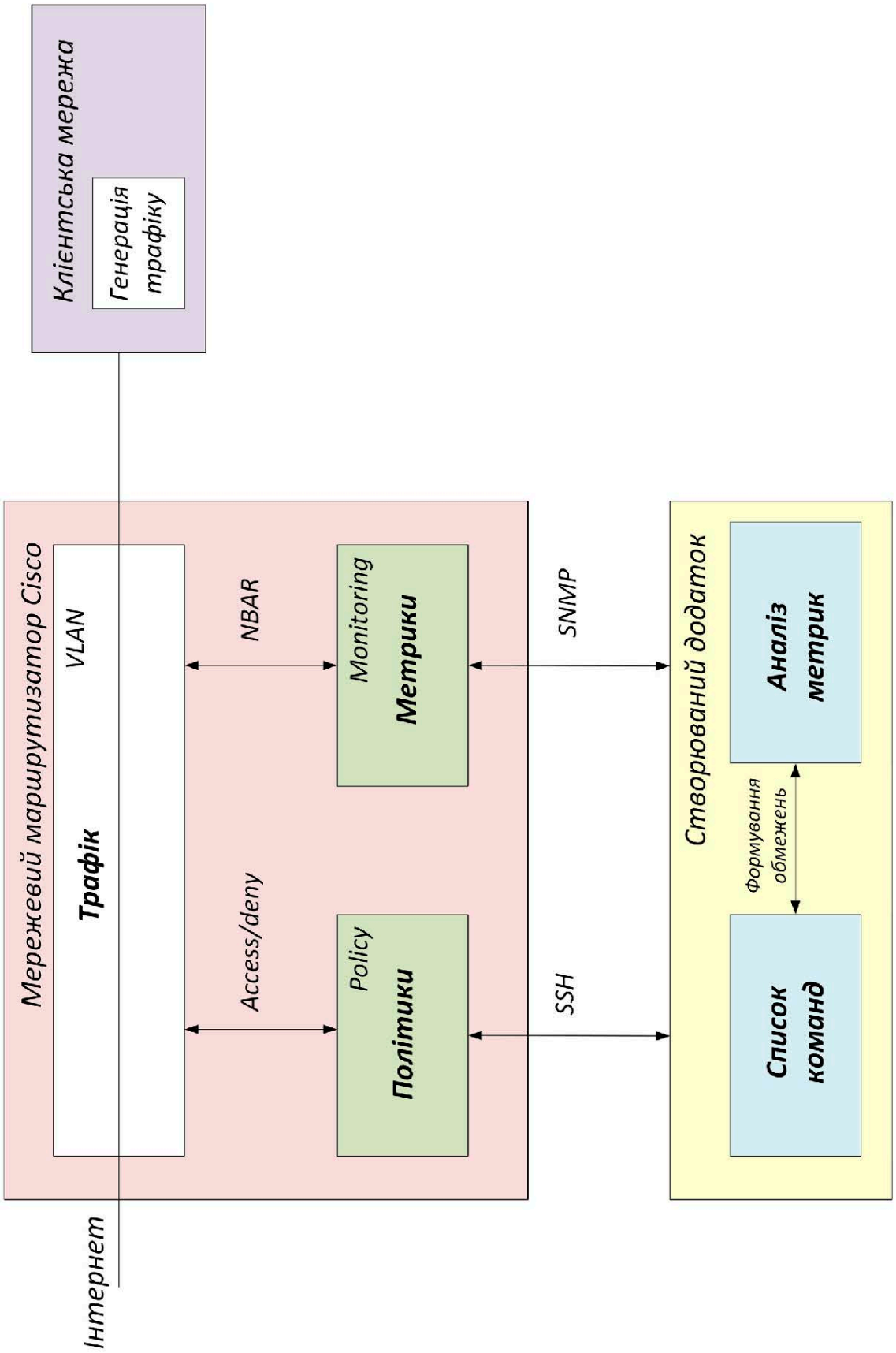


Рисунок 1.12. Схема структурна створюваної моделі розпізнавання потоку даних

У якості інтегрованого середовища розробки при створенні моделі розпізнавання потоку даних мережі для маршрутизаторів Cisco буде використовуватись Intellij IDEA, встановлено Java JRE 8 та Java SDK 8. Для автоматизації збірки проекту використовується фреймворк Apache Maven на основі опису структури в файлах мовою Project Object Model. Для моніторингу стану мережевого приладу та отримання графіків використання каналу мережі використано програмне забезпечення Check_MK. Це моніторингова система побудована на основі Nagios. Вона має велику кількість плагінів для роботи з пристроями різних вендорів та може працювати з протоколами SNMP, агентами для моніторингу. Система включає в себе веб-сервер Apache, систему оповіщення, базу даних для збереження накопичених метрик. Встановлений агент Check_MK збирає таку інформацію:

- завантаження центрального процесору;
- температуру системи;
- завантаження файлової системи;
- навантаження мережевих інтерфейсів;
- завантаження оперативної пам'яті;
- переключення контексту ядра;
- кількість активних потоків.

1.3.2 Розробка об'єктно-орієнтованої моделі програми

Створювана модель розпізнавання потоку даних мережі та програмний продукт буде надавати наступні можливості:

- імпорт файлу конфігурації системи описаний у форматі YAML;
- комунікацію з пристроями по протоколам SSH та SNMP;
- формування та видалення тимчасових конфігурацій для шейпінгу типів трафіку;
- можливість розширення функціоналу програми за рахунок додавання MIB для інших вендорів.

Програма буде складатися з декількох модулів:

- парсер конфігураційного файлу;
- класи, що описують пристрій та зберігають поточні конфігурації шейпінгу на пристрої та історію даних, що були отримані з цього маршрутизатору Cisco;
- логіку програми, що відслідковує зміни та в разі потреби формує політики шейпінгу;
- конектори для комунікації з пристроєм по протоколам SSH та SNMP.

Клас Config відповідає за парсинг конфігураційного файлу в форматі YAML, а класи ClassMap, Host, PolicyMap, Port, Snapshot, Timestamp, Traffic відповідають за опис маршрутизатору Cisco, поточні конфігурації та опис даних про використання мережі та типи трафіку. Клас Logic відповідає за моделювання алгоритму, а класи SSHClient, SNMPClient відповідають за комунікацію з пристроєм за протоколами SSH та SNMP. Функціональну схему діаграми класів, створених у програмі, наведено на рис.1.13.

При цьому клас Config містить список hosts зі списком пристроїв, які будуть відслідковуватись, та конструктор в якому описано парсинг конфігураційного файлу, шлях до якого передається як параметр. Клас ClassMap відповідає за опис класу трафіку, містить тип критерію match, назву класу трафіку name та назву протоколу, який блокується protocol. Також він містить методи, що зчитують значення цих змінних. Опис кожного з пристроїв зберігається в класі Hosts. Він містить такі змінні:

- змінна name – назва маршрутизатору Cisco. Може буди доменним ім'ям, або IP адресою;
- змінна user – ім'я користувача потрібне для з'єднання по SSH;
- змінна password – пароль користувача потрібний для з'єднання по SSH;
- змінна community – «рядок спільноти», що використовується для з'єднання по SNMPv1 або SNMPv2;
- змінна protocols – список протоколів, на які не будуть накладатись обмеження;

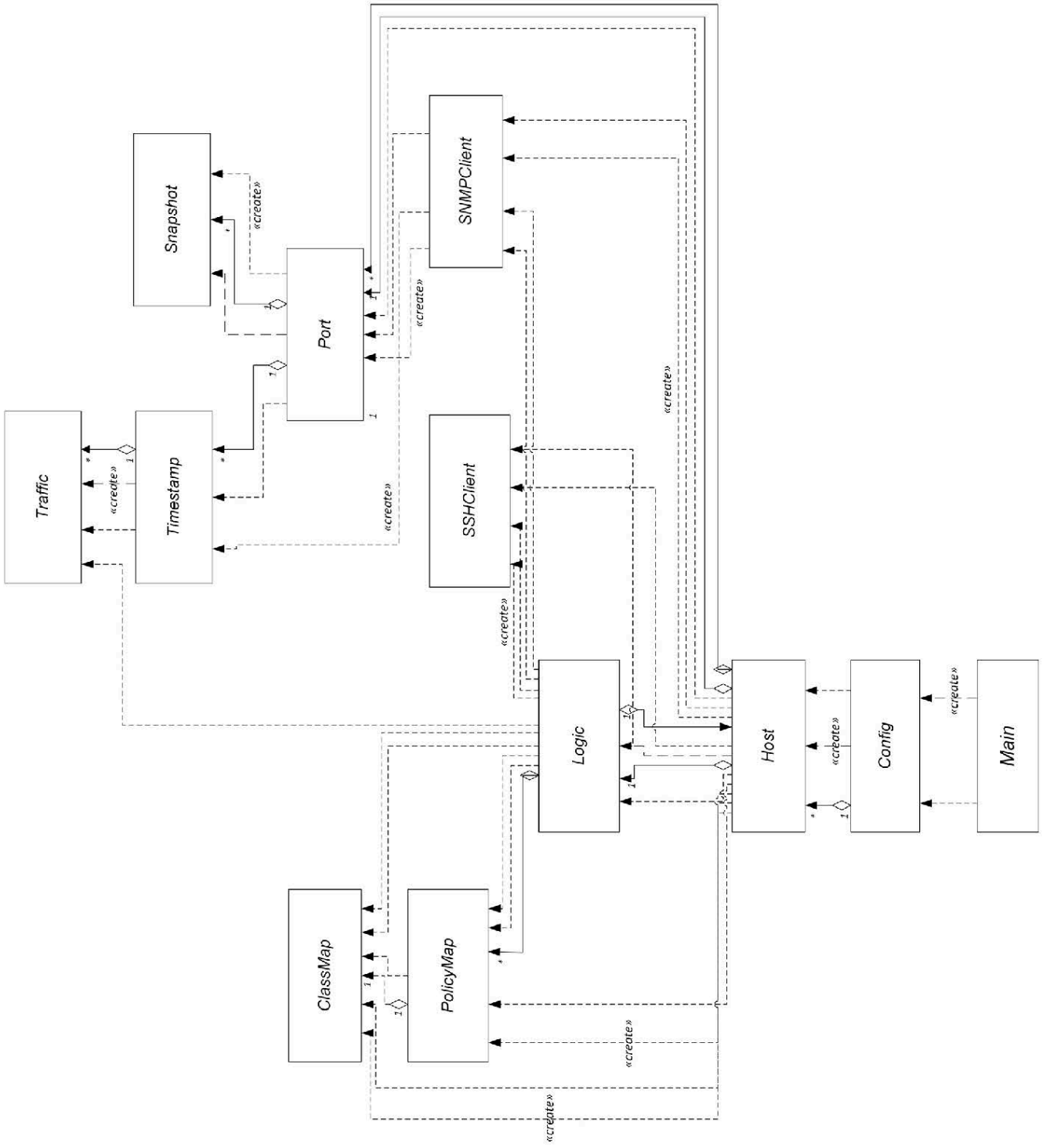


Рисунок 1.13. Функціональна схема діаграми класів у програмі

- змінна ports – список портів маршрутизатору Cisco;
- змінна WAN – порт маршрутизатору Cisco, який використовується для з'єднання з мережею Інтернет;
- змінна LAN – порт маршрутизатору Cisco, який використовується для з'єднання з локальною мережею;

- змінна logic – об’єкт для відслідковування активності маршрутизатору Cisco;
- змінна update – частота оновлень інформації про активність маршрутизатору Cisco;
- змінна TTL – час, на який застосовується політика обмеження;
- змінна bandwidth_usage – пороговий процент завантаженості каналу зв’язку, до якого політики обмеження не застосовуються;
- змінна allowed_percent – пороговий процент співвідношення протоколів, на які не накладаються обмеження до інших, після кого політики обмеження не застосовуються [9].

Нижче зазначені методи, які містить клас Host:

- метод getPorts – за допомогою SNMP отримує список портів маршрутизатору Cisco та ідентифікує WAN та LAN порт.
- метод makeLogic – створює об’єкт класу Logic для відслідковування мережевої активності маршрутизатору Cisco;
- метод cleanMaps – очищує Policy Map та Class Map, які могли залишитись на пристрої через некоректне завершення роботи програми;
- метод isData – перевіряє назви Policy Map чи Class Map на наявність дати;
- метод makeRevercePMAP – створює список команд для очищення Policy Map з маршрутизатору Cisco;
- метод makeReverceCMAP – створює список команд для очищення Class Map з маршрутизатору Cisco.

Нижче зазначені службові методи, які містить клас Host:

- метод getName – повертає ім’я маршрутизатору Cisco;
- метод getUser – повертає ім’я користувача потрібне для з’єднання по SSH;
- метод getPassword – повертає пароль користувача потрібний для з’єднання по SSH;

– метод getProtocols – повертає список протоколів на які не будуть накладатись обмеження.

За опис політики обмеження трафіку, що містить протоколи, описані в відповідних Class Map, відповідає клас PolicyMap. Він містить такі змінні:

– змінна classMap – тип класу трафіку;
– змінна name – назва політики;
– змінна shaper – максимальна швидкість передачі даних для даної політики;
– змінна TTL – час, на який дана політика застосовується;
– змінна reverceCommands – список команд, для відміни політики, за допомогою SSH.

Клас PolicyMap містить також службові методи, спрямовані на встановлення та повернення відповідних змінних. Клас Port відповідає за опис мережевого порту маршрутизатору Cisco. Він містить такі змінні:

– змінна ifIndex – індекс порту;
– змінна snapshotList – список звітів статистики використання мережі портом в певний момент часу;
– змінна TimestampList – список детальних звітів статистики з розпізнаванням типів трафіку за допомогою механізму NBAR в певний момент часу.

Треба зазначити, що клас Port містить метод addInfo, який додає нову інформацію до списку звітів статистики використання мережі портом. Також клас Port містить службові методи, спрямовані на встановлення та повернення відповідних змінних. Клас Snapshot відповідає за статистику використання мережі певним портом в певний момент часу. Він містить такі змінні:

– змінна ifDescr – опис порту;
– змінна ifAlias – «псевдонім» порту;
– змінна ifInOctets – загальна кількість октетів, отриманих на інтерфейсі, включаючи символи кадрування;

– змінна realSpeed – швидкість середи передачі інформації.

У класі Snapshot міститься метод addInfo, який додає нову інформацію до відповідних полів звіту статистики використання мережі портом. Також клас Snapshot інші поля з даними отриманими за допомогою SNMP, які можуть бути використані в майбутньому. Окрім цього містить службові методи спрямовані на встановлення та повернення відповідних змінних.

За формування звіту статистики з розпізнаванням типів трафіку за допомогою механізму NBAR в певний момент часу відповідає клас Timestamp. Він містить такі змінні:

– змінна Time – момент часу в який був сформований звіт;
– змінна dump – список детальної інформації по кожному типу трафіку, які були розпізнані.

При цьому клас Timestamp містить метод addInfo, який додає нову інформацію до звіту з розпізнаванням типів трафіку за допомогою механізму NBAR в певний момент часу. Також клас Timestamp містить службові методи спрямовані на встановлення та повернення відповідних змінних.

За статистику з розпізнаванням типів трафіку за допомогою механізму NBAR в певний момент часу відповідає клас Traffic. Він містить такі змінні:

– змінна Time – момент часу в який була зчитана інформація;
– змінна cnpdAllStatsProtocolsName – назва типу трафіку, який був розпізнаний;
– змінна cnpdAllStatsInBytes – загальна кількість вхідних пакетів певного типу трафіку.

Треба зазначити, що клас Traffic містить метод addInfo, який додає нову інформацію до відповідних полів статистики з розпізнаванням типів трафіку за допомогою механізму NBAR в певний момент часу. Також клас Traffic містить інші поля з даними отриманими за допомогою SNMP, які можуть бути використані в майбутньому. Окрім цього він містить службові методи спрямовані на встановлення та повернення відповідних змінних [10].

У класі Logic знаходиться опис логіки моделювання алгоритму. Він містить

такі змінні:

- змінна WAN – порт маршрутизатору Cisco, який використовується для з'єднання з мережею Інтернет;
- змінна LAN – порт маршрутизатору Cisco, який використовується для з'єднання з локальною мережею;
- змінна bandwidth – швидкість середи передачі інформації;
- змінна percent – пороговий процент завантаженості каналу зв'язку, до якого політики обмеження не застосовуються;
- змінна allowedPercent – пороговий процент співвідношення протоколів, на які не накладаються обмеження до інших, після кого політики обмеження не застосовуються;
- змінна allowed – список протоколів, на які не будуть накладатись обмеження;
- змінна policyMaps – список політик, що були сформовані для маршрутизатору Cisco;
- змінна shaper – максимальна швидкість передачі даних для даної політики;
- змінна host – пристрій, трафік якого аналізується;
- змінна update – частота оновлень інформації про активність маршрутизатору Cisco;
- TTL час, на який застосовується політика обмеження;

Нижче перелічені методи, що містяться у класі Host:

- метод watch – метод, що отримує актуальні звіти з маршрутизатору Cisco та на основі їх приймає політики обмеження, якщо це доцільно;
- метод makeShaping – створює політики та класи трафіку, виконує команди для застосування на пристрої та робить список зворотних команд;
- метод getCurrentTimeUsingDate – повертає поточний час у форматі «yyyyMMddHHmmss»;

- метод isData – перевіряє назви Policy Map чи Class Map на наявність дати;
- метод makeCommand – створює список команд для застосування Policy Map на пристрої;
- метод makeReverce – створює список команд для очищення Policy Map та Class Map з маршрутизатору Cisco;
- метод checkMaps – перевіряє чи не закінчився срок дії політики;
- метод stringToDate – перетворює строку формату «yyyyMMddHHmmss» в об'єкт класу Date.

Клас Host містить також службовий метод getCurrentDate для отримання поточної дати як об'єкт класу Date. Клас SSHClient створює з'єднання SSH з заданими параметрами та передає команди на маршрутизатор Cisco і повертає результати команд отриманих з маршрутизатору Cisco. Клас SNMPClient створює з'єднання SNMP з заданими параметрами та зчитує інформацію по заданому OID і повертає результати отримані з маршрутизатору Cisco у відповідному форматі.

Одразу після запуску програми викликається метод main в класі Main, що створює об'єкт класу Config. Конструктор класу Config зчитує конфігураційний файл у форматі YAML та створює об'єкт класу Host з відповідними параметрами. Конструктор об'єкту Host отримує список портів за допомогою SNMP, очищує класи трафіку та політики обмеження, що могли залишитись при аварійному завершенні програми та створює об'єкт класу Logic з відповідними параметрами.

За допомогою SSH конструктор об'єкту Logic отримує значення пропускної здатності каналу зв'язку, описану в описі порту WAN на пристрої. Далі за допомогою SNMP формуються початкові звіти статистики для портів та детальні звіти використовуваних типів трафіку. Зчитування з SNMP відбувається з інтервалом, що був заданий в конфігураційному файлі. Далі з заданим інтервалом виконується метод watch. Цей метод отримує поточні звіти та перевіряє завантаженість каналу зв'язку. Якщо він перевантажений – програма перевіряє чи не перевантажують канал зв'язку «дозволені» протоколи. Якщо ні – формуються класи трафіку та політики обмеження, що застосовуються на пристрої та додаються

до списку застосованих політик. В кінці список застосованих політик перевіряється на наявність політик, в яких строк дії закінчився. На маршрутизатор відсилається список зворотних команд, якщо цей строк закінчився. Блок-схема алгоритму основного коду програми розпізнавання потоку даних наведений на рис.1.14.

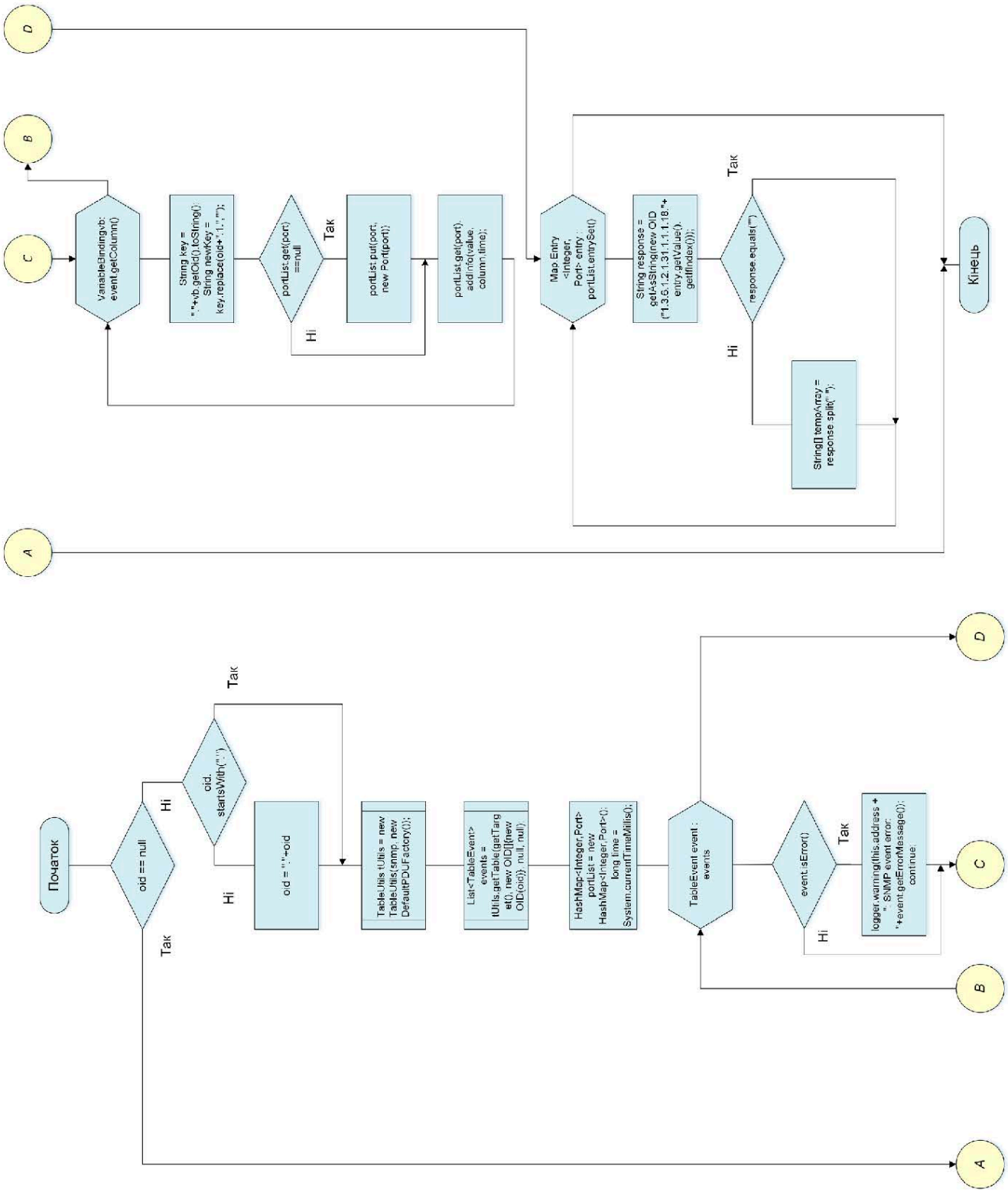


Рисунок 1.14. Блок-схема алгоритму основного коду програми

1.3.3 Реалізація інтерфейсу програми розпізнавання потоку даних

Створюваний програмний продукт буде реалізовувати розроблений вище алгоритм оптимізації мережевого трафіку. Цей програмний продукт надасть можливість оптимізувати трафік на маршрутизаторі Cisco, описаному в файлі конфігурації в форматі YAML. Приклад такого файлу конфігурації наведено на рис. 1.15.

```
host:
  name: 172.16.0.130
  user: cisco
  password: cisco123
  community: private
  update: 5
  TTL: 15
  bandwidth_usage: 0.8
  allowed_percent: 0.5
  protocols:
    - snmp
    - icmp
    - dns
```

Рисунок 1.15. Вміст файлу конфігурації для маршрутизатору Cisco

Відповідно до наведеного вище прикладу у файлі конфігурації зазначені такі параметри:

- name – назва маршрутизатору Cisco. Може буди доменним ім'ям, або IP адресою;
- user – ім'я користувача потрібне для з'єднання по SSH;
- password - пароль користувача потрібний для з'єднання по SSH;
- community – «рядок спільноти», що використовується для з'єднання по SNMPv1 або SNMPv2;
- update – частота оновлень інформації про активність маршрутизатору Cisco;
- TTL – час, на який застосовується політика обмеження;
- bandwidth_usage – пороговий процент завантаженості каналу зв'язку, до якого політики обмеження не застосовуються;

– allowed_percent – пороговий процент співвідношення протоколів, на які не накладаються обмеження до інших, після кого політики обмеження не застосовуються;
– protocols – список протоколів, на які не будуть накладатись обмеження.

Виведені у консоль програми часові параметри задаються в секундах. Програма виводить інформацію про поточні значення навантажень, звіт про використані типи трафіку та команди, що відправляються на маршрутизатор Cisco в stdout. Файл буде містити логування всіх даних при виведенні потоку в файл.

```
interface FastEthernet0/0
 description WAN 100000000
 bandwidth qos-reference 10000000
 ip address 172.16.0.130 255.255.255.0
 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 private RW
snmp-server host 172.16.0.141 version 2c private
line vty 0 4
 exec-timeout 60 0
 privilege level 15
 logging synchronous
 transport input ssh
!
username cisco privilege 15 password 7 02050D4808095E731F
ip ssh version 2
```

Рисунок 1.16. Лістинг виведення команди show running-config

1.3.4 Тестування моделі розпізнавання потоку даних

Розроблена програма потребує первинного налаштування мережевого обладнання. Усі тести проводились за допомогою маршрутизатора Cisco ASR1001-X. Для коректної роботи програми маршрутизатор має бути попередньо налаштований. Окрім базових налаштувань повинні бути налаштовані описи портів, ввімкнений SSH-сервер та SNMP-сервер, ввімкнений механізм NBAR. На рис. 1.16 зображений приклад виведення команди show running-config. На рис. 1.17 показано загальну структуру мережі для тестування.

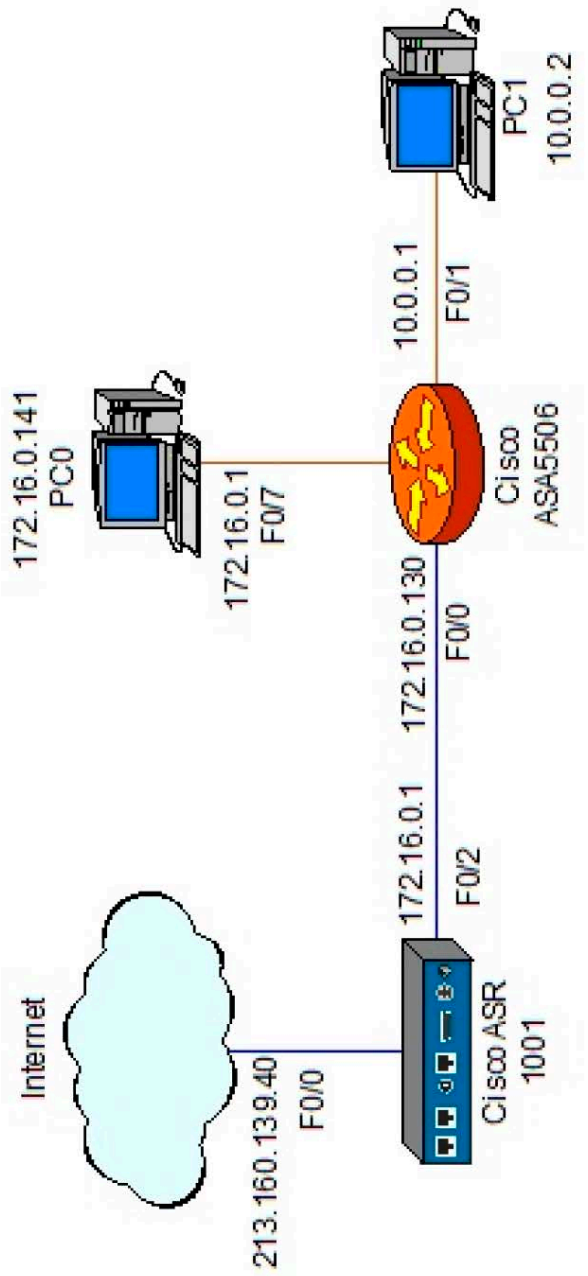


Рисунок 1.17. Структура мережі для тестування розробленої моделі

У тестованій моделі мережі організовано роботу двох робочих станцій з встановленими ОС Windows 10, маршрутизатору Cisco ASR1001 та міжмережевого екрану ASA5506.

```
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 1.18. Лістинг ініціалізації програми та очищення політик

```
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 1.20. Лістинг звіту з класифікацією трафіку мережі

```
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 1.22. Лістинг сформованих команд та виведення логу SSH-з'єднання

Для тестування розроблених програмних засобів у реальних умовах було проведене порівняння швидкості копіювання файлу образу (iso) в різних режимах роботи мережі. На рис. 1.23 показано швидкість копіювання даних користувача з PC0 до PC1 до та після включення розробленої моделі. Як видно з результатів тестування, швидкість для протоколу http була зменшена, проте це ніяк не вплинуло на швидкість передачі протоколів, описаних в файлі конфігурації.

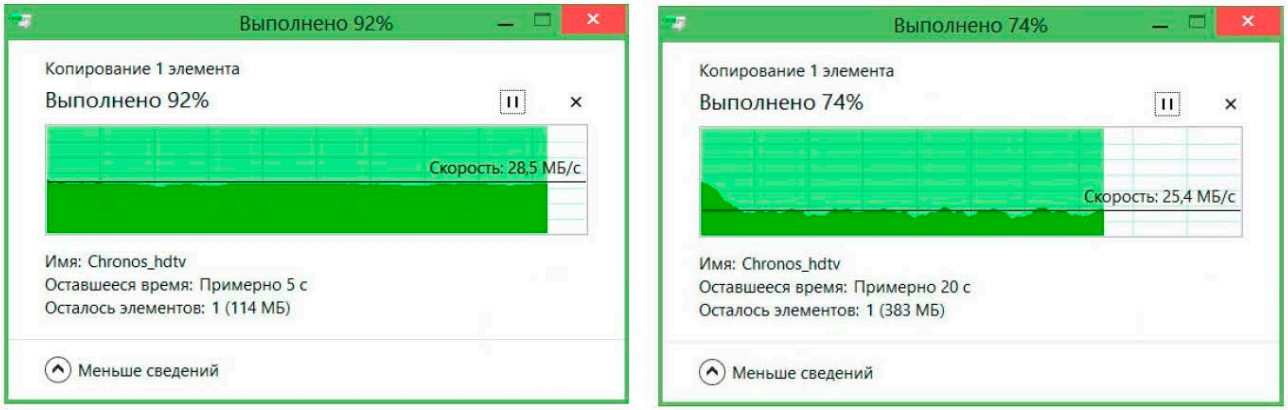


Рисунок 1.23. Оцінка швидкості копіювання даних користувача

Навантаження на канал зв'язку до застосування політики обмеження

1.3.5 Аналіз результатів моделювання та ефективності алгоритму

Розроблена програма надає можливість в автоматичному режимі відслідковувати навантаження в мережі, а також застосовувати політики обмеження на певні типи трафіку. Класифікація трафіку відбувається за допомогою технології NBAR. Програма підтримує можливість опису конфігурації у форматі YAML, реалізовано можливість очищення файлу конфігурації маршрутизатору Cisco після аварійного завершення програми. При обмеженні швидкості передачі даних для http-трафіку немає впливу на передачу інших типів даних. Запропонований алгоритм дозволяє істотно скоротити похибки, зумовлені людським фактором, кількість персоналу, а також автоматизувати рішення задачі обмеження швидкості для непріоритетних видів трафіку [11].

При використанні розробленого програмного забезпечення для розпізнавання потоку даних мережі проведено аналіз ефективності запропонованого алгоритму порівняно із ручним налаштуванням політик обмеження адміністратором мережі. При ручному налаштуванні системний адміністратор повинен мати змогу під'єднатись до мережевого обладнання та проаналізувати усю доступну інформацію, написати політики обмеження та прослідкувати за змінами і отриманим результатом. У випадку використання запропонованого алгоритму усі ці дії автоматизуються і виконуються в автоматичному режимі розробленою програмою. Це дозволяє зменшити помилки, зумовлені людським фактором, зменшити витрати на роботу спеціалістів, що повинні працювати на об'єкті, та збільшити відмовостійкість мережі [12]. Розроблена програма обмежує швидкість лише у тих випадках, коли процент використання каналу зв'язку перевищує заданий процент, та кількість трафіку, що можна обмежити в швидкості, перевищує заданий поріг. Це дозволяє обмежувати швидкість тільки тоді, коли це потрібно, і таким чином завжди мати резерв для важливих протоколів, відсутність яких може вплинути на роботу організації. В умовах недоступності високошвидкісних каналів передачі даних в деяких регіонах обмеження швидкості трафіку є оптимальним варіантом. Також слід зауважити, що при обмеженні швидкості доступність сервісу не втрачається.

2 ЕКОНОМІЧНА ЧАСТИНА

2.1 Резюме

Тема дипломного проекту Розробка моделі розпізнавання потоку даних мережі для маршрутизаторів Cisco. При моделюванні алгоритму розпізнавання потоку даних мережі для маршрутизаторів Cisco було розроблено програмне забезпечення мовою Java. Програма дозволяє обмежувати типи трафіку, які дають навантаження на канал зв'язку та не входять в список описаних в конфігураційному файлі типів трафіку.

Ефективність кожного програмного продукту визначається його якістю та ефективністю процесу розробки. Якість ПП визначається наступними складовими: з точки зору користувача; з позиції використання ресурсів; виконання вимог до програмного забезпечення.

Оцінка якості програмного продукту з точки зору користувача визначається необхідним на стадії функціонування розміром оперативної пам'яті ЕОТ, витратами машинного часу, пропускною спроможністю каналів передачі даних. Оцінка якості програмного продукту включає визначення трудомісткості і вартості його створення.

2.2 Визначення трудомісткості розробки програмного забезпечення

Тривалість розробки програмного продукту залежить від його обсягу, трудомісткості розробки, кваліфікації виконавців, а також планових термінів, визначених умовами ринку. Методом структурної аналогії по відповідних каталогах аналогів програмного забезпечення визначається обсяг програмних засобів, у тисячах умовних машинних команд програми 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 імітаційного моделювання	7800 – 8800

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 ПП, що містить Vo в умовних машинних командах, трудомісткості визначати на основі табл.2.2

Таблиця.2.2

Обсяг ПП, 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 0,7 = 160,3 (люд/годин).

Трудомісткість програмного продукту визначається по кожному етапу розробки окремо на підставі трудомісткості аналога з урахуванням складності

розробки, ступеня новизни і ступеня використання в розробці стандартних модулів на підставі формул:

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

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

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

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 ПО	1,75 – 1,2
Б	ПО – розвиток визначеного параметричного ряду	1,0 – 0,8
В	ПО 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 ПО 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

$$Т_{тз}=Та*L_1*К_н=160,3 *0,12*0,8= 15,38 \text{ (люд/годин)} \quad (2.4)$$

Трудомісткість розробки технічного проекту

$$Т_{тп}=Та*L_2*К_н =160,3 *0,11*0,8 = 14,11 \text{ (люд/годин)} \quad (2.5)$$

Трудомісткість розробки робочого проекту

$$Т_{рп}=Та*L_3*К_н*К_т=160,3 *0,61*0,8*0,8= 62,58 \text{ (люд/годин)} \quad (2.6)$$

Для подальших розрахунків визначили кількість папера, витраченого на кожен етап: технічне завдання $N_{тз}=3$ (стр), розробка ТП $N_{тп}=9$(стр), розробка робочого проекту $N_{рп}=14$ (стр), пояснювальна записка відповідно $N_{пз}=36$ (стр)

Розрахунок зведений у таблицю 2.6.

Таблиця 2.6 Розрахунок трудомісткості ПП

Найменування етапів	**Розрахунок, годин.**		
1	2	3	4
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 *9=6,3	$T_{нк}$=0,15*$N_{тп}$=0,15*9=1.4
3.Розробка РП	$T_{Ррп}$= 62,58	$T_{кк}$=0,7*$N_{рп}$=0,7 *14=9,8	$T_{нк}$=0,15*$N_{рп}$=0,15*14=2,1
4.Розробка ПЗ	$T_{пз}$=1,5**$N_{пз}$= 1,5*36 =54	$T_{кк}$=0,7*$N_{тз}$=0,7 *36=25,2	$T_{нк}$=0,15*$N_{пз}$=0,15*36 =5,4
Усього, в т.ч.:	198,9		
- на розробку	$\Sigma T_р$=146.1		
- контроль керівника		$\Sigma T_{кк}$=43,4	
- нормоконтроль			$\Sigma T_{нк}$=9,4

2.3 Розрахунок ціни програмного продукту

У цьому розділі для визначення ціни розраховуємо основну заробітну плату виконавців, матеріальні витрати, вартість машино – години і витрати на розробку ПО. Розрахунок основної заробітної плати виконавців приведений у таблиці 6.7. Відповідно до статті 8 «Закону про Державний бюджет України на 2023» встановлено мінімальну заробітну плату у місячному розмірі з 1 січня 2023 року - 6700 гривень; мінімальну погодинну тарифну ставку – 40.46 грн.

Таблиця 2.7 Розрахунок основної заробітної плати виконавців

Найменування робіт	**Трудомісткість робіт, години**	**Погодинна тарифна ставка, грн.**	**Розрахунок, грн.**
1.Розробка ПП	146,1	40,46	5911,21
2.Контроль керівника	43,4	60,10	2608,34
3.Нормоконтроль	9,4	60,10	564,94
Усього	-	-	ΣЗо= 9084,49

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

Таблиця 2.8 Розрахунок матеріальних витрат на розробку ПО

Найменування матеріальних витрат	Тип, модель	Кількість	Ціна одиниці, грн.	Вартість, грн.
Папір	Лист А4	60	3.0	180,0
Разом	-	-	-	$В_{мі}$=180,0
Транспортно – заготівельні Витрати (10%)				$Втр\_з = 0,1 \times Вм1$ = 0,1*180=18,00
Усього				$В_м$=$В_{мі}$+$В_{тр\ з}$= 198.00

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	198.00	$В_м$ (див. табл. 2.8)
2. Основна заробітна плата	9084,49	$З_о$ (див. табл. 2.7.)
3.Додаткова заробітна плата	908,49	$Зд = 0,1 \times Зо$ = 9084,89*0,1
4.Відрахування до єдиного фонду соціального внеску	2198,46	$Вє.с.в. = 0,22 \times (Зо + Зд)$ =0,22* (9084,49+908,49)
5. Накладні витрати	3633,79	$Внак. = 0,4 \times Зо$ =0,4 * 9084,49
6. Повна собівартість	16023,24	$С_{пов}$= $В_м$ +$З_о$ + $З_д$ +Вє.с.в. + Внак.= 198.00+9084,49+908,49+2198,46+ 3633,79

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

$$П=(С_п*Р)/100=(16023,24*10)/100=1602,32грн \qquad (2.7)$$

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

Оптова ціна (кошторисна вартість) визначається по формулі:

$$Ц_о = С_п+П = 16023,24+1602,32= 17625,56 грн; \qquad (2.8)$$

Виходячи з отриманих даних, ціна реалізації розробленого програмного продукту на основі наступної формули, становитиме:

$$Ц_р=Ц_о+ПДВ = 17625,56+17625,56 *0.2 = 21150,67 грн; \qquad (2.9)$$

3 ОХОРОНА ПРАЦІ

Однією із характерних особливостей сучасного розвитку суспільства є зростання сфер діяльності людини, в яких використовуються інформаційні технології. Широке розповсюдження отримали персональні комп'ютери. Однак їх використання загострило проблеми збереження власного та суспільного здоров'я, вимагає удосконалення існуючих та розробки нових підходів до організації робочих місць, проведення профілактичних заходів для запобігання розвитку негативних наслідків впливу ПК на здоров'я користувачів.

3.1 Аналіз небезпечних і шкідливих факторів, що впливають на програміста

В даному розділі дипломного проекту розглядається питання охорони праці програміста. Оператори і програмісти зіштовхуються із впливом таких фізично небезпечних і шкідливих виробничих факторів, як підвищений рівень шуму, підвищена температура зовнішнього середовища, відсутність або недостатня освітленість робочої зони, електричний струм, статична електрика тощо.

3.2 Гігієнічні вимоги до виробничого середовища

На робочому місці програміста повинні бути створені умови для безпечної та високопродуктивної праці.

3.2.1 Вимоги до приміщення

Об'ємно-планувальні рішення будівель та приміщень для роботи з ВДТ мають відповідати вимогам ДСанПІН 3.3.2.007-98 «Гігієнічні вимоги до організації роботи з візуальними дисплейними терміналами електронно-обчислювальних машин». Розміщення робочих місць з ВДТ ЕОМ і ПЕОМ у підвальних приміщеннях, на цокольних поверхах заборонено. Площа на одне робоче місце становить не менше 6,0 м^2, а об'єм – не менше ніж 20,0 м^3. У приміщеннях слід щоденно робити вологе прибирання. Вони повинні бути оснащені аптечками першої медичної допомоги. В будівлях мають бути обладнанні побутові приміщення для відпочинку.

3.2.2 Освітлення

У приміщеннях, призначених для роботи з відео терміналами (ПК) , доцільно, щоб вікна були орієнтовані на північ або північний захід. На вікнах повинні бути штора або жалюзі, що регулюють рівень освітленості і захищають від прямого влучення сонячних променів на робоче місце. При кольоровому оформленні виробничих і допоміжних приміщень необхідно враховувати орієнтацію їхніх вікон стосовно частин світу і використовувати гармонійне сполучення кольорів. Для стін і робочих поверхонь використовують мало насичені (основні) кольори, для невеликих помешкань або ділянок, що рідко потрапляють у поле зору працюючих, а також для створення контрастності – кольори середньої насиченості (допоміжні), для маленьких по площі поверхонь – насичені (акценти) – як функціональне фарбування. Стелі у всіх приміщеннях повинні бути білими. Поверхні устаткування в приміщеннях повинні бути матовими або напівматовими, для виключення випадку відблисків світла в очі працюючого, а стіни бути пофарбованими фарбами пастельних тонів.

Для штучного освітлення у приміщенні використовуються люмінесцентні лампи типу ЛБ, які в порівнянні з лампами розжарювання мають ряд істотних переваг. Допускається застосування ламп розжарювання у світильниках місцевого освітлення. Нормами для даних робіт установлена необхідна освітленість робочого місця E_H=300 лк.

3.2.3 Шум

Оптимальні показники рівня шумів у робочих приміщеннях конструкторських бюро, кабінетах розраховувачів, програмістів визначаються за ГОСТ 12.1.003-83.

Припустимий рівень шуму при розумовій праці, що вимагає зосередженості для програміста 50 дБ. Для зменшення шуму й вібрації в приміщенні устаткування, апарати й прилади встановлюються на спеціальні фундаменти й прокладки, що амортизують. Якщо стіни й стелі приміщення є джерелами шумоутворення, вони повинні бути облицьовані звуковбирним матеріалом.

3.2.4 Вимоги до організації робочого місця працівника

Обладнання і організація робочого місця з ПК мають забезпечувати відповідність конструкцій всіх елементів робочого місця та їх взаємного розташування, ергономічним вимогам з урахуванням характеру і особливостей трудової діяльності .

Конструкція робочого місця й взаємне розташування всіх його елементів (сидіння, органи керування, засобу відображення інформації) відповідають антропометричним, фізіологічним і психологічним вимогам, а також характеру роботи. Конструкція робочих меблів повинна забезпечувати можливість індивідуального регулювання відповідно росту працюючих для підтримки зручної пози. Робочий стіл повинен бути пофарбований матовою фарбою. Дисплей розташований так, що його верхній край перебуває на рівні очей на відстані близько 70 см, що укладається в у припустимі рамки від 60 до 90 см. Частота мерехтіння екрана $f_{мер}$=100 Гц, що відповідає умові $f_{мер}$>70 Гц.

Робоче місце розташоване перпендикулярно віконним прорізам, це зроблено з тією метою, щоб виключити пряму й відбиту мерехтливість екрана від вікон і приладів штучного освітлення, якими є лампи накалювання.

Працюючі з ВДТ підлягають обов'язковим медичним оглядам: попереднім – при влаштуванні на роботу і періодичним – протягом трудової діяльності, відповідно до наказу МЗ України № 45.

Основними критеріями оцінки придатності до роботи з ВДТ мають бути показники стану органів зору: гострота зору, показники рефракції, стану бінокулярного апарату ока тощо. При цьому необхідно враховувати також стан організму в цілому.

Виконання вимог в комплексі з практичним здійсненням первинних та спеціальних заходів повинно стати нормою діяльності всіх фахівців, безпосередньо пов'язаних з виробничими колективами.

3.2.5 Мікроклімат

У виробничих приміщеннях на робочих місцях мають забезпечуватись оптимальні значення параметрів мікроклімату: температури, відносної вологості й

рухливості повітря – ДСН 3.3.6.042-99 «Санітарні норми мікроклімату виробничих приміщень».

Параметри мікроклімату	значення параметрі	
	Взимку	влітку
Температура, C^0	22-24	23-25
Відносна вологість, %	40-60	40-60
Швидкість руху повітря, м/с	0,1	0,1-0,2

Для підтримки в приміщеннях нормального, що відповідає гігієнічним вимогам складу повітря, видалення з нього шкідливих газів, пилу використовують вентиляцію.

3.2.6 Електробезпека

Електроустановки повинні відповідати вимогам Правил пристрою електроустановок (ПУЕ), Правил технічної експлуатації споживачів (ПТЕ), Правил техніки безпеки під час експлуатації електроустановок (ПТБ) і інших нормативних документів, що діють.

З`єднання, відгалуження і закінчення проводів і кабелів повинні здійснюватися за допомогою зварки, спайки, опресовування або спеціальних затисків. Виконувати з`єднання жил проводів і кабелів методом скручування забороняється.

Не допускається:

- прокладка проводів і кабелів через складські приміщення, пожежонебезпечні та вибухонебезпечні зони;
- експлуатація проводів і кабелів з пошкодженою ізоляцією;
- залишати під напругою дроти і кабель з неізольованими струмопровідними жилами;
- використовувати саморобні (кустарного виробництва) подовжувачі, які не відповідають вимогам ПУЭ;
- використовувати для обігріву приміщення нестандартне (саморобне) електронагрівальне устаткування або лампи розжарювання;

– користуватися пошкодженими розетками, вимикачами та іншими електроприладами;
– використовувати для електромереж радіо і телефонний провід;
– залишати електроустаткування під напругою без нагляду.

3.3 Пожежна безпека

Всі приміщення повинні бути забезпечені первинними засобами пожежогасіння: пожежним водопостачанням (пожежні крани ПК), пожежні щити з набором пожежного інструменту, вуглекислотними або порошковими вогнегасниками.

Рисунок 3.1. Пожежна безпека офісів

У випадку виникнення пожежі необхідно відключити електроживлення, викликати по телефону 101 пожежну команду, евакуювати людей із приміщення відповідно до плану евакуації і приступити до ліквідації пожежі.

ВИСНОВКИ

Дипломна робота передбачала розробку моделі розпізнавання потоку даних мережі для маршрутизаторів Cisco. Вирішено задачу оптимізації мережевого трафіку за допомогою розпізнавання трафіку та його типізації на основі механізму розпізнавання мережевих додатків, що проводить глибокий аналіз пакетів мережі. Виявлено наступні переваги використання технології NBAR: велика база протоколів, що можуть бути розпізнані, підтримується багатьма пристроями Cisco, інтеграція з політиками обмеження QoS, можливість отримання метрик за допомогою протоколу SNMP. У якості показника навантаження каналу зв'язку обрано кількість вхідних октетів на порт WAN. При навантаженні аналізуються звіти NBAR, отримані за допомогою протоколу SNMP та створюються політики обмеження, які являють собою список команд для відправки на маршрутизатор Cisco. Доставка команд відбувається за допомогою протоколу SSH. Це дозволяє встановити безпечне з'єднання та відстежувати результати виконання команд. Результати програми надсилаються в stdout, тож можуть бути перенаправленні в файл для збереження логів. Також при аварійному завершенні програми передбачено очищення файлу конфігурації.

При моделюванні алгоритму розпізнавання потоку даних мережі для маршрутизаторів Cisco було розроблено програмне забезпечення мовою Java. Програма дозволяє обмежувати типи трафіку, які дають навантаження на канал зв'язку та не входять в список описаних в конфігураційному файлі типів трафіку. Проведений порівняльний аналіз запропонованого алгоритму та програмного забезпечення з ручним налаштуванням мережевого маршрутизатору Cisco дозволив виявити, що робота алгоритму автоматизує деякі задачі мережевого адміністратору. Це дозволяє зменшити кількість ситуацій, коли мережевий адміністратор допускає помилки в налаштуванні мережевих пристроїв. Також це дозволяє за відсутності високошвидкісного каналу зв'язку забезпечити безперебійну роботу критично важливих сервісів, без яких роботу об'єкту може бути призупинено.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Воробієнко П. Телекомунікаційні та інформаційні мережі / П.П. Воробієнко, Л.А. Нікітюк, П.І. Резніченко. – К.: Саміт-книга, 2010. – 635 с.
2. Цвіркун, Л.І. Розробка програмного забезпечення комп'ютерних систем. Програмування: навч. посібник / Л.І. Цвіркун, А.А. Євстігнєєва, Я.В. Панферова, під заг. ред. Л.І. Цвіркуна. – 3-є вид., випр. – Д.: Національний гірничий університет, 2016. – 223 с.
3. Комп'ютерні мережі: [навчальний посібник] / А. Г. Микитишин, М. М. Митник, П. Д. Стухляк, В. В. Пасічник. – Львів: «Магнолія 2006», 2013
4. Олифер В. Г., Олифер Н. А. Компьютерные сети. Принципы, технологии, протоколы. – 4-е изд. – СПб.: Питер, 2010.
5. Жураковский Б. Ю. Комп'ютерні мережі. Частина 1. Навчальний посібник [Електронний ресурс] / Б. Ю. Жураковский, І. О. Зенів // КПІ ім. Ігоря Сікорського. – 2020. – 336 с.
6. Блох, Дж. Java: эффективное программирование / Дж. Блох. – М.: Диалектика, 2019. – 464 с.
7. Блинов И.Н., Романчик В. С. Java. Методы программирования : уч.-мет. пособие / И. Н. Блинов, В. С. Романчик: издательство "Четыре четверти", 2013. 896 с.
8. Васильев А. Н. Java. Объектно-ориентированное программирование: Учебное пособие. СПб.: Питер. – 2011. – 400 с.
9. Технології створення програмних продуктів та інформаційних систем: навч. посібник / М. Ю. Карпенко, Н. О. Манакова, І. О. Гавриленко ; Харків. нац.ун-т ім. О. М. Бекетова. – Харків : ХНУМГ ім. О. М. Бекетова, 2017. – 93 с.
10. Хорстманн, К. С., Корнелл, Г. Библиотека профессионала. Java 2: Том 1. Основы. 8-е изд. М. : Вильямс. – 2013. – 816 с.
11. Thomas Barnett, Jr. 2018 Complete VNI Forecast Update – What's Trending? / Thomas Barnett, Jr.. // Cisco public. – 2018.
12. Cisco Visual Networking Index: Forecast and Trends, 2017–2022. // Cisco ublic. – 2018. – С. 1–38
13. Classifying Network Traffic Using NBAR. // Cisco Press. – 2018. – С. 8–16.

ДОДАТОК А. Лістинг класу logic для опису логіки спостереження мережевої активності (мова Java)

```
import java.util.*;
import java.text.DateFormat;
import java.text.SimpleDateFormat;
/* Class for describing logic of watching network activity */
public class Logic extends Thread{
   /* WAN port of host.  */
   Port WAN;
   /* Bandwidth of link channel.   */
   Long bandwidth;
   /* LAN port of host.  */
   Port LAN;
   /* Allowed percent of channel load.  */
   double percent;
   /* Percent of allowed protocols. When allowed protocols uses much bigger part
      of bandwidth it's no need to block something.*/
   double allowedPercent;
   /* List of allowed protocols.  */
   ArrayList<String> allowed;
   /* List of Policy Maps created for host.*/
   ArrayList<PolicyMap> policyMaps;
   /* Speed for some protocol after making a shaping. */
   long shaper;
   /*  Host for which making protocol analyzing.   */
   Host host;
   /* Update rate in seconds when watch run.  */
   int update;
   /* TTL for policy map.  */
   int TTL;
   /* Constructor for logic of watching network activity.
    * @param host host for which making protocol analyzing.
    * @param WAN WAN port of host.
    * @param LAN LAN port of host.
    * @param percent allowed percent of channel load.
    * @param allowedPercent percent of allowed protocols. When allowed protocols uses much
bigger part of bandwidth it's no need to block something.
    * @param TTL TTL for policy map.
    * @param update update rate in seconds when watch run.  */
   public Logic(Host host, Port WAN, Port LAN, double percent, double allowedPercent, int TTL,
int update) {
      this.host = host;
      this.WAN = WAN;
      this.LAN = LAN;
      this.percent = percent; // Завантаженість каналу
      allowed = host.protocols;
      this.allowedPercent = allowedPercent; // Захист від блокування дозволених
                                            // (коли дозволені, "з'їдають" трафік)
      this.TTL = TTL;
      this.update = update;
      policyMaps = new ArrayList<PolicyMap>();
      SSHClient ssh = new SSHClient(host.getUser(), host.getName(), host.getPassword());
      ArrayList<String> result = ssh.executeCommand("show running-config interface " +
WAN.snapshotList.get(WAN.snapshotList.size() - 1).getIfDescr());
```

```
        for (String str : result) {
            if (str.contains(" description WAN ")) {
                bandwidth = Long.parseLong(str.replaceAll(" description WAN ", ""))/(8*1000);
                shaper = Long.parseLong(str.replaceAll(" description WAN ", ""))/10;
                break;
            }
        }
        try {
            SNMPClient snmpClient = new SNMPClient(host.getName(),
host.getCommunity(), "161");
            snmpClient.start();
            snmpClient.updatePort(WAN, ".1.3.6.1.2.1.2.2");
            snmpClient.stop();
            snmpClient.start();
            snmpClient.updateTrafficTable(".1.3.6.1.4.1.9.9.244.1.2.1", host.ports);
            snmpClient.stop();
            Thread.sleep(5000);
            snmpClient.start();
            snmpClient.updatePort(WAN, ".1.3.6.1.2.1.2.2");
            snmpClient.stop();
            snmpClient.start();
            snmpClient.updateTrafficTable(".1.3.6.1.4.1.9.9.244.1.2.1", host.ports);
            snmpClient.stop();
            Thread.sleep(5000);
            while (true) {
                try {
                    watch();
                    Thread.sleep(update);
                } catch (Exception ee) {
                    System.out.println(ee);
                }
            }
        } catch (Exception ee) {
            System.out.println(ee);
        }
    }
    /* Method to watch network activity on the host.  */
    private void watch() {
        HashMap<Integer, Traffic> lastDump =
WAN.getTimestampList().get(WAN.getTimestampList().size() - 1).getDump();
        HashMap<Integer, Traffic> preLastDump =
WAN.getTimestampList().get(WAN.getTimestampList().size() - 2).getDump();
        SNMPClient snmp = new SNMPClient(host.getName(), host.getCommunity(), "161");
        try {
            snmp.start();
            snmp.updatePort(WAN, ".1.3.6.1.2.1.2.2");
            snmp.stop();
        } catch (Exception ee) {
            System.out.println(ee);
        }
        long lastUsageTime = WAN.snapshotList.get(WAN.snapshotList.size() - 1).Time;
        long preLastUsageTime = WAN.snapshotList.get(WAN.snapshotList.size() - 2).Time;
        long lastUsageOctets = WAN.snapshotList.get(WAN.snapshotList.size() - 1).getIfInOctets();
        long preLastUsageOctets =
WAN.snapshotList.get(WAN.snapshotList.size() - 2).getIfInOctets();
```

```
        double currentBandwidthUsage = (double)(lastUsageOctets - preLastUsageOctets) /
(double)(lastUsageTime - preLastUsageTime);
        System.out.println("currentBandwidthUsage "+ currentBandwidthUsage);
        System.out.println("Bandwidth "+ bandwidth);
        System.out.println("Current percent "+ currentBandwidthUsage / bandwidth);
        System.out.println("Allowed percent "+ percent);
        if (currentBandwidthUsage / bandwidth > percent) {
            System.out.println("NBAR analysis");
            HashMap<Integer, Long> delta = new HashMap<Integer, Long>();
            long maxDelta = 0;
            String deltaName = "";
            long deltaSum = 0;
            long maxAllowed = 0;
            long allowedSum = 0;
            for (HashMap.Entry<Integer, Traffic> entry : lastDump.entrySet()) {
                int index = entry.getKey();
                if (preLastDump.get(index) == null) delta.put(index,
entry.getValue().getCnpdAllStatsInBytes());
                else
                    delta.put(index, entry.getValue().getCnpdAllStatsInBytes() -
preLastDump.get(index).getCnpdAllStatsInBytes());
                System.out.println(lastDump.get(index).getCnpdAllStatsProtocolsName()+
"\n Last usage = "+entry.getValue().getCnpdAllStatsInBytes()+"\n Prelast usage = "+
preLastDump.get(index).getCnpdAllStatsInBytes()+"\n delta = " + delta.get(index));
                if (maxDelta < delta.get(index)) {
                    if (!allowed.contains(lastDump.get(index).getCnpdAllStatsProtocolsName())) {
                        maxDelta = delta.get(index);
                        deltaSum += maxDelta;
                        deltaName = lastDump.get(index).getCnpdAllStatsProtocolsName();
                    } else {
                        maxAllowed = delta.get(index);
                        allowedSum += delta.get(index);
                    }
                }
            }
            // Треба знайти суму дозволених и недозволених октетів і блокувати
            if (maxDelta != 0) {
                if ((allowedSum / (deltaSum + allowedSum)) < allowedPercent) {
                    System.out.println("Shaping on " + deltaName);
                    makeShaping(deltaName, TTL);
                }
            }
        }
        checkMaps();
System.out.println("_____________________________________________________________");
    }
    /* Method to make the commands for shaping and send them to host.
     * @param deltaName name of protocol.
     * @param ttl time to live of shaping.*/
    private void makeShaping(String deltaName, int ttl) {
        String Time = getCurrentTimeUsingDate();
        ClassMap CMAP = new ClassMap("match-all", "CMAP" + Time, deltaName);
        PolicyMap PMAP = new PolicyMap("PMAP" + Time, CMAP, shaper);
        SSHClient ssh = new SSHClient(host.getUser(), host.getName(), host.getPassword());
        try {
            ArrayList<String> commands = makeCommand(PMAP);
```

```
            System.out.println("Straight commands: ");
            for ( String command: commands
                ) {
               System.out.println(command);
            }
            ssh.RunCommands(commands);
        } catch (Exception ee) {
            System.out.println(ee);
        }
        PMAP.setReverceCommands(makeReverse(PMAP));
        PMAP.setTTL(ttl);
        policyMaps.add(PMAP);
    }
    /* Method to get string of current time.
     * @return string like "yyyyMMddHHmmss".*/
    public String getCurrentTimeUsingDate() {
        Date date = new Date();
        String strDateFormat = "yyyyMMddHHmmss";
        DateFormat dateFormat = new SimpleDateFormat(strDateFormat);
        return dateFormat.format(date);
    }
    /* Method to get current date.
     * @return object Date with current parameters.
     */
    public Date getCurrentDate() {
        Date date = new Date();
        return date;
    }
    /* Method to make straight command to set CMAP and PMAP.
     * @param PMAP PMAP which will be convert into commands.
     * @return list of commands */
    private ArrayList<String> makeCommand(PolicyMap PMAP) {
        ArrayList<String> commands = new ArrayList<String>();
        commands.add("configure terminal");
        ClassMap CMAP = PMAP.getClassMap();
        commands.add("class-map " + CMAP.getMatch() + " " + CMAP.getName());
        commands.add("match protocol " + CMAP.getProtocol());
        commands.add("exit");
        commands.add("policy-map " + PMAP.getName());
        commands.add("class " + CMAP.getName());
        commands.add("police " + PMAP.getShaper() +
 " conform-action transmit exceed-action drop");
        commands.add("exit");
        commands.add("exit");
        commands.add("interface " +
LAN.snapshotList.get(LAN.snapshotList.size() - 1).getIfDescr());
        commands.add("service-policy output " + PMAP.getName());
        commands.add("exit");
        commands.add("exit");
        return commands;
    }
    /* Method to make reverse command to set CMAP and PMAP.
     * @param PMAP PMAP which will be convert into commands.
     * @return list of commands.*/
    private ArrayList<String> makeReverse(PolicyMap PMAP) {
        ArrayList<String> commands = new ArrayList<String>();
```

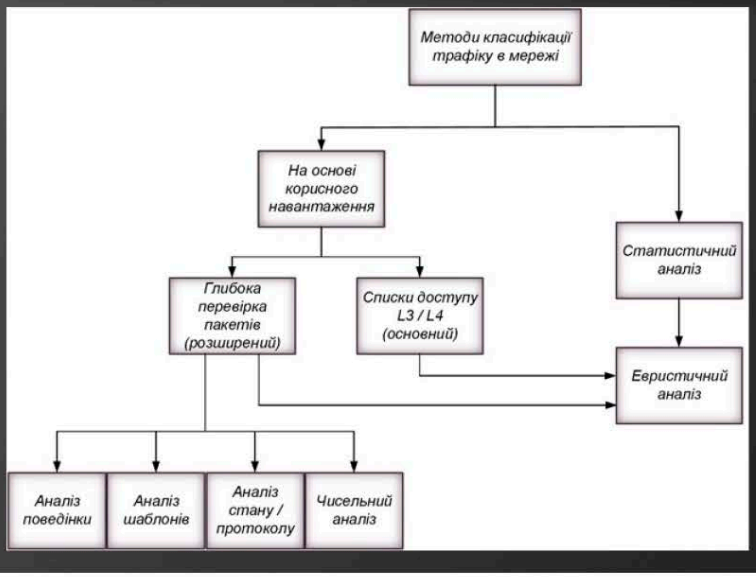
```
        commands.add("configure terminal");
        ClassMap CMAP = PMAP.getClassMap();
        commands.add("no class-map " + CMAP.getMatch() + " " + CMAP.getName());
        commands.add("no policy-map " + PMAP.getName());
        commands.add("interface " +
LAN.snapshotList.get(LAN.snapshotList.size() - 1).getIfDescr());
        commands.add("no service-policy output " + PMAP.getName());
        commands.add("exit");
        commands.add("exit");
        return commands;
    }
    /* Method to check if PMAP ttl expired. */
    public void checkMaps() {
        if (policyMaps.size() != 0) {
            Iterator<PolicyMap> p = policyMaps.iterator();
            while (p.hasNext()) {
                PolicyMap PMAP = p.next();
                String name = PMAP.getName();
                System.out.println("Cheking "+name);
                Date Time = stringToDate(name.replace("PMAP", ""));
                Calendar cal = Calendar.getInstance();
                cal.setTime(Time);
                cal.add(Calendar.SECOND, PMAP.getTTL());
                Date newTime = cal.getTime();
                if (getCurrentDate().after(newTime)) {
                    SSHClient ssh = new SSHClient(host.getUser(), host.getName(), host.getPassword());
                    try {
                        System.out.println("!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!Reverse commands: ");
                        for ( String command: PMAP.getReverceCommands()
                        ) {
                            System.out.println(command);
                        }
                        ssh.RunCommands(PMAP.getReverceCommands());
                    } catch (Exception ee) {
                        System.out.println(ee);
                    }
                    p.remove();
                }
            }
        }
    }
    /* Method to covert string to date format.
     * @param Time time in string format.
     * @return time in DateFormat.*/
    public Date stringToDate(String Time) {
        Date date = new Date();
        DateFormat format = new SimpleDateFormat("yyyyMMddHHmmss");
        try {
            date = format.parse(Time);
        } catch (Exception ee) {
            System.out.println(ee);
        }
        return date;
    }
}
```

ДОДАТОК Б

Слайди мультимедійної презентації

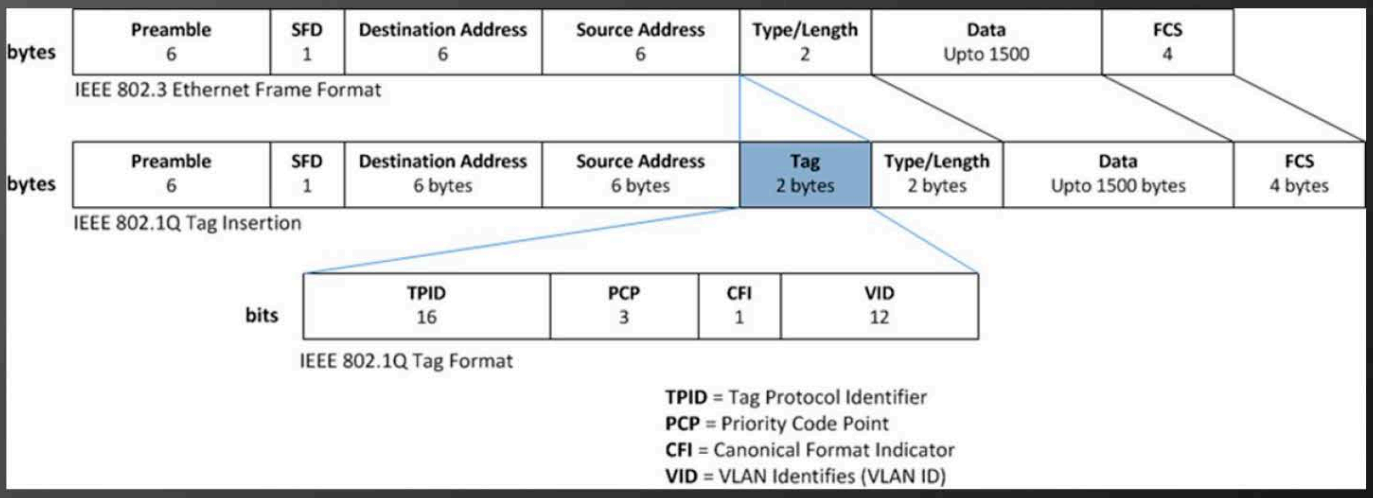
Методи класифікації трафіку комп'ютерної мережі

2



Вміст кадру Ethernet 802.1q

3



Типи трафіку у стандарті 802.1p

4

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

5

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				

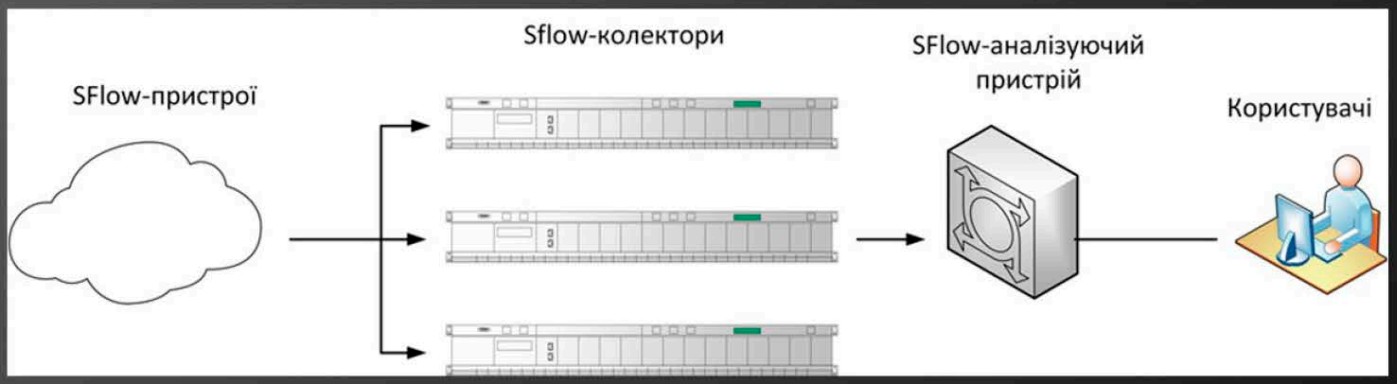
Класифікація бітів пріоритету

6

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

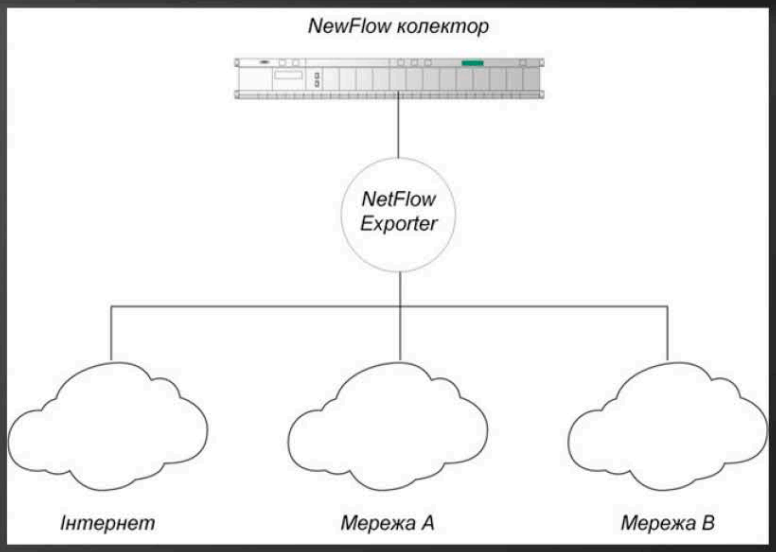
Організація роботи технології SFlow у мережі

7



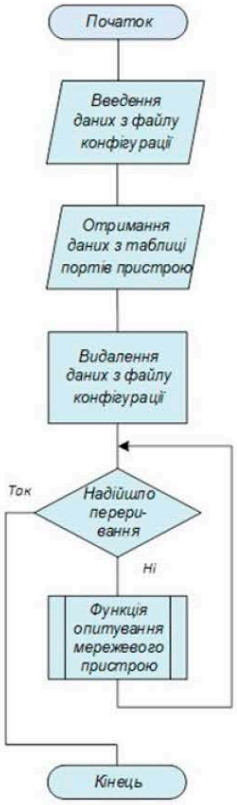
Організація роботи технології NetFlow у мережі

8



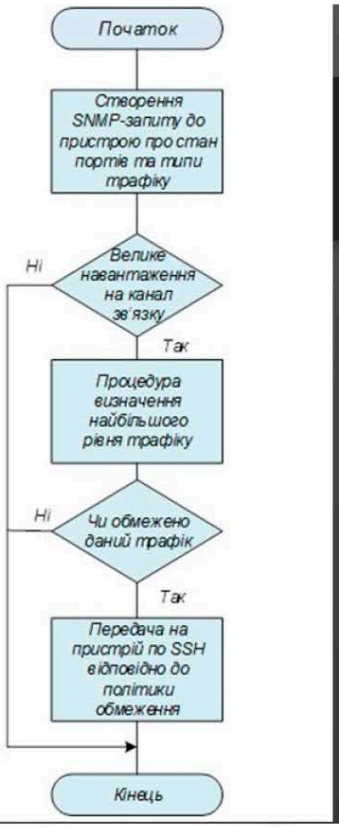
Блок-схема алгоритму оптимізації мережевого трафіку

9



Блок-схема алгоритму читання таблиці портів маршрутизатору Cisco

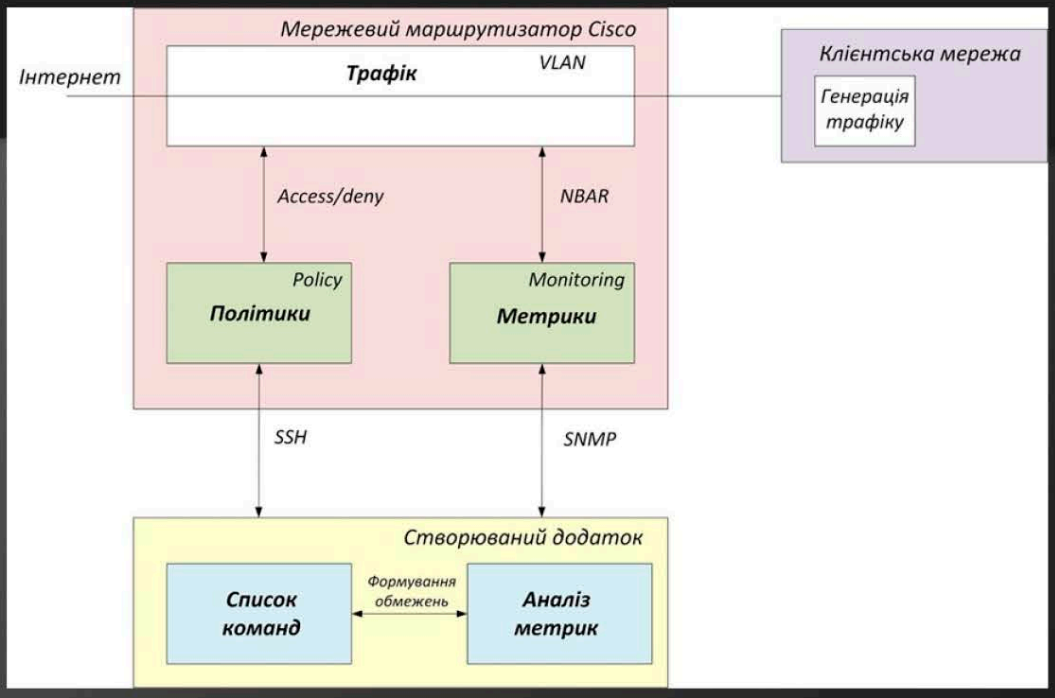
Блок-схема алгоритму моніторингу навантаження каналу зв’язку



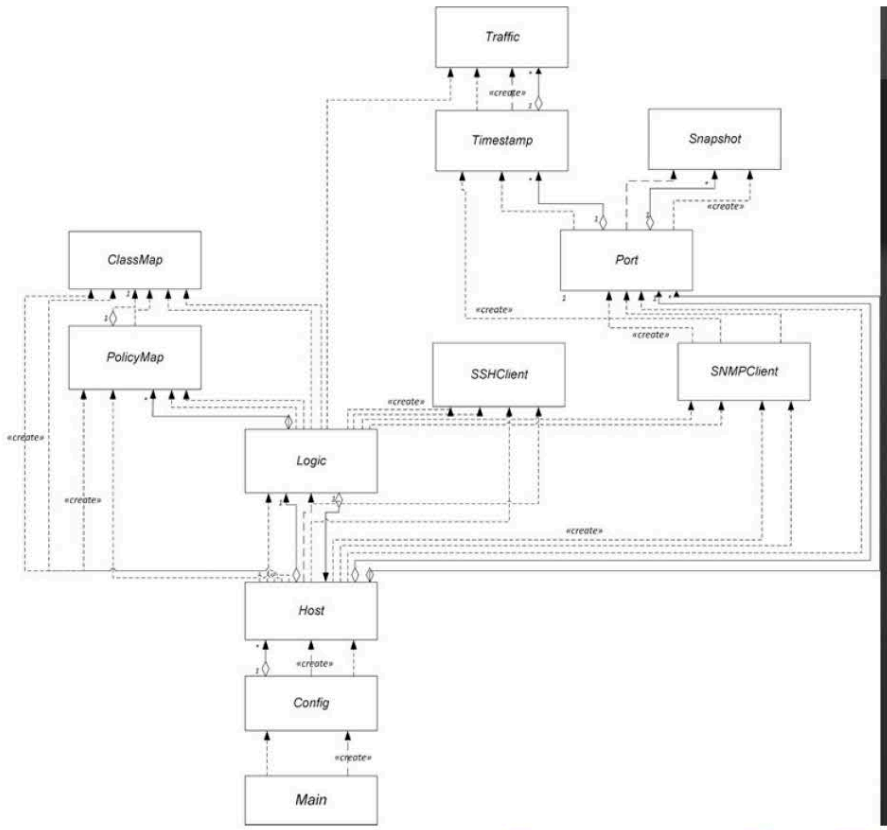
10

Блок-схема алгоритму скидання файлу конфігурації пристрою

Структурна схема моделі розпізнавання потоку даних

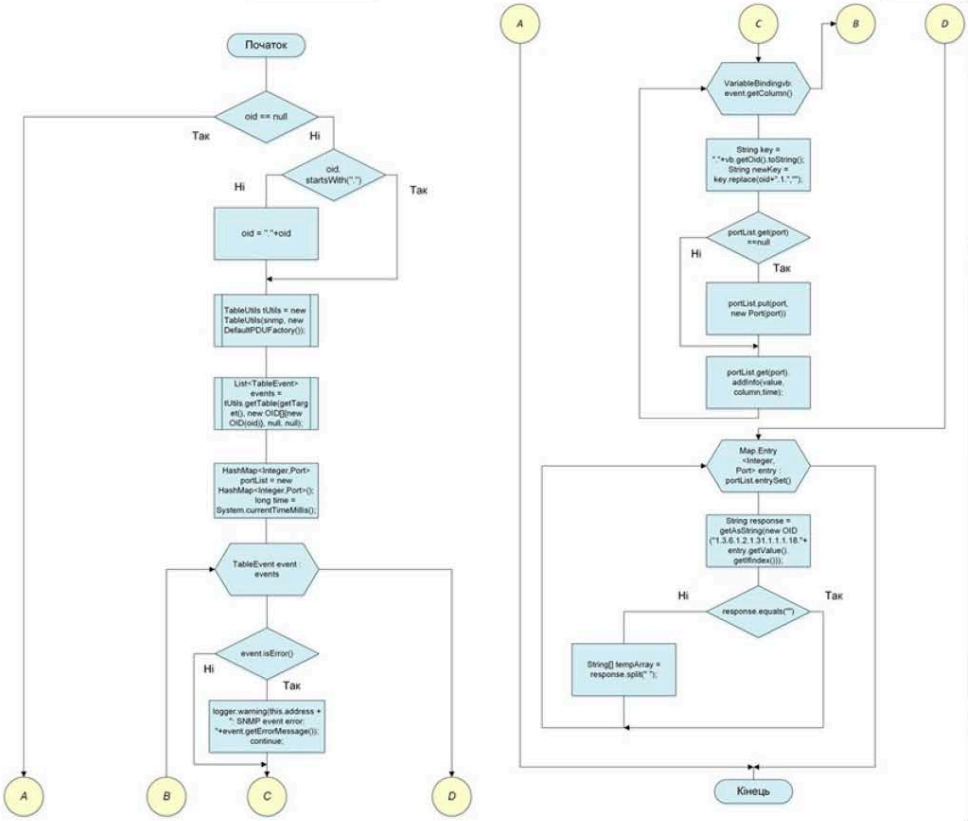


11



12

Функціональна схема діаграми класів у програмі

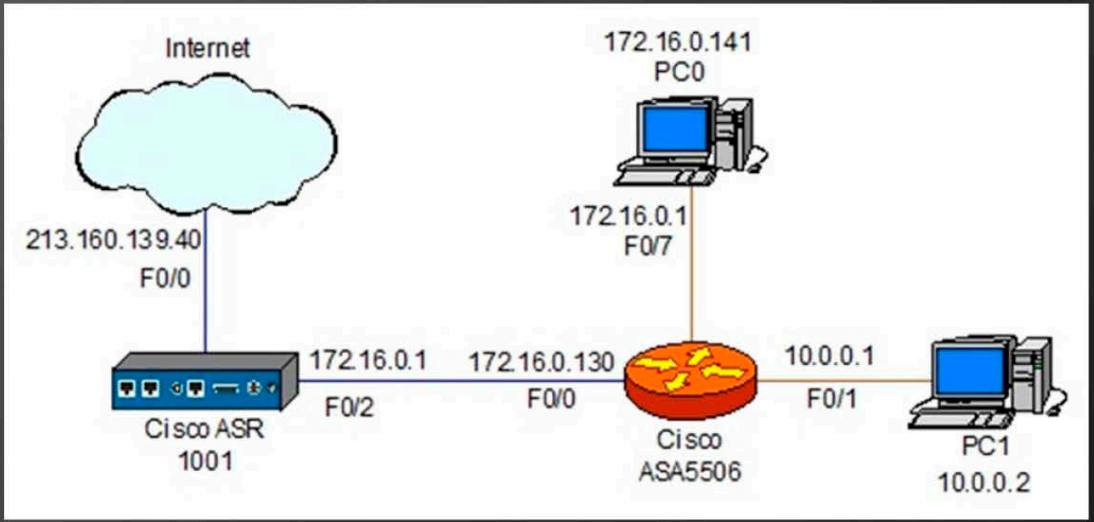


13

Блок-схема алгоритму основного коду програми

Структура мережі для тестування розробленої моделі

14



Лістинг сформованих команд та виведення логу SSH-з'єднання

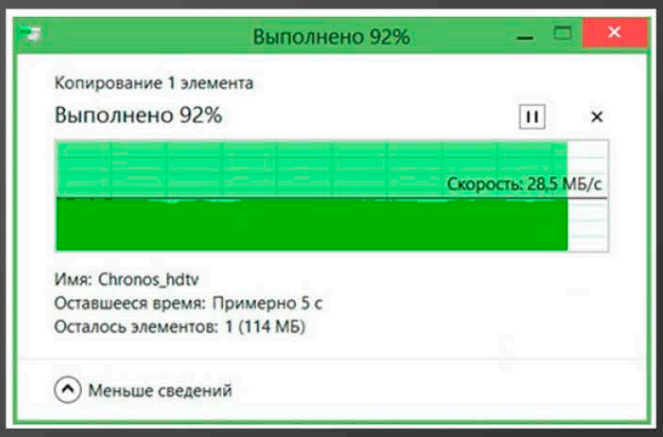
15

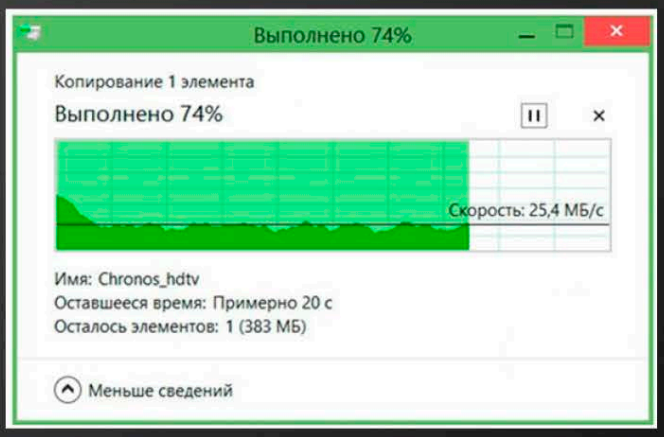
```
Straight commands:
configure terminal
class-map match-all CMAP20190430193451
match protocol http
exit
policy-map PMAP20190430193451
class CMAP20190430193451
police 10000000 conform-action transmit exceed-action drop
exit
exit
interface FastEthernet0/1
service-policy output PMAP20190430193451
exit
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

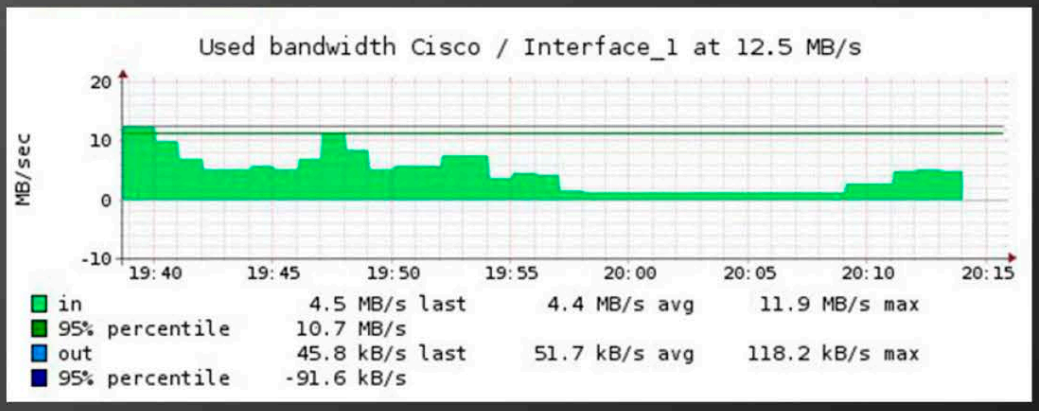
16





Визначення навантаження на канал зв’язку в мережі

17



ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

РЕЦЕНЗІЯ

на дипломний проект (роботу) здобувача (здобувачки) освіти

відділення комп'ютерних систем

Бухтєєва Максима Ігоровича

(прізвище, ім`я та по батькові)

Спеціальність 123 "Комп'ютерна інженерія"

Освітня програма «Комп'ютерна графіка і Web-дизайн»

Керівник дипломного проекту (роботи) Кривченко Анастасія Анатоліївна

(прізвище, ім`я та по батькові)

Тема дипломного проекту (роботи) Розробка моделі розпізнавання потоку даних мережі для маршрутизаторів Cisco

Обсяг розрахунково-пояснювальної записки 68 сторінок

Обсяг графічної (презентаційної) частини 17 аркушів (слайдів)

ХАРАКТЕРИСТИКА ДИПЛОМНОГО ПРОЕКТУ (РОБОТИ)

а) заключення про ступінь відповідності виконаного дипломного проекту (роботи) завданню

Представлений на рецензію дипломний проект повністю відповідає меті проектування та технічному завданню. Тематика дипломного проекту є актуальною та присвячена розробки моделі розпізнавання потоку даних мережі для маршрутизаторів Cisco, алгоритмічного та програмного забезпечення.

б) характеристика виконання кожного розділу дипломного проекту (роботи)

Дипломний проект складається зі вступу, трьох розділів, висновків, переліку використаних джерел. У технологічному розділі виконано огляд і аналіз моделей розпізнавання мережевих потоків даних, алгоритмів розпізнавання потоку даних мережі, розробка програмного забезпечення для розпізнавання потоку даних мережі, тестування моделі розпізнавання потоку даних.

в) оцінка якості виконання пояснювальної записки та графічної частини дипломного проекту (роботи) Графічна частина виконана на достатньо високому рівні у вигляді презентації із використанням офісного пакету Microsoft PowerPoint та Visio. Пояснювальна записка виконана акуратно та у відповідності до норм оформлення документів із використанням офісного пакету Microsoft Word. Загальна якість виконання документації – добра, академічного плагіату у роботі не виявлено

г) перелік позитивних якостей дипломного проекту (роботи)

Програма дозволяє обмежувати типи трафіку, які дають навантаження на канал зв'язку та не входять в список описаних в конфігураційному файлі типів трафіку.

д) основні недоліки дипломного проекту (роботи)

У розділі охорони праці наведені відомі нормативні вимоги загального плану замість конкретних розрахунків освітлення приміщення, вентиляції, рівня шуму. Забагато алгоритмів наведено у ПЗ.

Оцінка розрахункової частини відмінно

Оцінка графічної частини відмінно

Загальна оцінка відмінно

Прізвище, ім'я, по батькові рецензента Сідень Сергій Віталійович,

Місце роботи і посада рецензента "Державний університет інтелектуальних технологій і зв'язку", к.т.н, в.о. зав. каф. радіоелектронних систем і технологій

Підпис:

« 16 » червня 2023 р.

ПІДПИС ПОСВІДЧУЮ
НАЧАЛЬНИК ВІДДІЛУ
КАДРІВ ДУІТЗ

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ * ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНТЕЛЕКТУАЛЬНИХ ТЕХНОЛОГІЙ І ЗВ'ЯЗКУ * 43997335 *
ВІДДІЛ КАДРІВ

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

ВІДГУК

керівника на дипломний проект здобувача (здобувачки) освіти

відділення комп'ютерних систем

Бухтєєва Максима Ігоровича

(прізвище, ім`я та по батькові)

Спеціальність: 123 "Комп'ютерна інженерія"

Освітня програма: «Комп'ютерна графіка і Web-дизайн»

Тема дипломного проекту: Розробка моделі розпізнавання потоку даних мережі для маршрутизаторів Cisco

ХАРАКТЕРИСТИКА ДИПЛОМНОГО ПРОЕКТУ

а) обсяг і якість виконання проекту (графічного матеріалу і розрахунково-пояснювальної записки) Дипломний проект виконано відповідно технічному завданню. Пояснювальна записка містить 68 сторінок. У пояснювальній записці наведено етапи розробки моделі розпізнавання потоку даних мережі для маршрутизаторів Cisco, алгоритмічного та програмного забезпечення. Графічна частина складається з 17 слайдів мультимедійної презентації, передбачених 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 Бухтєєв М.І. 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 Бухтєєв М.І. під час роботи над дипломним проектом вивчив достатню кількість літературних джерел та матеріалів за даною тематикою. Вважаю, що теоретична підготовка дипломника достатня і він готовий до захисту дипломного проекту

г) вміння розв'язувати виробничі та конструкторські питання

Під час дипломного проектування здобувач освіти Бухтєєв М.І мав змогу самостійно приймати окремі рішення з реалізації програмного забезпечення та показав вміння організовано працювати над поставленим завданням, використовуючи сучасні програмні засоби розробки, зокрема Intellij IDEA

Оцінка розрахункової частини Добре

Оцінка графічної частини Відмінно

Загальна оцінка Добре

Прізвище, ім'я, по батькові керівника дипломного проекту

Кривченко Анастасія Анатоліївна

Місце роботи і посада керівника дипломного проекту

ВСП "Одеський технічний фаховий коледж ОНТУ", викладач спецдисциплін комісії комп'ютерних технологій та програмної інженерії, голова обл. метод. комісії викладачів комп'ютерної інженерії

Підпис

«11» 06 2023 р.

ДОЗВІЛ
НА РОЗМІЩЕННЯ
ВИПУСКНОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ
В ЕЛЕКТРОННОМУ РЕПОЗИТАРІЇ ВСП «ОТФК ОНТУ»

Ми, що нижче підписалися,

Бухтєєв Максим Ігорович,
здобувач освіти гр. 4КГ-06, та

Кривченко Анастасія Анатоліївна,
керівник дипломного проекту,

не заперечуємо щодо розміщення електронного варіанту пояснювальної записки до випускної кваліфікаційної роботи молодшого спеціаліста на тему:

«Розробка моделі розпізнавання потоку даних мережі для маршрутизаторів Cisco» (автор роботи – Бухтєєв М.І., керівник роботи – Кривченко А.А.)

виконаного у ВСП «Одеський технічний фаховий коледж Одеського національного технологічного університету» в 2023 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 _____ / Бухтєєв М.І. /

Керівник _____ / Кривченко А.А. /

« 12 » червня 2023 р.

UNICHECK

Ім'я користувача:
Наталія Вікторівна Копусь

ID перевірки:
1015208336

Дата перевірки:
23.05.2023 15:40:26 EEST

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
23.05.2023 15:46:39 EEST

ID користувача:
100011688

Назва документа: Бухтєєв М.І. 4КГ-06

Кількість сторінок: 56 Кількість слів: 10248 Кількість символів: 78572 Розмір файлу: 3.24 MB ID файлу: 1014886430

11.4%
Схожість

Найбільша схожість: 1.99% з Інтернет-джерелом (https://docplayer.net/66622628-Osnovi-ohoroni-praci.html)

11.4% Джерела з Інтернету | 1000 ..... Сторінка 58

Не знайдено джерел з Бібліотеки

0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

0%
Вилучень

Немає вилучених джерел

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Замінені символи | 190