Міністерство освіти і науки України
Київський національний університет імені Тараса Шевченка

Факультет інформаційних технологій
Кафедра мережевих та інтернет технологій

ЗАТВЕРДЖУЮ
завідувач кафедри
мережевих та інтернет технологій
_____ Ю.В. Кравченко
«_____»_____2021 року

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

галузі знань 17 «Електроніка та телекомунікації»
за спеціальністю 172 «Телекомунікації та радіотехніка»

на тему:

РОЗРОБКА РЕКОМЕНДАЦІЙ ЩОДО КОМПЛЕКСНОГО УПРАВЛІННЯ МЕРЕЖНИМИ РЕСУРСАМИ В СУЧАСНИХ МЕРЕЖАХ

Виконав: студент групи МІТ-41

Нікольчев Кирил Степанович _____
(прізвище ім'я по-батькові) (підпис)

Керівник: асистент кафедри мережевих та інтернет технологій

к.т.н., асистент Герасименко К.В. _____
(посада, прізвище ім'я по-батькові) (підпис)

Київ 2021

Міністерство освіти і науки України
Київський національний університет імені Тараса Шевченка

Факультет інформаційних технологій
Кафедра мережевих та інтернет технологій

ЗАТВЕРДЖУЮ
завідувач кафедри
мережевих та інтернет технологій
_____ Ю.В. Кравченко

«_____»_____2020 року

ЗАВДАННЯ
НА ДИПЛОМНУ РОБОТУ

Здобувачу вищої освіти Нікольчеву Кирилу Степановичу
(прізвище, ім'я, по батькові)

1. Тема роботи:

Розробка рекомендацій щодо комплексного управління мережними ресурсами в сучасних мережах

затверджена на засіданні кафедри МІТ «__04__» ___грудня_____ 2020 р. протокол № 8

2. Термін здачі закінченої роботи «30» травня 2021 р

3. Вихідні дані до проекту (роботи)

Мультисервісна мережа, різні види трафіку, засоби управління мережними ресурсами.

4. Зміст пояснювальної записки (перелік питань, що їх потрібно розробити, обсяг – 35-40 стор.)

1. Аналіз стану сучасних мультисервісних мереж.
2. Огляд методів управління мережними ресурсами в сучасних мультисервісних мережах.
   2.1. Класифікація методів управління мережними ресурсами
3. Розробка рекомендацій щодо комплексного управління мережними ресурсами в сучасних мережах.
   3.1 Експериментальне дослідження різних методів управління мережними ресурсами в сучасних мультисервісних мережах;
   3.2 Рекомендації щодо комплексного управління мережними ресурсами в сучасних мережах.

5. Перелік графічного матеріалу 8-10 слайдів

1. Актуальність роботи
2. Класифікація методів управління трафіком
3. Вимоги до телекомунікаційних мереж
4. Експериментальне дослідження
5. Розробка рекомендацій щодо впровадження комплексу інструментів управління мережними ресурсами
6. Висновки

Дата видачі завдання

Керівник роботи к.т.н., асистент Герасименко К. В.

(підпис) (посада, прізвище, ім'я, по батькові)

Завдання прийняв до виконання Нікольчев Кирил Степанович

(підпис) (прізвище, ім'я, по батькові)

КАЛЕНДАРНИЙ ПЛАН ВИКОНАННЯ РОБОТИ

Номер	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Підготовчий	28.01.2021	
2	Розділ 1	01.03.2021	
3	Розділ 2	01.04.2021	
4	Розділ 3	01.05.2021	
5	Доповідь та слайди	25.05.2021	
6	Пояснювальна записка	30.05.2021	

Здобувач вищої освіти _____ Нікольчев Кирил Степанович

(підпис) (прізвище, ім'я, по батькові)

Керівник _____ к.т.н., асистент Герасименко К. В.

(підпис) (прізвище, ім'я, по батькові)

РЕФЕРАТ

Пояснювальна записка: 50 с., 23 рис., 4 табл., 21 джерело.

Об’єкт дослідження: інформаційні (мультисервісні) мережі зв’язку.

Мета роботи (проекту): аналіз засобів управління трафіком в сучасних мультисервісних мережах, що сприятиме визначенню ефективних підходів до управління мережними ресурсами. Розробка підходу до управління мережними ресурсами в сучасних мультисервісних мережах.

Методи дослідження: метод аналізу, імітаційний метод та метод натурного експерименту.

В роботі проведено аналіз методів управління мережними ресурсами в сучасних мультисервісних мережах. Приведена класифікація методів управління мережними ресурсами, основними з яких є методи вибору оптимального шляху проходження трафіку, алгоритми управління інтенсивністю, механізми управління чергами та запобігання перевантажень.

Розроблено рекомендації щодо комплексного управління мережними ресурсами в сучасних мережах.

Практичне значення роботи полягає у розробці комплексу рекомендацій щодо управління мережними ресурсами в мережі підприємства з різнорідним трафіком, використовуючи існуючі механізми управління трафіком.

Результати здійснених у дипломному проекті досліджень можуть бути використані в управлінні мережними ресурсами в сучасних мультисервісних мережах.

Наукова новизна дослідження полягає в оновленні існуючої теоретичної бази з точки зору засобів управління мережними ресурсами.

Галузь використання – сучасні інформаційні (мультисервісні) мережі України.

У подальшому планується розробити універсальний метод управління мережними ресурсами в умовах мультисервісності сучасних мереж.

Ключові слова: МУЛЬТИСЕРВІСНА МЕРЕЖА, ЯКІСТЬ ОБСЛУГОВУВАННЯ, УПРАВЛІННЯ ТРАФІКОМ, МЕРЕЖНІ РЕСУРСИ, НЕОДНОРІДНІСТЬ ТРАФІКУ, CLASS-BASED FAIR QUEUING, WEIGHTED RANDOM EARLY DETECTION, TRAFFIC ENGINEERING

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕЬ, СИМВОЛІВ,ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ЛОМ – локальна обчислювальна мережа
МСМ – мультисервісна мережа
МПД – мережа передачі даних
ТМ – таблиця маршрутизації
ЦУ – центр управління
AF – Assured forwarding, гарантована доставка
ARED – Adaptive, адаптивне завчасне виявлення
BE – Best-Effort, краще зусилля
CIR – Committed Information Rate, узгодження інтенсивності передавання пакетів
CS – Class selector, селектор класу
CQ – Custom Queuing, настроювані черги
CBWFQ – Class-Based Weighted Fair Queuing, зважене справедливо обслуговування на основі класів
DSCP – Differentiated Service Code Point, код диференційованої послуги
CWND – congestion window, «TCP-вікно» або «вікно перевантаження»
EF – Expedited forwarding, негайна доставка
FIFO – First In - First Out, алгоритм обробки «перший прийшов – перший
FWFQ – Flow-based Weighted Fair Queuing, зважене справедливо обслуговування на основі потоків
IETF – Internet Engineering Task Force, робоча група проектувальників Інтернет
IntServ – Integrated Service, інтегроване обслуговування
Leaky bucket – алгоритм «дірявого відра»
LLQ – Low Latency Queuing буферизація з низькою затримкою
MPLS – Multiprotocol Label Switching — багатопротокольна комутація за мітками
PQ – Priority Queuing, пріоритетне обслуговування
PHB – Per-Hop Behavior, політика покрокового обслуговування пакетів
QoS – Quality of Service, якість обслуговування
RIO – RED In & Out, випадкове завчасне виявлення з використанням
RED – Random Early Detection, випадкове завчасне виявлення
RSVP – Resource Reservation Protocol, протокол резервування мережних ресурсів
SLA – Service-level agreement, угода про рівень обслуговування
TS – Traffic policing, функція обмеженя трафіка
TP – Traffic shaping, функція вирівнювання трафіка
Token bucket – алгоритм «кошика маркерів»
ToS – Type of Service, байт типу обслуговування заголовка IPv4
WRED – Weighted RED, зважене випадкове завчасне виявлення промаркованих і непромаркованих пакетів
WFQ – Weighted Fair Queuing, зважене справедливе обслуговування
WRR – Weighted Round Robin, зважене кругове обслуговування

ВСТУП

Враховуючи рівень сучасного розвитку інформаційних технологій, можна зрозуміти, що все більшого використання набувають мультисервісні послуги та конвергенція телекомунікаційних мереж. Ця тенденція вимагає відповідних мережних ресурсів, оскільки у відповідності до вимог часу, сучасні мережі передавання даних повинні проектуватися як високонадійні системи, що здатні забезпечити задані показники QoS. Задля забезпечення різних вимог параметрів QoS мультисервісних послуг у системах передачі даних необхідним кроком є впровадження алгоритмів та механізмів управління трафіком, для врахування особливостей різних видів послуг, а також забезпечення ефективного використання мережних ресурсів.

На сьогоднішній день вже існує досить багато механізмів обробки мережного трафіку, які сприяють забезпеченню вимог до якості обслуговування. Основним та найбільш важливим параметром та механізмом є буферний ресурс, а також механізми управління доступом та перевантаженнями. Використання цих параметрів та механізмів дозволяє суттєво зменшити такі показники як середня затримка, джитер і рівень втрат пакетів без нарощування величини доступних мережних ресурсів, насамперед, пропускної здатності каналів зв'язку. В свою чергу, ефективність вирішення задач управління чергами залежить від адекватності моделей і методів, що закладені в їх основу. Тому одним з найбільш важливих етапів розробки моделей і методів управління трафіком в телекомунікаційній мережі є їх перевірка в ході експериментального (лабораторного) дослідження.

Методи управління трафіком відрізняються залежно від завдань, які постають перед мережним адміністратором. Деякі методи спрямовані на розв'язання однієї задачі, але зовсім не здатні допомогти у інших випадках. Існують доволі універсальні, та вони не розв'язують ту чи іншу проблему повною мірою, а можуть лише частково покращити ситуацію. Однак при поєднанні з іншими здатні суттєво покращити параметри зв'язку.

Актуальність дипломної роботи. Насьогодні користувачі мультисервісних мереж все більше зацікавлені у використанні таких служб як відеоконференцзв'язок, доступ до web-служб, які доволі чутливі до затримок та джитеру. Аналіз літератури виявив, що на сучасному етапі відсутній опрацьований комплексний підхід до управління мережними ресурсами, до яких відноситься канальні ресурси, буферний простір мережних пристроїв і трафік. Для вирішення даної проблеми є необхідним впровадження такого комплексного підходу з урахуванням класів обслуговування для різних видів трафіку, та системи управління трафіком, що буде забезпечувати задані класи обслуговування для різних видів трафіку, перерозподіляючи мережевий ресурс.

Дипломна робота, яка присвячена розробці комплексного управління трафіком в сучасних мультисервісних мережах є актуальною, доцільною, тому що до цього часу немає комплексного підходу до вирішення цієї проблеми. Для забезпечення ефективної передачі трафіку з підтримкою параметрів QoS можливо використовувати технології ТЕ (Traffic Engineering) за рахунок використання процедур резервування, вибору оптимального маршруту

проходження трафіку, механізмів збалансованого завантаження ресурсів мережі і запобігання перевантажень.

Метою дипломної роботи є аналіз засобів управління трафіком в сучасних мультисервісних мережах, що сприятиме визначенню ефективних підходів до управління трафіком. Розробка підходу до управління трафіком в сучасних мультисервісних мережах. Для досягнення поставленої мети в роботі були розв'язані наступні задачі дослідження:

1) аналіз основних методів управління трафіком та їх використання в існуючих сервісних моделях якості обслуговування QoS;
2) формулювання основних вимог, які повинні відповідати перспективним рішенням в галузі забезпечення гарантованої якості обслуговування;
3) розробка комплексного підходу до управління трафіком шляхом проведення натурного експерименту.

Об'єктом дослідження є інформаційні (мультисервісні) мережі зв'язку, а предметом досліджень – підхід до управління трафіком в сучасних мультисервісних мережах.

Завдання дипломної роботи розв'язувались із застосуванням імітаційного методу дослідження , а також методу натурного експерименту.

1 АНАЛІЗ СТАНУ СУЧАСНИХ МУЛЬТИСЕРВІСНИХ МЕРЕЖ

Різкий стрибок у розвитку телекомунікаційних та інформаційних технологій призвів до того, що з'явилися мережі, з виділяючою особливістю - неоднорідним трафіком [1, 2, 3]. При неоднорідному трафіку по телекомунікаційній мережі передаються пакети різних типів (мовних, відео, аудіо, та текстових пакетів і т. д.), до яких пред'явлені різні вимоги [4]. Такі мережі дістали назву мультисервісних мереж (МСМ), тобто мереж, які здатні надати різнорідні інформаційні та телекомунікаційні послуги.

МСМ - це універсальне багатоцільове середовище, яке призначене для передачі різнорідного трафіка з використанням комутації пакетів.

МСМ відрізняється ступенем надійності, що характерна для телефонних мереж (на відміну від негарантованої якості обслуговування через мережу Інтернет) і забезпечує невисоку вартість передачі в розрахунку на одиницю об'єму інформації (близьку до вартості передачі даних через Інтернет).

Основна задача МСМ – забезпечення роботи різнорідних телекомунікаційних та інформаційних систем і прикладних програм у єдиному транспортному середовищі, в той час коли для передачі звичайного трафіка (даних) і трафіка іншої інформації (мови, відео та ін.) використовується єдина інфраструктура. МСМ використовує один спільний канал для передачі даних різного виду, що дозволяє зменшити різнотипність обладнання, використовувати єдині технології, стандарти і керувати комунікаційним середовищем централізовано.

Перехід до нових мультисервісних технологій змінює концепцію, за якою надаються послуги, коли якість гарантована не тільки на рівні вимог дотримання стандартів і угод з постачальником послуг, а і на рівні операторських мереж і технологій. Задля можливості виконання приведених угод є необхідним відповідність поточних параметрів якості обслуговування QoS (Quality of Service) мережі встановленим параметрам, як, наприклад, смуга пропускання, затримка, варіація затримки та рівень втрати пакетів [5, 6]. Задля одночасного забезпечення різних вимог QoS в МСМ необхідно використовувати засоби управління трафіком, що повинні враховувати особливості різних класів трафіка і забезпечувати ефективне розподілення ресурсів мережі.

Концепція мультисервісності суміщає декілька аспектів стосовно різних сторін побудови мережі.

Перш за все, конвергенція завантаження мережі, яка встановлює передачу різнотипного трафіку в рамках одного формату представлення даних. Сьогодні передача голосових і відеопотоків проходить в основному через мережі, які більше орієнтовані на комутацію каналів, а передача даних - використовуючи мережу з комутованими пакетами. Конвергенція завантаженості мережі встановлює тенденцію експлуатації мереж з комутованими пакетами для передачі та голосового і відеотрафіку, і звичайних даних мереж. Але це не є запереченням основних вимог диференціювання трафіку до надання якості послуг.

По-друге, перехід від великої кількості вже використовуваних мережних протоколів до єдиного (зазвичай, IP). Тоді коли існуючі мережі створені для управління різними протоколами, такими як IP, IPX, AppleTalk, і спільного типу

даних, МСМ орієнтовані на єдиний протокол і різнорідні сервіси, які використовуються для підтримки різного типу трафіку.

Фізична конвергенція, з передачею різнотипного трафіку через єдину мережеву інфраструктуру. Як мультимедійний, так і голосовий потоки можуть бути передані використовуючи одне і те ж саме обладнання враховуючи різні вимоги до пропускної здатності, затримок і «джитеру» частоти. Резервуючі протоколи ресурсу, формування черг пріоритетності і якості обслуговування (QoS) надають можливість диференціювати послуги, які використовуються для різних типів трафіку.

Конвергенція пристроїв, визначаюча архітектурну тенденцію побудови мережних пристроїв, що підтримує трафік різного типу в рамках однієї системи. Наприклад, комутатор здатний підтримувати комутацію пакетів Ethernet, IP-маршрутизацію, а також з'єднання ATM. Мережеві пристрої мають можливість обробляти дані відповідно до загального протоколу мережі та мають різні сервісні вимоги, такі як затримка, гарантії ширини смуги пропускання та ін. Окрім цього, пристрої надають можливість працювати як Web-додаткам, так і пакетній телефонії.

Конвергенція додатків, яка визначає інтеграцію різних функцій в одному програмному засобі. Так, Web-браузер може об'єднати в рамках однієї сторінки різні дані типу звукового, відеосигналу, графіки високої роздільної здатності та ін.

Конвергенція технологій прагне створити єдину загальну технологічну базу для побудови мереж зв'язку, здатну задовольнити вимоги як регіональних мереж зв'язку, так і локальних обчислювальних мереж. Існує, наприклад, асинхронна система передачі (ATM) може бути використана для побудови як регіональних, так і локальних обчислювальних мереж.

Усі перераховані аспекти показують різні сторони проблеми роботи МСМ, що можуть передавати різнорідний трафік в периферійній частині мережі та її ядрі.

1.1 Вимоги до мультисервісних мереж

МСМ надають можливість операторам збільшити свої мережеві магістралі в сторону додавання новіших сервісів, при цьому пропонуючи додаткові послуги для великої кількості корпоративних клієнтів. Під МСМ розуміється надання телекомунікаційних послуг різного виду по єдиній інфраструктурі передачі даних.

Коли говориться про реалізацію МСМ, зазвичай розглядаються чотири технічних питання: пропускна здатність, розсинхронізація, управління, затримка.

Сьогоднішній зростаючий попит на новітні широкосмугові передачі даних, потреба в доступі до глобальної мережі Інтернет при жорсткій конкуренції змушує провайдерів збільшувати різноманітність послуг, зменшувати витрати на інфраструктуру та інше. Отже, виникає необхідність у платформі, яка буде спроможна запропонувати комплексне рішення, що надає можливість використовувати великий спектр послуг: ATM, Frame Relay, Internet, IP, передачі відеосигналу і голосового трафіку з гарантованою якістю обслуговування (QoS).

Якщо подивитися на проектування мережі, то МСМ потребують зовсім іншого підходу. Відео і голосовий трафіки повинні реалізовуватися в реальному часі - з необхідністю пріоритезувати у випадку перевантажень транспортної мережі. Але мережева індустрія ніколи не намагалася орієнтуватися на мережі реального часу, зазвичай дані завжди доставлялися відповідно до існуючих можливостей мережі в конкретно визначений проміжок часу.

Сьогодні існує багато різних способів побудови мереж передачі даних, голосу і відео. Одним таким способом є поділ фізичної лінії на різні види зв'язку або, по-іншому, - мультиплексування. Також можливе надання декількох сервісів в рамках однієї мережі, наприклад, передача даних і голосового трафіку в оцифрованому вигляді через цифрові мережі (наприклад IP-телефонія, що забезпечує передачу голосової інформації через IP-мережі).

Як перший, так і другий способи побудови МСМ знайшли своє відображення в рішеннях трьох найбільших виробників телекомунікаційного обладнання - Cisco Systems, Avaya Communications/Lucent Technologies і Nortel Networks, що розподілили між собою фактично весь ринок обладнання для МСМ по всьому світу.

Кожна з перелічених компаній має величезний досвід у розробці різних систем зв'язку і передачі інформації, а також свій підхід, який, власне, і реалізується при роботі над рішеннями для МСМ.

1.2 Архітектура мультисервісної мережі

Є велика кількість варіацій проектування МСМ. Однією з них є будування гомогенної інфраструктури, що може бути або повністю пакетною, яка не орієнтована на з'єднання мережі (такі як роздільні і комутовані ЛВС, пакетні мережі зв'язку), або орієнтованою на з'єднання мережі (типу АТМ). Через наявні відмінності в економічних і функціональних вимогах для локальних обчислювальних мереж і регіональних мереж зв'язку ні одна з зазначених архітектур окремо практично не може задовольнити користувачів під час побудови МСМ.

Взагалі, периферійні локальні мережі користуються різними технологіями. В той час як одна з мереж може працювати на комутованій Ethernet-технології (без використання пристроїв маршрутизації), тоді як інша - на технології АТМ ЛВС, і третя - на маршрутизованих сегментах Ethernet-мережі.

На основі таких технологій як frame relay, асинхронна система передачі або Internet може бути побудовано ядро мережі.

Мережі, в основі яких є передача пакетів, здебільшого Internet, забезпечують високу якість потокового трафіку, який не є чутливим до затримок, але не можуть бути використані для передачі трафіку з високими вимогами до смуги пропускання, затримки і «джитеру». Асинхронні системи передачі даних, що орієнтовані на з'єднання, навпаки, забезпечують високу якість сервісу для потоків даних з високими вимогами до смуги пропускання, затримки і «джитеру».

В даний час найвисокоякіснішим рішенням для магістралей мережі є технологія АТМ, яка може забезпечити масштабовану пропускну здатність і гарантовану якість послуг QoS. Багатофункціональні комутатори АТМ, що

надають різні інтерфейси для підключення кінцевого обладнання, надають взаємодію через єдину інфраструктуру. За допомогою них великі підприємства можуть об'єднати трафік багатьох мереж в одній магістралі.

Велику увагу сьогодні привертає ще одна нова технологія - телефонія на базі IP (Voice over IP, VOIP). Для комерційних підприємств найбільш значущою перевагою передачі голосу по IP є скорочення витрат: наявна мережа передачі даних має можливість передавати голосовий трафік замість використання платної загальнодоступної телефонної мережі. Багато великих корпорацій вже мають великі мережі на базі IP-телефонії.

2 ОГЛЯД МЕТОДІВ УПРАВЛІННЯ МЕРЕЖНИМИ РЕСУРСАМИ В СУЧАСНИХ МУЛЬТИСЕРВІСНИХ МЕРЕЖАХ

Методи управління трафіком розрізняються в залежності від задач, що постають перед адміністратором мережі. Деякі методи спрямовані на вирішення деякого конкретного завдання, проте зовсім не здатні допомогти в інших випадках по забезпеченню вимог QoS. Інші методи більш універсальні, але все таки в повній мірі не вирішують ту чи іншу проблему, а лише частково покращують ситуацію. Вирішити ці проблеми допомагає поєднання цих методів, тобто комплексне використання засобів управління трафіком, яке може суттєво покращити параметри зв'язку.

2.1 Класифікація методів управління мережними ресурсами

На сьогоднішній день відома велика кількість методів управління трафіком, що включать в себе використання механізмів збалансованого завантаження ресурсів мережі, вибору оптимального маршруту проходження трафіку, використовування механізмів запобігання перевантажень і процедур резервування.

Основні і найбільш використовувані засоби управління трафіком представлені на рис. 2.1.

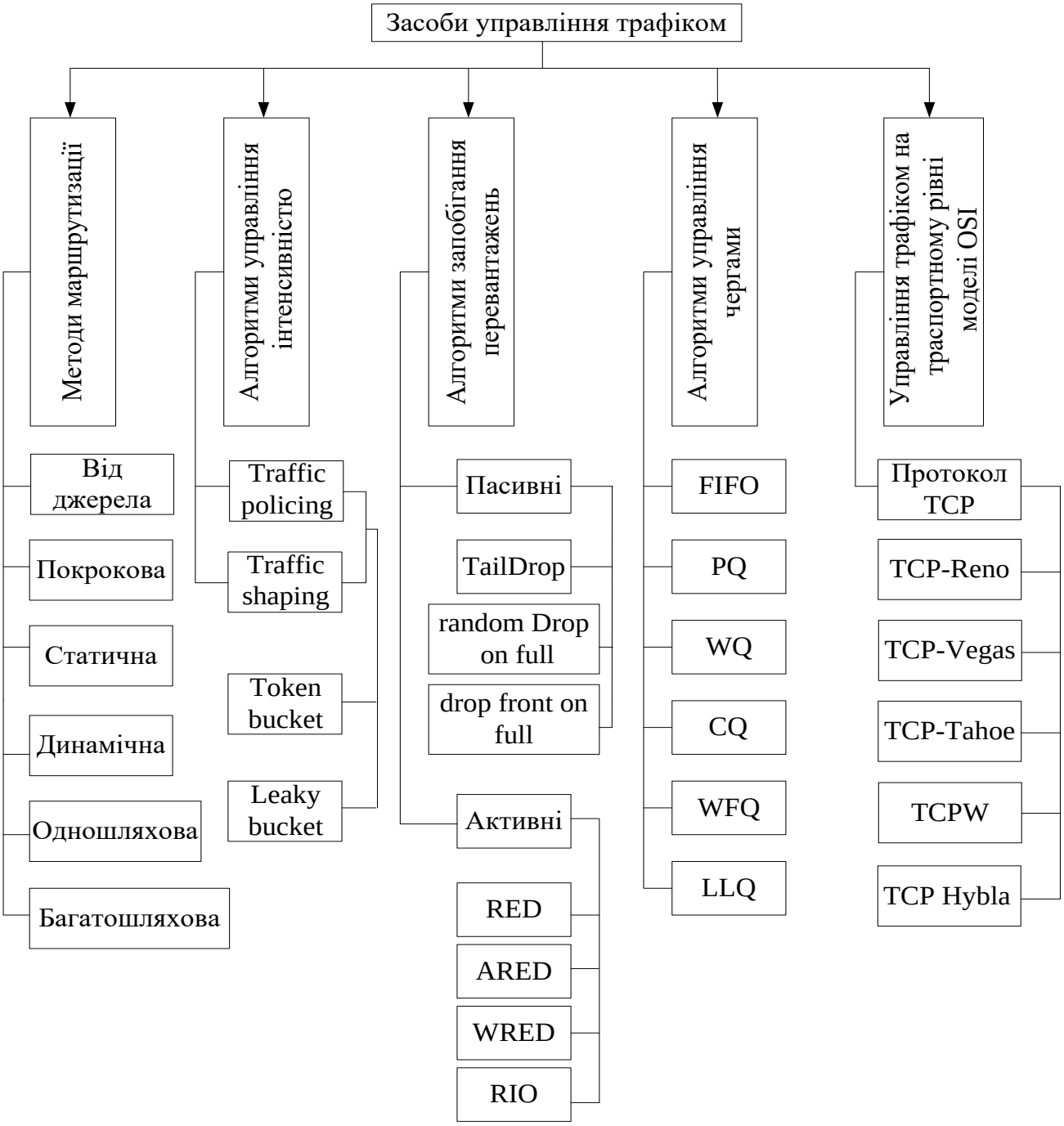

Рисунок 2.1 – Засоби управління трафіком

2.1.1 Алгоритми управління інтенсивністю трафіка

Для забезпечення функцій якості обслуговування увесь трафік, який йде до окремої мережі, повинен піддаватися суворому контролю на межі цієї мережі щодо відповідності його інтенсивності параметрам, які зазначено в угоді про рівень обслуговування (SLA) та які можуть підтримуватися даною мережею. Такий набір параметрів іноді називають профілем потоку, а управління інтенсивністю, яке зумовлює приведення потоку у відповідність до профілю цього потоку – профілюванням трафіка.

Відповідно до наведеної схеми організації управління доступом (рис. 2.2) потік даних, який виділено класифікатором із вхідного потоку трафіка на підставі певної ділянки заголовка пакета, направляється на вхід вимірювача.

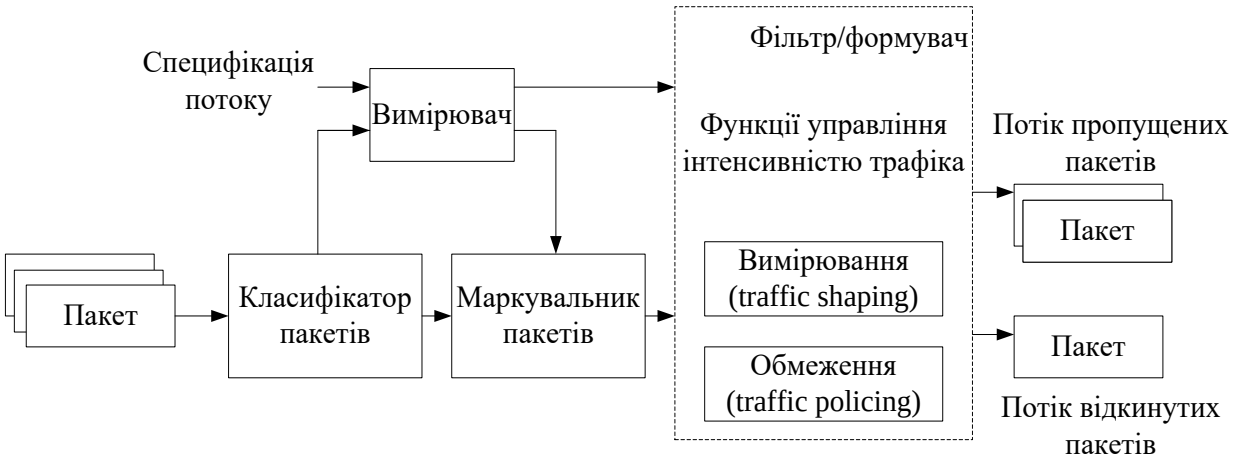

Рисунок 2.2 – Схема управління доступом

Вимірювач буде порівнювати часові характеристики потоку з заявленими в SLA, використовуючи, наприклад, алгоритм «кошика маркерів» [1]. Далі вся ця інформація надається на входам фільтра/формувача та маркувальника. Маркувальник установить значення, такі як поля DSCP, і зарахує його до агрегатора поведінки (Behavior Aggregate, BA) – класу обслуговування. Значення DSCP може залежати від результатів класифікації, а також від стану вимірювача. Деякі пакети, в яких параметри не співпадають з профілем потоку, будуть мати маркування DSCP значення, яке надає низчий пріоритет в обслуговуванні.
Далі промарковані пакети будуть надходити до фільтра/формувача, який буде реалізовувати управління інтенсивністю трафіка. Є можливість досягнути такого управління застосовуючи такі функції:

- обмеження трафіка (traffic policing, TP);
- вирівнювання трафіка (traffic shaping,TS).

Не дивлячись на спільне призначення, обидві функції мають суттєву різницю у способі обробки трафіка в час, коли порушуються параметри профілю (рис.2.3). У табл. 2.1 наведена порівняльна характеристика функції обмеження і функції вирівнювання трафіка.

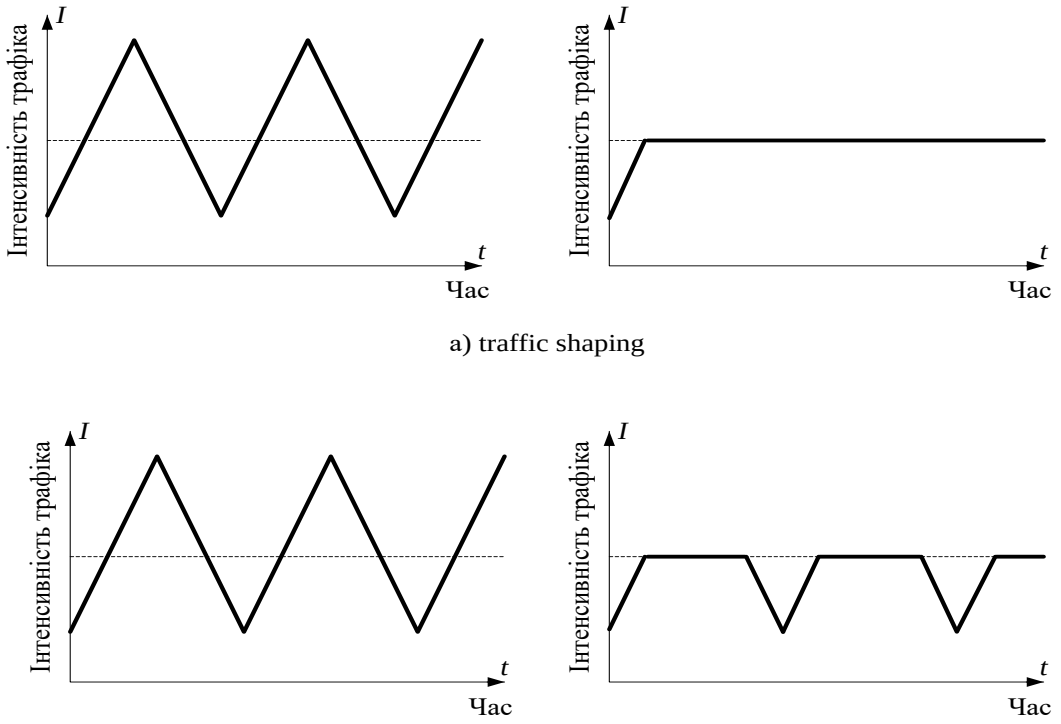

Рисунок 2.3 – Приклад методів управління інтенсивністю трафіка

Профілювання трафіка на основі правил політики (policing) відкидає пакет або, за можливістю, маркує його зі зменшенням пріоритету у тому випадку, коли порушуються параметри профілю (такі як перевищення середньої швидкості або тривалості пульсації). Відкидання частини пакетів значно зменшує інтенсивність потоку і встановлює параметри цього потоку до таких, що були зазначені у профілі. Маркування пакетів без їх відкидання необхідне для того, щоб ці пакети могли бути оброблені цим вузлом (або наступними за потоком), але вже зі зниженою якістю.

Функція вирівнювання трафіка (shaping) використовується для того, щоб надавати трафіку, який зміг пройти профілювання, потрібну «форму» у часі. Ця функція можлива завдяки буферизації пакетів. За допомогою неї найчастіше намагаються згладити стрибки трафіка, і цим зменшити розмір черг на мережевих вузлах, які обробляють трафік далі за потоком. Варто використовувати вирівнювання для того, щоб відновити часові співвідношення трафіка додатків, які користуються рівномірними потоками, такі як мовні додатки.

Таблиця 2.1 – Характеристика порівняння функції обмеження і функції вирівнювання трафіка

	Функція обмеження трафіку (policing)	Функція вирівнювання трафіка (shaping)
Характеристика	Обмеження інтенсивності методом відкидання пакетів при перевищенні заданої швидкості	Вирівнювання інтенсивності за допомогою метода затримки (буферизації пакетів) та наступного пересилання з узгодженою інтенсивністю при перевищенні швидкості, що була задана
Призначення	Обмеження трафіка до швидкості зазначеної в SLA з метою захисту мережі від можливого перевантаження. Обмеження трафіка може допомогти й у випадку запобігання DOS атакам	1. На межі двох мереж на виході першої слід реалізувати вирівнювання, для того, щоб уникнути втрат на вході другої мережі при виконанні функції обмеження трафіка 2. На випадок, якщо в якомусь місці далі в мережі може бути переповнення вхідних черг, а QoS там не є можливим
Маркування пакетів	Підтримується, а також може змінити маркування пакета	Не підтримується
Область застосування	Найчастіше на вході першого мережевого пристрою (маршрутизатора або комутатора). Взігалі може Бути виконана як на вхідних, так і на вихідних портах. Зазвичай на вхідних, бо в такому випадку пакети, які відкидаються, не будуть доходити до процесу маршрутизації, і таким чином заощаджуватимуться ресурси	У будь-якому випадку на вихідному інтерфейсі. Частіше за все реалізується на виході маршрутизатора, який граничить з наступною мережею

Для того, щоб перевірити відповідність вхідного трафіку до заданого профілю як механізм для обмеження трафіка, так і механізм для вирівнювання трафіка повинні мати вимірювання параметрів потоку. Слід зазначити, що основні параметри, які використовуються для вимірювання (дозування) трафіка, є

- T – період усереднення швидкості;

- *CIR* (Committed Information Rate) біт/с – середня (узгоджена) швидкість, що не повинна перевищуватися трафіком;

- B_c, байт – обсяг пульсації, який відповідає середній швидкості *CIR* і періоду *T*, $B_c = CIR \times T$ (узгоджений або стандартний розмір сплеску);

- B_e, байт – допустиме перевищення обсягу пульсації (розширений розмір сплеску).

Для вимірювання трафіка існує два основних алгоритми – «кошика маркерів» (token bucket) і «дірявого відра» (leaky bucket). «Кошика маркерів» найчастіше використовується для того, щоб контролювати параметри трафіка в IP-мережах, а другий – у для мереж ATM і Frame Relay.

2.1.1.1 Алгоритм «кошика маркерів» (token bucket)

Алгоритм «кошика маркерів» надає модливість оцінювати й обмежувати середню швидкість і пульсацію потоку пакетів. Його основою є порівняння потоку пакетів з деяким еталонним потоком маркерів, які наповнюють уявний кошик – «кошик маркерів» (рис. 2.4).

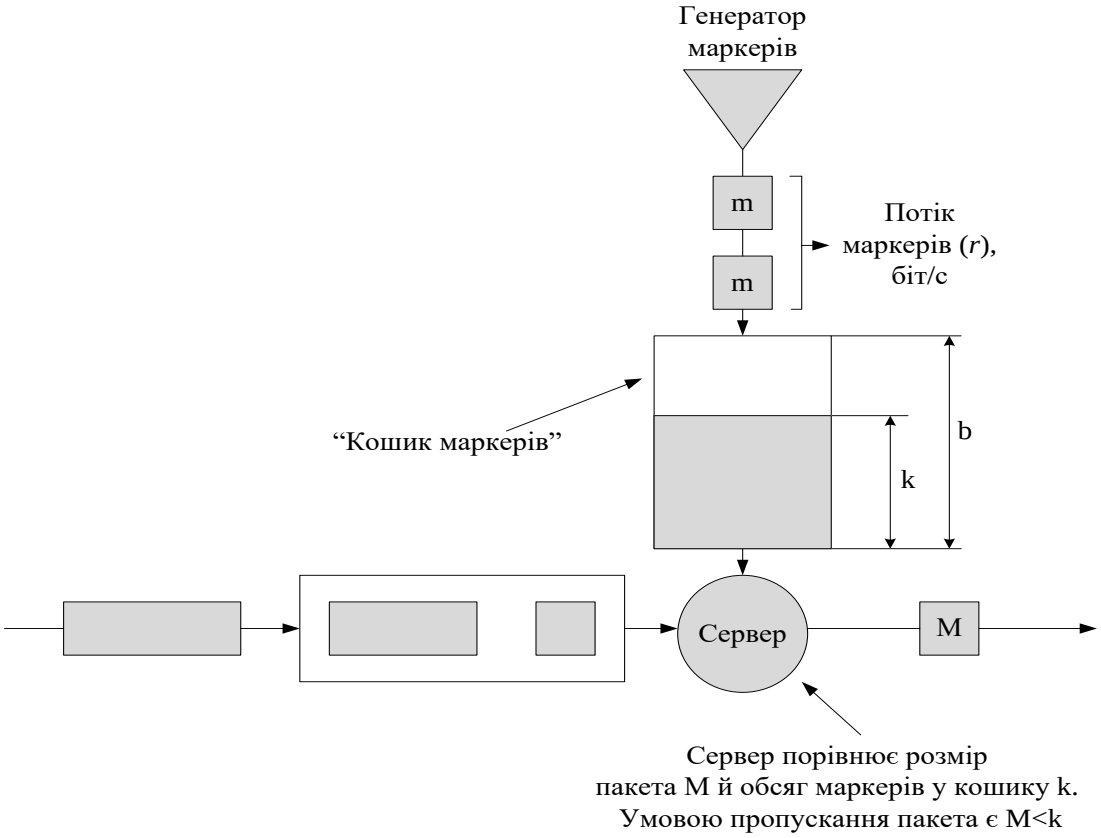

Рисунок 2.4– Алгоритм «Кошика маркерів»

У цьому випадку під маркером мається на увазі деякий абстрактний об'єкт, що носить «порції» інформації. Періодами з постійним інтервалом $t_п$ генератор маркерів направлятиме кожен наступний маркер у «кошик» з визначеним обсягом *b* байт. Усі маркери матимуть один обсяг *m* байт, а маркери генеруються

так, що «кошик» буде заповнюватися зі швидкістю r біт/с, де $r = 8m/t_п$. Така швидкість r і є максимальною середньою швидкістю трафіку пакетів, а обсяг «кошика» дорівнює максимальному розміру пульсації потоку пакетів, тобто при використанні зазначених вище термінів $r = CIR$, $b = B_c$. У випадку, коли загальний обсяг маркерів у «кошику» дорівнює b, тоді надходження маркерів зупиняється на деякий час. Тож фактично «кошик маркерів» є лічильником, до якого додається m кожні $t_п$ секунд.

Коли впроваджується алгоритм «кошика маркерів», профіль трафіка знаходиться за допомогою середньої швидкості r і обсягу пульсації b. Сервер порівнює еталонний і реальний потоки – абстрактний пристрій, якому належать два входи. Перший вхід зв'язаний з чергою пакетів, а другий – з «кошиком маркерів». Також свій вихід має сервер. Він передає пакети на нього з вхідної черги пакетів. Перший вхід на сервері моделює вхідний інтерфейс маршрутизатора, а його вихід – вихідний інтерфейс. Коли відбувається передавання пакета з «кошика» забираються маркери загальним розміром у M байт (з точністю до розміру кожного маркера, тобто до m байт).

У випадку, якщо «кошик» недостатньо заповнений, то пакет буде оброблятися якимсь із описаних нижче нестандартних способів. Його вибір буде залежати від мети використання алгоритму. У першому варіанті, алгоритм «кошика маркерів» впроваджується для згладжування трафіка. Таким чином пакет просто буде затриманий у черзі на додатковий проміжок часу, поки він очікує, щоб в «кошик» надійшла достатня кількість маркерів. З цього видно, що пакети виходять з черги більш рівномірно, у такому темпі, який був заданий генератором маркерів навіть тоді, коли в результаті пульсації до системи надходять великі групи пакетів. При другому варіанті, алгоритм «кошика маркерів» застосовується для обмеження трафіка. У цьому випадку пакет буде відкинутий як не відповідний профілю. Не таким жорстким рішенням є маркування пакета повторно зі зниженням статусу цього пакета в подальшому обслуговуванні. Так є можливість позначити пакет особливою позначкою (відкинути при необхідності). В такому випадку маршрутизатори будуть відкидати такий пакет у першу чергу під час перевантажень. Під час диференційованого обслуговування пакет може бути переведений до іншого класу, що буде обслуговуватися з нижчою якістю.

Сутність такого алгоритму є в тому, що він дозволяє передачу даних у пачках, але при цьому тривалість пачки буде обмежена. Припустимо, що пропускна здатність вихідного інтерфейсу, який моделюється виходом сервера, дорівнює R. У такому випадку сервер не зможе передавати дані на вихід зі швидкістю, яка буде перевищувати R біт/с. Можна побачити, що при будь-якому інтервалі часу t середня швидкість потоку, що виходитиме із сервера, буде дорівнювати мінімальній із двох величин: R і $r+b/t$. При високих значеннях t швидкість вихідного потоку наближатиметься до r – це є свідченням того, що алгоритм може забезпечити потрібну середню швидкість. Так само протягом невеликого часу t пакети мають можливість виходити із сервера зі швидкістю, що перевищує r. У випадку $r + b/t < R$ вони виходитимуть із сервера зі швидкістю $r + b/t$, у іншому випадку інтерфейс обмежить цю швидкість до величини R. Період часу t відповідатиме пульсації трафіка. Така ситуація спостерігатиметься у випадку, коли на протязі деякого часу пакети не мали можливості надходити

до сервера, привівши до того, що «кошик» повністю наповнився маркерами (часу, більшого за *b/r*). Якщо наступною до входу сервера прийде велика послідовність пакетів, що йтимуть один за одним, то ці пакети будуть передаватися на вихід зі швидкістю вихідного інтерфейса *R* так само один за одним і без інтервалів. Максимальний розмір часу такої пульсації складатиме *b/(R-r)* секунд. Після цього обов'язково наступить пауза, що є необхідною для заповнення пустого «кошика». Обсяг пульсації складатиме $R_b/(R-r)$ байт. З приведеного співвідношення можна побачити, що алгоритм «кошика маркерів» працює погано, у випадку, коли середня швидкість *r* є близькою до пропускної здатності вихідного інтерфейса. У такому випадку пульсація продовжуватиметься доволі довго, що робить алгоритм неефективним.

2.1.1.2 Алгоритм «дірявого відра» (leaky bucket)

Для того, щоб можна було контролювати угоду про параметри якості обслуговування всі комутатори мережі Frame Relay підтримують алгоритм «дірявого відра» (leaky bucket). Такий алгоритм, так само як і алгоритм «кошика маркерів», надає можливість контролю середньої швидкості і пульсації трафіка, але це реалізується іншим способом.

Алгоритм передбачає, що трафік буде контролюватися кожні *T* секунд. На усіх цих інтервалах часу трафік повинен мати середню швидкість не більше за обумовлену швидкість *CIR*. Ця швидкість контрольована на базі підрахунку загального обсягу даних, які надійшли за період *T*. У випадку, якщо цей обсяг є меншим або дорівнює B_c, то фактична швидкість трафіка була менше B_c / T, іншими словами менше *CIR*. Якщо обсяг пульсації перевищує обумовлене значення B_c на величину B_e , то таке перевищення вважається «м'яким» порушенням – пакети-порушники повинні бути позначені *DE=1*, але не мають бути відкинуті. Під час перевищення обсягом пульсації величини $B_c + B_e$ кадри будуть відкидатися.

Алгоритм (рис. 2.5) використовує лічильник байт *C*, що надходять від користувача. Кожні *T* секунд лічильник зменшується на величину B_c (або скидається до 0, у разі, коли значення лічильника менше, ніж B_c). Частіше за все це ілюструється «відром», з якого кожні *T* секунд, дискретно, витікає обсяг, що є рівним мінімальному з чисел B_c або *C*.

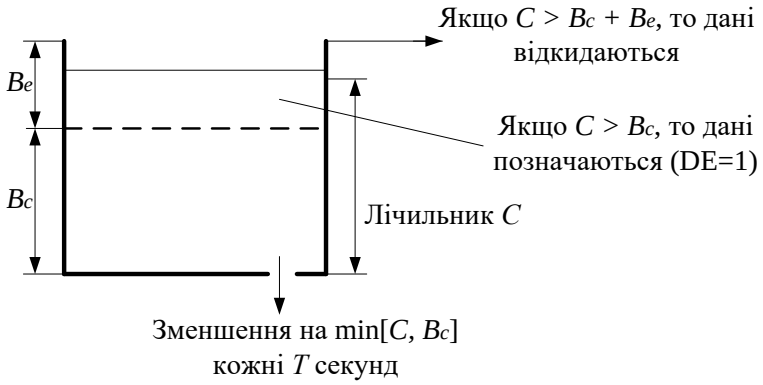

Рисунок 2.5 – Алгоритм «дірявого відра»

Всі кадри, що не збільшували значення лічильника своїми даними вище порогу B_c, пропускатимуться до мережі з позначкою *DE=0*. Кадри, чиї дані привели до значення лічильника, більшого за B_c, але меншого за B_c+B_e, також будуть передаватися до мережі, але зі значенням позначки *DE=1*. А кадри, які привели до значення лічильника більше B_c+B_e, будуть відкинуті комутатором.

Користувач може використовувати не всі параметри якості обслуговування для віртуального каналу, а тільки частини з них. Наприклад, є можливість використовувати тільки параметри *CIR* і B_c. Такий варіант може надавати більш якісне обслуговування, через те що кадри в жодному разі не будуть відкидатися комутатором одразу. Комутатор позначатиме тільки ті кадри, які будуть перевищувати поріг B_c за час *T*, позначкою *DE=1*. Якщо в мережі не буде перевантажень, то кадри в такому каналі завжди надходять до вузла призначення, навіть у випадку, коли користувач систематично порушує договір з мережею.

Як можна побачити з описаного, алгоритм «дірявого відра» більш «суворо» бере під контроль пульсації трафіка, за алгоритм «кошика маркерів». Алгоритм «кошика маркерів» надає можливість трафіку в періоди пониженої активності набирати обсяг пульсації, а після цього використовувати ці накопичення під час сплесків трафіка. При використанні алгоритму «дірявого відра» такої можливості не буде, через те що лічильник C скидається до 0 за примусом в кінці кожного періоду T незважаючи на кількість байтів, що надійшли від користувача до мережі на протязі цього періоду. Ще одним розходженням двох алгоритмів є те, що під час переповнення «маркерного кошика» алгоритм буде ігнорувати маркери, але ні в якому разі не буде відкидати пакети. Алгоритм «дірявого відра», навпаки, під час переповнення викидатиме самі пакети.

2.1.2 Алгоритми запобігання перевантажень

Алгоритми управління чергами (buffer management) приймають рішення в який час і який пакет потрібно відкинути задля запобігання створення чи збільшення ступеня перевантаження. При оцінюванні функціонування алгоритму управління чергами враховується його здатність ефективного контролю трафіку під час перенавантаження. Рішення відкидати пакет зазвичай приймається коли цей пакет надходить в систему, або коли утворюється перевантаження, при якому є вірогідність відкидання пакетів, які вже знаходяться в чергах для того, щоб звільнився буферний простір для нових високоприорітетних пакетів.

2.1.2.1 Алгоритми пасивного управління чергами

Часто використовуваним і в той же час найменш складним алгоритмом для управління чергами є Tail Drop, що, як правило, реалізований за допомогою черги з правилом обслуговування «перший прийшов – перший обслуговуваний» (FCFS, First Come First Served). Вхідний пакет відкидається тільки у тому разі, коли в черзі вже немає вільних місць. Коли в черзі з’являється вільне місце – його

займає пакет, що прийшов самим першим з того часу як у черзі звільнилося місце.

Через свою простоту Tail Drop має ряд серйозних недоліків, серед яких варто відмітити, що можливе заповнення черги пакетами одного або кількох потоків, тобто «справедливий розподіл ресурсів» не реалізується. Також одним з серйозних недоліків є неможливість передбачення моменту перевантаження завчасно, тож констатація перевантаження є можливою тільки після того, коли перший пакет відкидається через переповнення черги.

Отже, така черга досить довго може бути цілком або майже повністю завантажена, та джерела через, які виникає навантаження не будуть інформуватися про це. У такому випадку надходження стеку пакетів від якогось джерела TCP, коли воно не має інформації про перевантаження чи високе навантаження черги і вікно cwnd має досить великий розмір, може привести до відкидання частини пакетів, а можливо навіть усіх, що були надіслані. У наслідок цього може статися «глобальна синхронізація» (global synchronization), тобто всі джерела TCP, які пересилають дані через маршрутизатор через виявлене перевантаження швидко і водночас зменшать розмір вікна cwnd, через що помітно знизиться навантаження черги і ресурси не будуть задіяні; через деякий час ці джерела знову ж будуть підвищувати навантаження одночасно, і це призведе до нового перевантаження. Через це моменти перевантаження чергуватимуться з моментами невисокого вжитку ресурсів, тобто середня пропускна здатність TCP-з'єднання буде невисокою, що призведе до неефективного використання ресурсів.

Наявні в мережних пристроях буфери можуть дозволити обробку «пачкового» трафіку. Відповідно до [RFC2309] головне завдання буфера - прийом і обробка пачок пакетів. В теорії для того, щоб буфер мав можливість прийняти пачку доволі великого розміру, потрібно збільшити його. Але при розширенні буфера збільшується і час який пакети перебувають у черзі, а протокол TCP після закінчення часу визначеного таймеру буде зменшувати розмір вікна cwnd і заново пересилати ті ж пакети, допустивши, що пакети, які все ще в буфері, уже втрачені, через що може виникнути перевантаження. Таким чином виходить, що розмір буфера, має бути як достатньо великим для надходження пачки пакетів, так і не занадто високим, для того щоб могла бути забезпечена задовільна якість обслуговування (обмежений розмір затримки).

Після виявлення проблем алгоритму Tail Drop і їх вивчення було розроблено декілька алгоритмів, що покращують функціонування цього алгоритму. Серед них необхідно відмітити наступні:

- «ймовірнісне скидання пакету» (random drop on full): при переповненні черги виконується обчислення ймовірності, з якою пакет відкидається;
- «скидання початку черги» (drop front on full): при переповненні черги виконується обчислення ймовірності, з якою перший в черзі на обслуговування пакет відкидається.

За допомогою цих стратегій можна обійти проблеми захоплення черги пакетами різних потоків, але залишається проблема неможливості передбачити можливе перевантаження. При певних умовах ці алгоритми можуть стати конкуренцією з складнішими алгоритмами активного управління чергами.

2.1.2.2 Алгоритми активного управління чергами. Алгоритм RED

Алгоритми активного управління чергами при розробці були орієнтовані на з'єднання TCP через те, що саме в протоколі TCP реалізовані механізми регулювання навантаження в залежності від стану мережі. Реалізація алгоритмів активного управління чергами дозволяє досягти ряду позитивних моментів:

- можливості уникнути проблеми захоплення черги пакетами одного чи декількох потоків;
- вирішує проблему неможливості завчасного виявлення перевантаження;
- забезпечити невисоку затримку пакета в буфері (по відношенню до Tail Drop);
- збільшується коефіцієнт використання мережевих ресурсів.

Алгоритм управління чергами «ймовірнісне завчасне виявлення перевантаження» (RED, Random Early Detection) надає можливість контролю навантаження в рамках черги маршрутизатора та при виявленні перевантаження чи стану близького до перевантаження виконує ймовірнісне скидання пакетів. Скидання пакета виконується на базі обчислення ймовірності, тому:

- можна дотримуватися принципу «справедливого розподілення ресурсів» і уникати виникнення проблеми заняття черги пакетами потоків;
- виконувати послідовне скидання пакетів, що належать різним з'єднанням, уникаючи «глобальної синхронізації».

Алгоритм RED є таким, що орієнтований протоколTCP та роботу з ним, тому скидання пакету дозволить джерелу навантаження зменшити розмір вікна cwnd і цим знизити навантаження. Розглянемо принцип функціонування базового алгоритму RED.

Алгоритм складається із двох частин: оцінка середнього розміру черги і прийняття рішення про скидання пакету, що надійшов. При надходженні кожного нового пакета, використовуючи фільтр низьких частот разом з «експоненціально зваженим ковзним середнім» EWMA (low-pass filter with an exponentially weighed moving average), RED визначає середній розмір черги *avg*. Після цього значення *avg* порівнюється з значеннями раніше визначених границь: верхньої max_*th* і нижньої min_*th*, що визначають степінь навантаження в черзі.

У випадку якщо значення avg належить інтервалу [*0;* min_*th*] - пакет що надходить поміщується в чергу, якщо ж значення avg належить інтервалу [min_*th;* max_*th*], то обчислюється ймовірність скидання пакета P_a, і або з ймовірністю P_a виконується скидання вхідного пакету, або з ймовірністю ($1 - P_a$) вхідний пакет поміщується в чергу. Якщо ж значення середнього розміру черги *avg* перевищує значення max_*th*, то вхідний пакет обов'язково відкидається. Ймовірність скидання пакета є функцією від змінної *avg*, тобто з підвищенням навантаження підвищується ймовірність скидання вхідного пакету. Графічна інтерпретація RED представлена на рисунку 2.6.

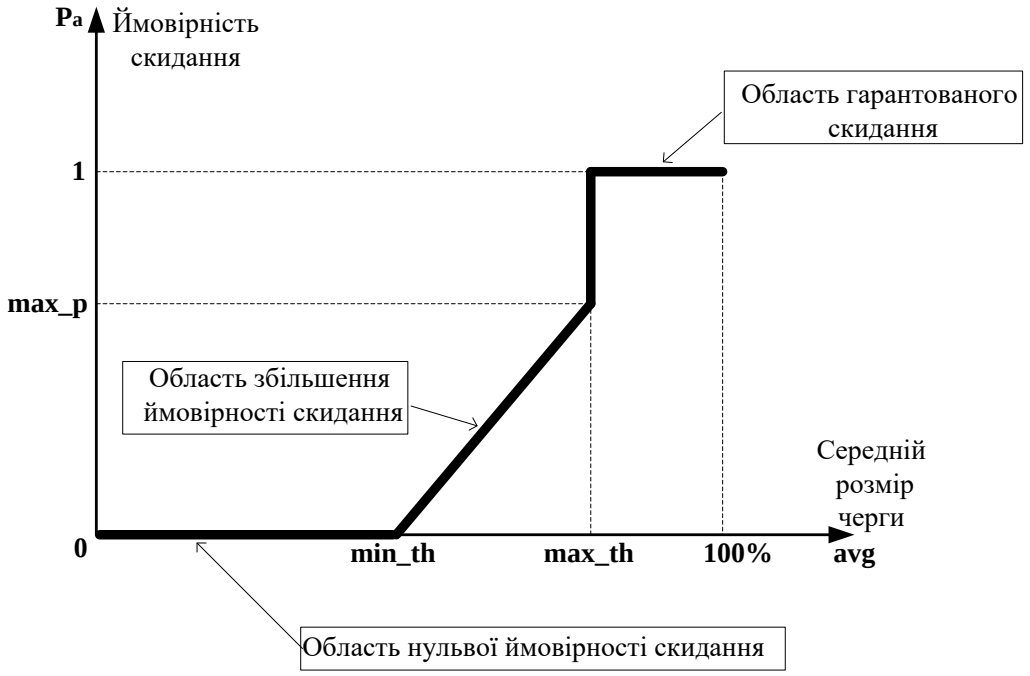

Рисунок 2.6 – RED: ймовірність втрати пакета збільшується з збільшенням середнього розміру черги

Таким чином, RED складається з двох різних алгоритмів: за допомогою алгоритму обчислення середнього розміру черги фактично обчислюється степінь пачковості навантаження, яка може бути прийнята буфером; алгоритм підрахунку ймовірності маркування або скидання пакета на основі відомого значення навантаження (тобто значення середньої довжини черги) визначає як часто маршрутизатор проводить маркування чи скидання пакетів. Ціллю функціонування RED маршрутизаторі є така реалізація маркування/скидання пакетів при якій буде дотримуватися принцип «справедливого розподілу ресурсів» і уникнення «глобальної синхронізації».

Щодо до принципу функціонування розширеного алгоритму RED, то саме розширена версія RED використовується для реалізації в реальних маршрутизаторах. Перед тим як перейти до принципу функціонування розширеного RED, необхідно визначити змінні які він використовує.

Глобальні змінні:

avg: середній розмір черги;
q_time: момент часу коли черга стала порожньою;
count: кількість пакетів, що надійшли до черги з моменту останнього скидання.

Фіксовані параметри алгоритму:

w: ваговий коефіцієнт черги;
min_*th*: нижня границя;
max_*th*: верхня границя
max_*p*: максимальне значення ймовірності P_a.

Інші параметри:

P_a: поточна ймовірність маркування/скидання пакета;

q: поточний розмір черги ;

time: поточний час;

f(time): лінійна функція часу.

Розширений алгоритм RED функціонує наступним чином. При активації алгоритму відбувається ініціалізація змінних *count* і *avg*. Далі з приходом кожного нового пакету в систему виконується ряд дій.

Спочатку обчислюється значення середнього розміру черги *avg*:

$$avg = (1 - w) \times avg_n + w \times q, \tag{2.1}$$

де avg_n – попереднє значення середньої довжини черги.

Якщо черга пуста, то оцінюється кількість пакетів невеликої довжини, що могли б бути передані в період відсутності пакетів в черзі:

$$avg = avg_n \times (1 - m) \tag{2.2}$$

Далі для обчислення середнього розміру черги avg припускається, що за час відсутності пакетів в черзі надійшло m пакетв. Якщо не зробити таке припущення, то значення avg буде обчислено неправильно.

Після підрахунку avg необхідно оцінити його значення. Якщо avg належить інтервалу [min_*th;* max_*th*], то обчислюється ймовірність маркування/скидання пакета P_b, яка лінійно змінюється в інтервалі від 0 до max_*p*:

$$P_b = \max\_ \ p \times (avg - \min\_ \ th) / (\max\_ \ th \times \min\_ \ th) \tag{2.3}$$

Ймовірність, на основі якої виконується маркування/скидання вхідного пакета обчислюється з використанням лічильника кількості пакетів, що надійшли до черги починаючи з моменту останнього скидання count – чим більше пакетів прийшло, тим вища ймовірність скидання. Такий підхід гарантує, що RED не буде чекати занадто довго, перш ніж виконає маркування/скидання пакета, тобто маркування/скидання виконується пропорційно до навантаження:

$$P_a = P_b / (1 - count \times P_b) \tag{2.4}$$

Якщо значення avg перевищує значення max_th, то вхідний пакет маркується або скидається в залежності від реалізації.

Додатковою можливою опцією алгоритму RED є вимірювання середнього розміру черги не в пакетах, а в байтах. В цьому випадку значення середнього розміру черги точно відображає затримку, що вноситься буфером в передачу пакета. При використанні цієї опції алгоритм має бути модифікований для того,

щоб ймовірність з якою пакет маркується/скидається, була пропорційною його довжині:

$$P_b = \max\_\ p \times (avg - \min\_\ th) / (\max\_\ th \times \min\_\ th) \quad (2.3)$$

$$P_b = P_b \times \frac{PacketSize}{MaximumPacketSize} \quad (2.\ 5)$$

$$P_a = P_b / (1 - count \times P_b) \quad (2.4)$$

Таким чином, пакети великого розміру, наприклад, пакети FTP, будуть маркуватися або скидатися з більшою ймовірністю, ніж пакети малою довжини, наприклад, пакети Telnet.

Для ефективного використання активного управління чергами, що реалізовані в маршрутизаторах, необхідно виконувати коректне налаштування параметрів алгоритмів. Налаштування параметрів алгоритму неправильно може привести до суттєвого погіршення параметрів функціонування маршрутизатора. Налаштування потребують наступні параметри:

- розмір буфера;
- ваговий коефіцієнт ;
- нижня границя;
- верхня границя;
- максимальне значення ймовірності скидання.

Значення розміру буфера має вплив на затримку пакета в маршрутизаторі, через що його розмір має бути зконфігурований у відповідності зі значенням максимальної затримки, яку може отримати пакет при надходженні в маршрутизатор. Ваговий коефіцієнт *w* визначається виходячи з гіпотетичних максимальних значень розміру і тривалості пачок пакетів, що можуть бути прийняті без втрат чергою на обслуговування. Якщо значення w високе, то є загроза, що алгоритм обчислення середнього розміру черги avg не матиме можливості фільтрації короткочасного перевантаження. Але якщо значення *w* дуже мале, то його вплив на динаміку avg також невелике, тому значення *avg* не зможе достатньо швидко реагувати на зміну поточного розміру черги *q*, і як наслідок RED не буде мати можливості визначити стан, що є близьким до перевантаження.

Для того, щоб визначити значення нижньої і верхньої границь необхідно брати до уваги, що вони залежать від бажаного максимального середнього розміру черги. Якщо припустити, що навантаження буде мати достатньо високу пачковість , то значення min_*th* має бути достатньо великим для того щоб степінь використання каналу зберігалася на прийнятно високому рівні. Визначення значення max_*th* напряму залежить від значення середньої затримки пакета в черзі: чим вище значення max_*th*, тим більше значення середньої затримки. Існує і рекомендоване загальне співвідношення значень границь: значення max_*th* має

бути, як мінімум, в два раз більше за значення min_*th*.

Налаштування параметрів алгоритму RED є достатньо складним завданням. Значення параметрів залежать, в першу чергу, від степені пачковості і характеру навантаження, яка буде надходити до буфера, в кому реалізований RED, і від конфігурації мережі.

2.1.2.3 Алгоритм ARED

Очевидно, що швидкість з якою буде зростати навантаження в черзі, залежить від кількості TCP з'єднань, що передають дані через маршрутизатор. У випадку коли кількість цих з'єднань невелика, швидкість зростання навантаження буде відносно низькою, тому значення параметра *w* повинно бути порівняно малим. Але вжиток одного і того ж самого низького значення *w* для випадку з великою кількістю з єднань TCP може призвести до неефективного функціонування RED по причині того, що є вірогідність, що алгоритм не матиме можливості реагувати на короткочасне перевантаження. З іншого боку, встановлення занадто високого значення w може призвести до занадто високої ймовірності, що пакети будуть скинуті навіть під час проходження через маршрутизатор кількох з'єднань TCP. Виходячи з цього можна припустити, що динамічна зміна параметрів RED дозволить адаптувати алгоритм до динамічного навантаження. З цією метою був запропонований адаптивний алгоритм RED (Adaptive RED, ARED), базовим принципом якого являються параметри, що змінюються динамічно. Обчислення нових значень параметрів виконується на основі даних про навантаження за визначений проміжок часу. ARED вимірює значення параметра max_*p* динамічно, беручи за основу недавні значення середньої довжини черги *avg*.

Розглянемо алгоритм функціонування ARED. Якщо середня довжина черги приймає, що єменшим за значення нижньої границі min_*th*, то параметр max_*p* приймає достатньо мале значення – припускається , що інтенсивність вхідного навантаження мала (рисунок 2.7). Також очевидно, що при ініціалізації ARED параметр max_p приймає досить мале значення в через те, що черга пуста, тобто її середній розмір менший ніж нижня границя. Як тільки середній розмір черги починає перевищувати max_*th*, то обчислюється нове значення параметра max_*p*, яке суттєво вище за попереднє.

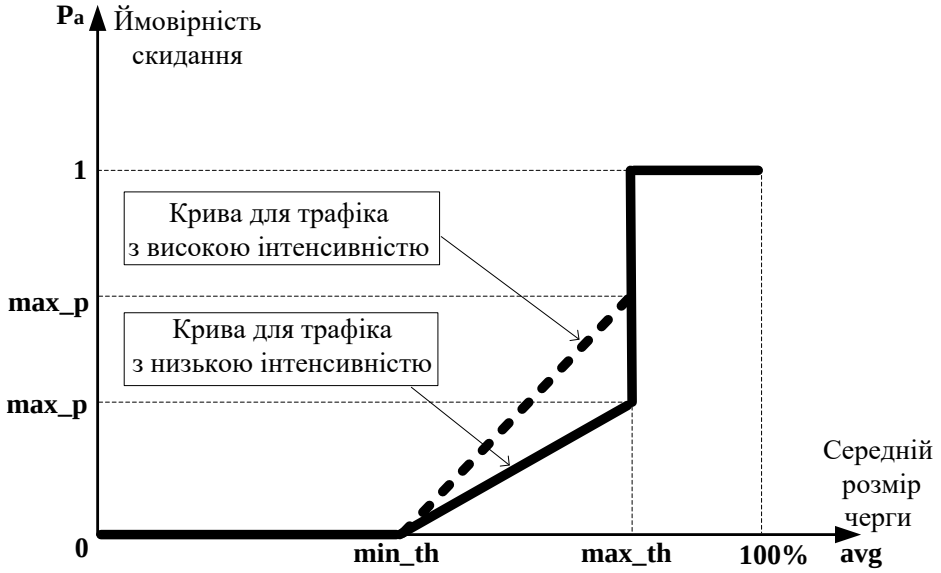

Рисунок 2.7 – AREDзмінює значення параметра max_p в залежності від інтенсивності навантаження

Таким чином AREDадаптується до зростаючого навантаження – ймовірність скидання вхідного пакета лишається меншою за одиницю, в той час якщо б в розглянутому маршрутизаторі був застосований RED, то ймовірність скидання вхідного пакета дорівнювала б одиниці. Достатньо важливим аспектом в роботі AREDє його поведінка в областях, що є близькими до верхньої та нижньої границь. У випадку якщо середнє значення розміру черги коливається в межах нижньої границі, то AREDпоступово знижує значення параметра max*_p*, так як навантаження невисоке і ймовірність скидання вхідного пакета можна знизити. У випадку якщо *avg* коливається в межах верхньої границі max*_th*, то AREDпоступово збільшу значення max*_p*, так як,можливо, для запобігання перевантаження необхідно підвищити ймовірність скидання пакета.

2.1.2.4 Алгоритм WRED

Відмінність WREDвід REDполягає лише в тому, що для пакетів кожного пріоритету визначений окремий набір параметрів. Наприклад,розглянемо найпростіший випадок, коли в чергу маршрутизатора поступають пакети з двома пріоритетами – далі під пакетом з низьким пріоритетом будемо розуміти промаркований пакет, а під пакетом з високим пріоритетом – звичайний непромаркований пакет. Функція залежності ймовірності скидання пакета від значення середнього розміру черги поводить себе наступним чином: при однаковому навантаженні повинен дотримуватися принцип, що ймовірність скидання промаркованого пакета повинна перевищувати ймовірність скидання звичайного пакета. Параметризація алгоритму WREDдостатньо складна, так як кількість значень, які необхідно встановити, прямо пропорційна кількості пріоритетів з якими пакети можуть надходити в маршрутизатор. Наприклад, якщо врахувати, що при реалізації AF PHBкількість пріоритетів може зрости до 12, то для повної параметризації WREDнеобхідно встановити значення 12×{12×min*X_th,* max*X_th,* max*X_p, avgX}* параметрів, де X – номер пріоритету.

Окремий випадок функції ймовірності скидання пакета для двох пріоритетів алгоритму WRED представлений на рисунку 2.8.

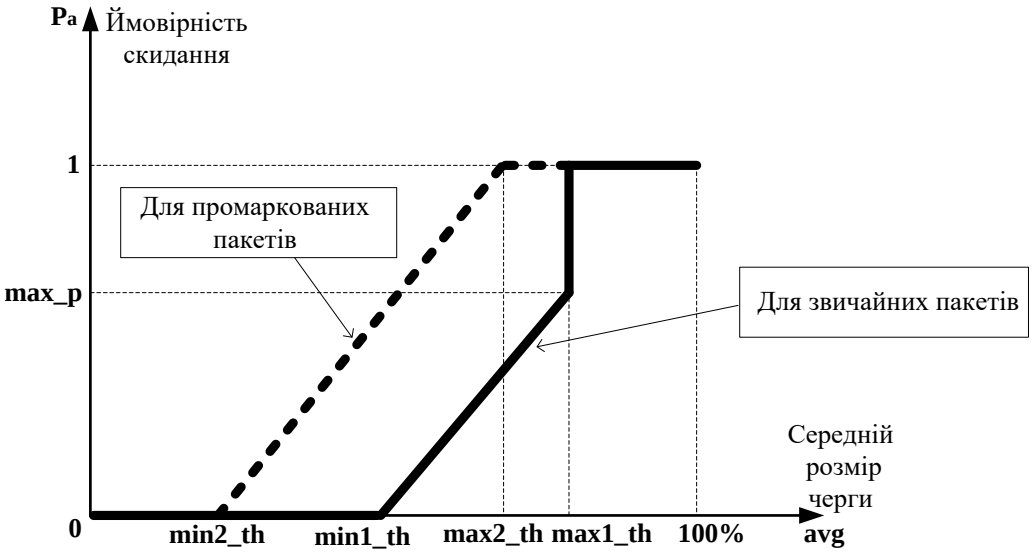

Рисунок 2.8 – WRED: маркування пакетів може змінити ймовірність скидання пакета

2.1.2.5 Алгоритм RIO

Алгоритм управління чергою RIO (RED In & Out) являє собою повний аналог алгоритму RED. Різницею є те, що він параметризується двома наборами значень параметрів – кожний набір для окремого класу пакетів. Передбачається що існує два класи пакетів – непромарковані (In-пакети), тобто коли пакет належить потоку, значення параметрів якого не перевищують заздалегідь визначених величин, і промарковані (Out-пакети) – параметри потоку перевищують передчасно визначені величини. RIO відрізняє обидва класи пакетів використовуючи аналіз заголовка і завдяки цьому може виконувати скидання Out-пакетів з більшою ймовірністю, ніж In-пакетів, таким чином забезпечуючи захист від скидання In-пакетів.

Через те, що алгоритм RIO являється модифікацією алгоритму RED, очевидно, що ці алгоритми мають однакові принципи функціонування. Фактично RIO являє собою ускладнену версію RED, так як існує необхідність контролювати два набори параметрів: один – для Out-пакетів, інший – для In-пакетів.

Принцип роботи алгоритму RIO. При надходженні пакетів в чергу необхідно визначити до якого класу відноситься даний пакет. Якщо надходить In-пакет , то обчислюється значення середнього розміру черги *avg_in*, яка і буде складатися з In-пакетів. Якщо надійшов Out-пакет, то виконується обчислення середнього розміру всієї черги *avg_total*. Тут варто відмітити важливу деталь, що ймовірність скидання In-пакетів залежить від середнього розміру черги, яка складається тільки із непромаркованих пакетів, в той час як ймовірність скидання Out-пакету залежить від середнього розміру черги яка складається із пакетів обох класів. Всі пакети незалежно від їх класу поступають в одну фізичну

чергу типу FIFO. Черга, що складається виключно з In-пакетів, не існує фізично, а її розмір розраховується за допомогою спеціальної процедури. На рисунку 2.9 представлені функції ймовірності скидання пакета в залежності від середнього розміру черги для In- і Out-пакетів. Для кожного з класів пакетів потрібно визначити такі параметри функціонування:

min_*in*: нижня границя для In-пакетів;
max_*in*: верхня границя для In-пакетів;
min_*out*: нижня границя для Out-пакетів;
max_*out*: верхня границя для Out-пакетів;
max_*in_p*: максимальне значення ймовірності скидання для In-пакетів;
max_*out_p*: максимальне значення ймовірності для Out-пакетів.

Таким чином, визначаються набори параметрів на базі яких обчислюється ймовірність скидання вхідного пакета: для In- пакетів {min_*in*, max_*in*, max_*in_p*, avg_*in*} і для Out-пакетів {min_*out*, max_*out*, max_*out*_p, *avg_out*}.

З цих параметрів визначаються три області функціонування системи відносно In-пакетів:

- «нормальне функціонування» $avg\_in \in [0; \min\_in]$;
- «запобігання перевантаженню» $avg\_in \in [\min\_in; \max\_in]$;
- «управління перевантаженням» $avg\_in \in [\max\_in; \infty]$.

А також три області функціонування системи відносно Out-пакетів:

- «нормальне функціонування» $avg\_total \in [0; \min\_out]$;
- «запобігання перевантаженню» $avg\_total \in [\min\_out; \max\_out]$;
- «управління перевантаженням» $avg\_total \in [\max\_out; \infty]$.

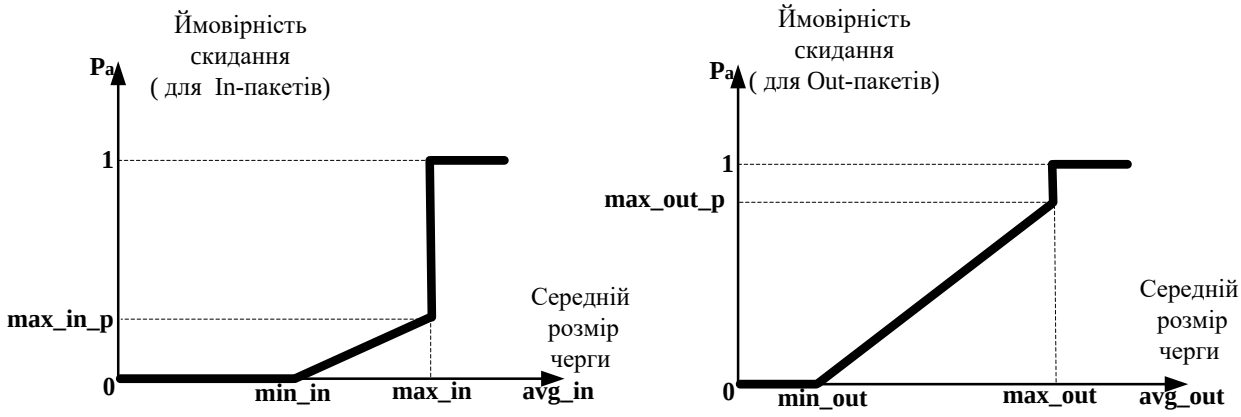

Рисунок 2.9 – RIO: функції ймовірності скидання пакета в залежності від середнього розміру відповідної черги

Існує три базових принципи, спільна реалізація яких забезпечить ефективне функціонування RIO:

- З самого початку значення нижньої границі для In-пакетів має бути встановлено більшим за значення нижньої границі для Out-пакетів, тобто

$\min\_ out < \max\_ in$. Це співвідношення дозволяє бути впевненим, що маршрутизатор з реалізованим алгоритмом RIO, при досягненні певного навантаження починає скидати тільки Out-пакети;

- під час фази «запобігання перевантаженню» кількість скинутих Out-пакетів буде перевищувати кількість скинутих In-пакетів, якщо виконується умова $\max\_ in\_p < \max\_ out\_p$;
- якщо встановити значення верхньої границі для In- пакетів значно більшим, ніж значення верхньої границі для Out-пакетів, тобто $\min\_ in >> \max\_ out$, то фаза «управління перевантаженням» для Out-пакетів наступить суттєво раніше, і тому в цей момент всі Out-пакети будуть скидатися.

Таким чином, коротко, динаміку поведінки RIO можна описати наступним чином. При достатньо низьких навантаженнях RIO не вмішується в процес надходження і обслуговування пакетів. Як тільки навантаження досягає певного рівня, неявно заданого параметром min_*out*, то RIO починає виконувати ймовірнісне скидання Out-пакетів. З ростом навантаження процент скинутих Out-пакетів зростає. При визначенні стану достатньо близького до перевантаження, In- і Out-пакети можуть скидатися почергово, але RIO забезпечує значення ймовірності скидання In- пакетів суттєво нижчою, ніж для Out-пакетів. Однак, при правильно зконфігурованому RIO такого випадку відбутися не повинно, тільки у випадку, коли в черзі знаходиться тільки In-пакети, вхідний In-пакет може бути скинутий.

2.1.3 Алгоритми обслуговування черг

Механізми черг використовуються в усіх мережевих пристроях, де застосовується комутація пакетів – маршрутизаторах, комутаторах локальних або глобальних мереж, кінцевих вузлах. Черги стають необхідними, коли з'являються тимчасові перевантаження, якщо мережний пристрій не може вчасно передавати поступаючі пакети на вихідний інтерфейс. Причиною перевантаження може бути процесорний блок мережного пристрою. В такому випадку ще не оброблені пакети на деякий час поміщаються у вхідну чергу, тобто в чергу на вхідному інтерфейсі. Якщо ж причиною перевантаження є обмежена швидкість вихідного інтерфейсу (а вона не може мати швидкість вище протоколу, що підтримується), то пакети на деякий час зберігаються у вихідних чергах.

Оцінювання можливої довжини черги в мережному пристрою дозволила б виміряти параметри якості обслуговування при наявних характеристиках трафіку мережі. Але зміна черг являє собою імовірний процес, який залежить від багатьох факторів, особливо при складних алгоритмах обробки черг відповідно пріоритетів, що були задані або використовуючи зважене обслуговування різних потоків. За аналіз черг відповідає окрема область прикладної математики – теорія масового обслуговування (queuing theory), але за допомогою неї можна отримати тільки кількісні оцінки для найпростіших ситуацій, що не підходять для

реальних умов роботи мережних пристроїв. Тому QoS-служба користується доволі складною моделлю для підтримки гарантованої якості обслуговування, вирішуючи завдання в комплексі. Це робиться за допомогою таких методів:

- попереднє резервування пропускної здатності для трафіку з відомими параметрами (наприклад, для середніх значень інтенсивності і величини блоку пакетів);
- примусове профілювання вхідного трафіку для утримання коефіцієнта навантаження пристрою на потрібному рівні;
- використання складних алгоритмів керування чергами.

Частіше за все в маршрутизаторах і комутаторах використовуються наступні алгоритми обробки черг:

- традиційний алгоритм FIFO;
- пріоритетне обслуговування (Priority Queuing);
- настроювані черги (Custom Queuing);
- зважене справедливе обслуговування (Weighted Fair Queuing, WFQ).

Кожен з алгоритмів розроблений для вирішення певних завдань і через це специфічним чином має вплив на якість обслуговування різнотипного трафіку в мережі. Можливе комплексне застосування цих алгоритмів.

Принцип роботи алгоритму FIFO полягає в наступному. У випадку перевантаження пакети надходять в чергу, а коли перевантаження усувається або зменшується, пакети надходять на вихід у тому порядку, в якому надійшли («першим прийшов - першим пішов», First In - First Out). Такий алгоритм обробки черг використовується на всіх пристроях з комутацією пакетів за замовчуванням. Його особливостями є простота реалізації і відсутність потреби в конфігуруванні, але він також має суттєвий недолік - диференційована обробка пакетів з різних потоків не реалізована. Черги FIFO потрібні для нормальної роботи мережних пристроїв, але вони не справляються з підтримкою диференційованого якості обслуговування.

2.1.3.1 Механізм пріоритетної обробки трафіку

Механізм пріоритетної обробки трафіку (Priority Queuing) передбачає поділ усього мережного трафіку на різні класи з призначенням кожному з цих класів деякої числової ознаки – пріоритету. Поділ на класи (класифікація) може проводитися різними способами.

Класифікація трафіку являється окремою задачею. Пакети можуть бути розбиті на класи за пріоритетами відповідно типу мережного протоколу, як наприклад, IP, IPX або DECnet (такий спосіб можливий тільки для пристроїв, які працюють на другому рівні), на підставі адрес одержувача і відправника, номера порту TCP/UDP і будь-яких інших комбінацій ознак, які містяться в пакетах. Правила класифікації пакетів на пріоритетні класи є складовою частиною політики управління мережею.

На рисунку 2.10 приведено приклад використання чотирьох пріоритетних черг: з високим, середнім, нормальним і низьким пріоритетами. Пріоритети черг мають абсолютний характер переваги при обробці: до того часу, коли з черги з більшим пріоритетом не будуть вибрані всі пакети, пристрій не перейде до обробки наступної черги, яка є менш пріоритетною. Через це пакети із середнім

пріоритетом у будь-якому випадку обробляються тільки у тому разі, коли черга пакетів з високим пріоритетом порожня, а пакети з невисоким пріоритетом – тільки коли порожні всі вище розташовані черги.

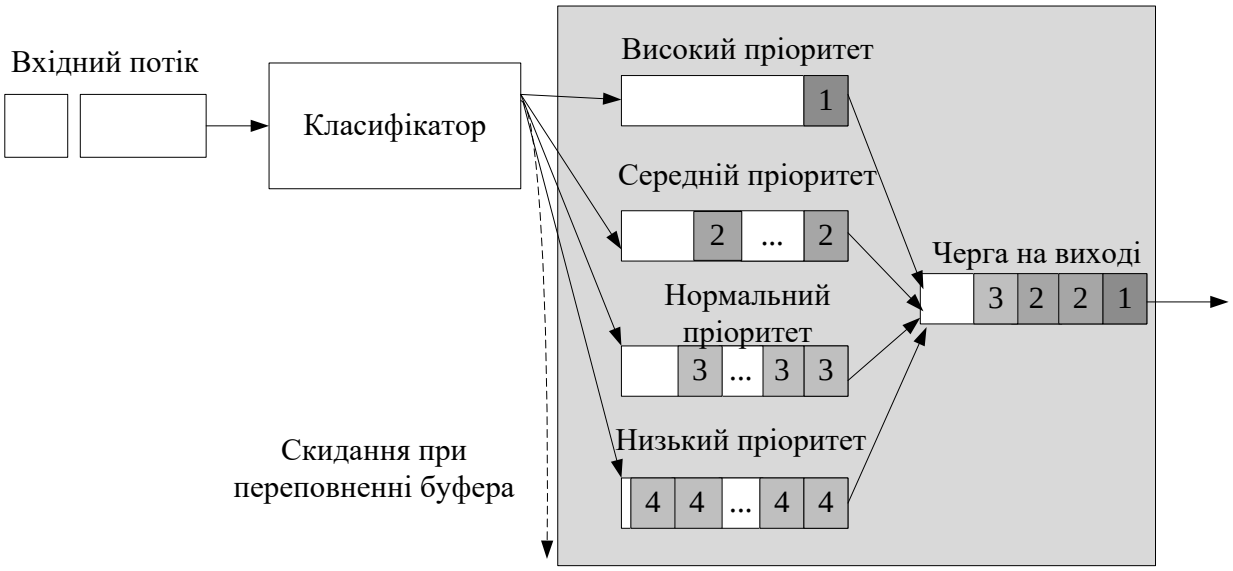

Рисунок 2.10 – Механізм Priority Queuing

Кінцевий розмір пам'яті буферу мережевого пристрою передбачає деяку граничну довжину кожної черги. Найчастіше для всіх пріоритетних черг за замовчуванням виділяються буфери з однаковим розміром, але значна частина пристроїв дозволяє адміністраторам виділяти для кожної черги індивідуальний буфер. Гранична кількість пакетів, що можуть бути збережені в черзі даного пріоритету визначається його максимальною довжиною. Пакет, що надійшов у той час, коли буфер заповнений, просто буде відкинутий.

Висока якість сервісу для пакетів з найпріоритетнішої черги забезпечується пріоритетним обслуговуванням черг. Якщо пропускна здатність вихідного інтерфейсу (і продуктивності внутрішніх блоків самого пристрою, які приймають участь в просуванні пакетів) не перевищується середньою інтенсивністю надходження пакетів до пристрою, то пакети, які мають найвищий пріоритет будуть отримувати потрібну їм пропускну здатність у будь-якому випадку. Щодо інших класів пріоритетів, то їх якість обслуговування нижче, ніж у пакетів з самим високим пріоритетом і може знижуватися досить істотно, у випадку коли дані з високим пріоритетом передаються з великою інтенсивністю.

Через це пріоритетне обслуговування зазвичай використовується тоді, коли в мережі є трафік, що чутливий до затримок, та його інтенсивність невелика, так що його наявність не занадто обмежує інший трафік. Наприклад, голосовий трафік є чутливим до затримок, але, зазвичай, його інтенсивність не перевищує 8-16 Кбіт/с і інші класи трафіку не постраждають при визначенні для нього найвищого пріоритету. Проте в мережі бувають різні ситуації. Наприклад, відеотрафік теж може вимагати обслуговування в першу чергу, але він має значно вищу інтенсивність. У таких випадках використовуються алгоритми

управління чергами, які надають низькопріоритетному трафіку деякі гарантії навіть тоді, коли підвищується інтенсивність високопріоритетного трафіку.

2.1.3.2 Механізм зваженого обслуговування

Алгоритм зважених черг (Weighted Queuing) розроблений для того, щоб для всіх класів трафіку можливо було надавати певний мінімум пропускної здатності або задовольняти вимоги до затримок. Під вагою якого-небудь класу розуміється частина пропускної здатності вихідного інтерфейсу, яка виділяється певному виду трафіку. Алгоритм, в якому адміністратор може призначати вагу класів трафіку, називається «настроюваною чергою» (Custom Queuing).

Як при зваженому, так і при пріоритетному обслуговуванні трафік поділяється на декілька різних класів, і для кожного з них вводиться окрема черга пакетів. З кожною з цих черг пов'язується частина пропускної здатності вихідного інтерфейсу, яка гарантується даному класу трафіку при перевантаженнях цього інтерфейсу. У прикладі, наведеному на рисунку 2.11, пристрій підтримує три черги для трьох класів трафіку. Цим чергам відповідає 20, 30, і 50% пропускної спроможності вихідного інтерфейсу при перевантаженнях.

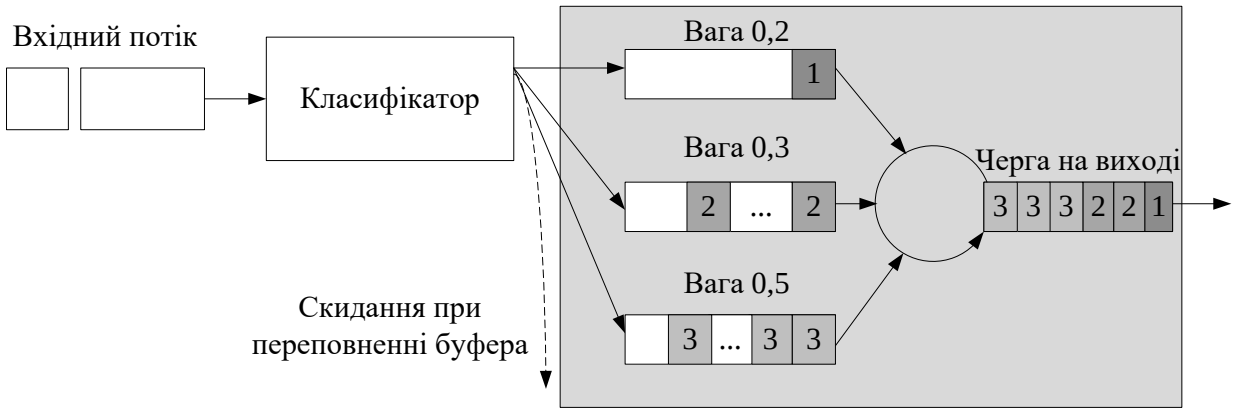

Рисунок 2.11 – Механізм Weighted Queuing

Поставлена мета може бути досягнута за допомогою обслуговування черг послідовно і циклічно, і в кожному з цих циклів з черг забирається така кількість байт, яка відповідає вазі кожної черги. Наприклад, у випадку, коли цикл перегляду черг у приведеному прикладі дорівнює 1 сек., а швидкість вихідного інтерфейсу становить 100 Мбіт /с, то у час перевантажень в кожному з циклів першої черги забирається 20 Мбіт даних, другої – 30 Мбіт, а третьої – 50 Мбіт.

Так, кожен з класів трафіка отримує гарантований мінімум пропускної здатності, що в більшості випадків є більш бажаним результатом, ніж обмеження низькопріоритетних класів пріоритетнішим.

2.1.3.3 Зважене справедливе обслуговування і його модифікації

Зважене справедливе обслуговування (Weighted Fair Queuing, WFQ) – це комплексний механізм обслуговування черг, який комбінує пріоритетне обслуговування зі зваженим. Виробники мережевого обладнання часто надають численні власні реалізації WFQ, які відрізняються способом призначення ваг і можливістю підтримувати різні режими роботи, тож в кожному конкретному випадку слід уважно вивчити всі деталі WFQ, яке підтримується на пристрої.

Найпоширеніша схема представлена на рисунку 2.12. Вона передбачає собою наявність однієї особливої черги, що обслуговується за пріоритетною схемою – завжди першочергово і до того часу, поки всі заявки з неї не будуть виконані. Ця черга може бути використана для повідомлень управління мережею, системних повідомлень і, можливо, пакетів найкритичніших і найвимогливіших додатків. У будь-якому разі передбачається, що такий трафік має не дуже високу інтенсивність, тому залишається велика частина пропускної здатності вихідного інтерфейсу для інших класів трафіку. Інші черги пристрій переглядає послідовно, відповідно алгоритму зваженого обслуговування. Адміністратор може визначити вагу для кожного класу трафіку так, як це виконується у випадку зваженого обслуговування. Стандартний варіант роботи передбачає для всіх інших класів трафіку однакові частини пропускної здатності вихідного інтерфейсу (за рахунок тієї що залишилася від пріоритетного трафіку).

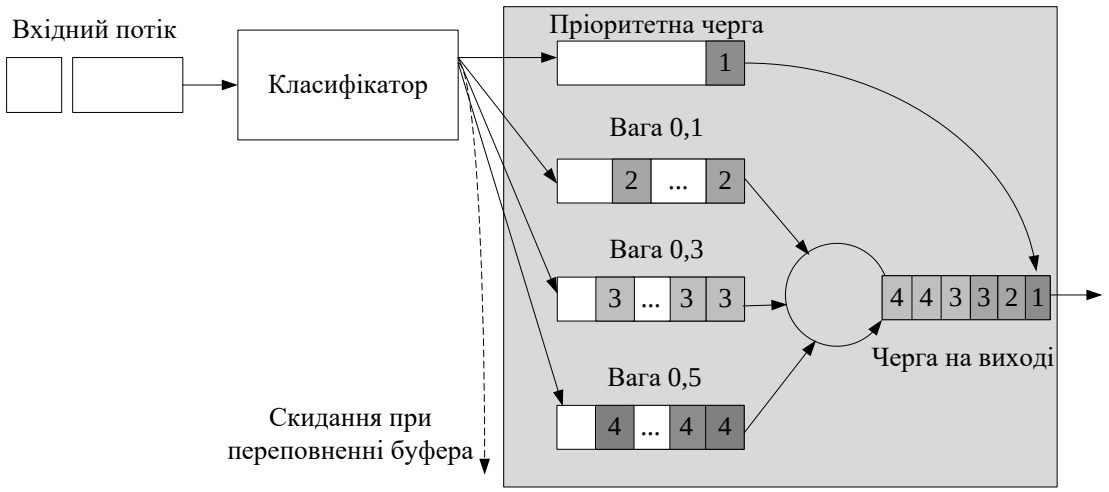

Рисунок 2.12 – Механізм Weighted Fair Queuing

Виробники обладнання покращують роботу механізму WFQ, додаючи деякі корисні режими. Наприклад, в маршрутизаторах компанії Cisco існує декілька різновидів WFQ:

- заснований на потоках (Flow-based) режим WFQ (FWFQ);
- заснований на класах (Class-based) режим WFQ (CBWFQ).

Для варіанту FWFQ на базі потоків в маршрутизаторі генерується стільки черг, скільки потоків використовується трафіком. В даному випадку під потоком розуміються пакети з певними значеннями IP-адрес відправника і одержувача та/або портів TCP/UDP відправника і одержувача, а також однаковими

значеннями поля ToS. Іншими словами, потік є послідовністю пакетів, що надходять від однієї програми з певними параметрами якості обслуговування, заданими в полі ToS. Кожному з потоків відповідає окрема вихідна черга, для якої механізм WFQ в періоди перевантажень надає однакові частки пропускної здатності порту. Тому іноді алгоритм FWFQ називають FQ (Fair Queuing) – справедливее обслуговування.

Варіант CBWFQ на базі класів в маршрутизаторах Cisco має два підваріанти:

- на підставі так званих груп QoS, відповідних набору ознак зі списку управління доступом (ACL), наприклад, номером вхідного інтерфейсу або номером хоста або підмережі, визначаються класи трафіку;
- значення полів ToS визначають класи трафіку.

Адміністратор задає максимальну довжину черги, а також ваги пропускної здатності для варіанту груп QoS, що був виділений для кожної групи QoS. Пакети, що не були віднесені до жодної з груп, будуть включені до групи 0. При визначенні ваг WFQ необхідно звернути увагу на наступне:

- групі QoS з номером 0 автоматично призначається 1% від всієї наявної пропускної здатності;
- загальна вага всіх інших груп не має бути більшою за 99%;
- пропускна спроможність, що лишається після визначення ваг буде виділена групі 0.

У варіанті класифікації на підставі значенні ToS передбачені ваги класів за замовчуванням. Якщо адміністратор явно не задав їх за допомогою команди weight, вони вступлять в силу. Для класифікації можуть бути використані два молодших біта трьохрозрядного підполя IP Precedence з поля ToS, таким чином, що в цьому варіанті існує всього чотири класи трафіку. Класу 0 виділяється 10% вихідної пропускної спроможності за замовчуванням, класу 1 – 20%, класу 2 – 30% і класу 3 – 40%. Чим вище клас, тим важливіший трафік, тож надання більшої частки пропускної спроможності створює для нього комфортніші умови просування.

У багатьох мережевих пристроях механізм WFQ є одним з основних для підтримки якості обслуговування, в тому числі у випадку різних протоколів, які використовують методи сигналізації для координованої поведінки всіх пристроїв мережі, наприклад, при застосуванні протоколу RSVP.

Low Latency Queuing. Алгоритм обробки черг з малою затримкою (Low Latency Queuing, LLQ) є модифікацією CBWFQ. Алгоритм (подібно до PQ) куристується виділеною чергою для обробки трафіку, що є чутливим до затримок. Інший трафік обробляється по алгоритму CBWFQ.

3 РОЗРОБКА РЕКОМЕНДАЦІЙ ЩОДО КОМПЛЕКСНОГО УПРАВЛІННЯ МЕРЕЖНИМИ РЕСУРСАМИ В СУЧАСНИХ МЕРЕЖАХ

3.1 Вихідні дані для проведення експериментального дослідження

В експерименті взаємодіяли три модулі. Модулі 1 та 3 представляли собою кінцеві станції абонентів. Відправником трафіка є модуль 1, а отримувачем модуль 3. Взаємодія кінцевих станцій здійснювалася через модуль 2, який являв собою ділянку мережі, представлену маршрутизаторами компанії Cisco серії C2900. В якості кінцевих станцій абонентів використовувалися два ноутбуки під управлінням операційних систем Windows 7 та Windows 10, які підключалися до маршрутизаторів безпосередньо за допомогою технології FastEthernet. Загальна схема натурного експерименту представлена на рис. 3.1.

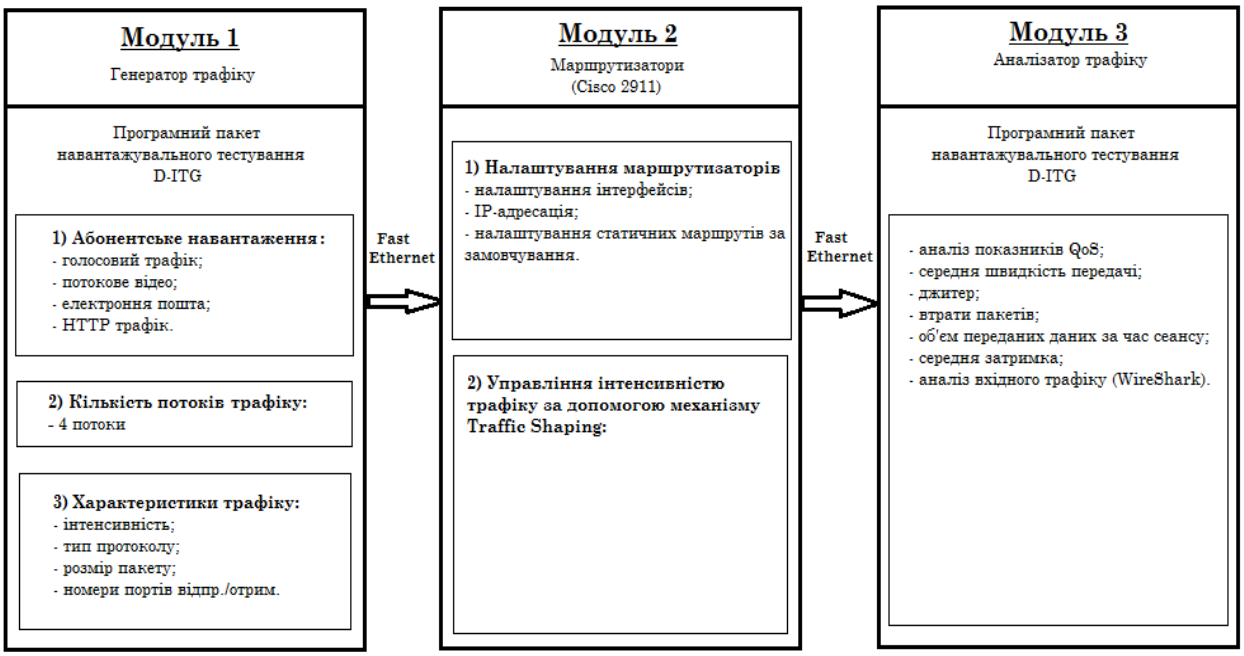

Рисунок 3.1 – Загальна схема натурного експерименту

Модуль 1 виконував генерування трафіка. За допомогою пакету навантажувального тестування D-ITG було згенеровано чотири потоки трафіку для чотирьох різних служб: потокове відео, VoIP, електронна пошта і HTTP. В модулі 2 були налаштовані інтерфейси маршрутизаторів, статична маршрутизація, класи трафіка, політики обслуговування класів трафіка. Модуль 3 виступав аналізатором трафіка. Завдяки вбудованим можливостям пакету D-ITG були отримані показники, по яким оцінювалася якість обслуговування для кожного трафіка відповідно до налаштованих в модулі 2 політик QoS: швидкість передачі даних, середня затримка, джитер, втрати пакетів.

Більш детальна інформація про використане мережне обладнання представлена в таблиці 3.1.

Таблиця 3.1 – Параметри маршрутизаторів

Пристрій	Платформа	Модулі	NV-RAM	Flash	DRAM	Версія IOS
Router 1 (R14)	Cisco2911	3 Gigabit Ethernet/IEEE 802.3 interface(s) 2 Serial (sync/async) network interfaces(s) – HWIC-2T	262136 bytes	256483328 bytes	483328K/40960K bytes	15.0(1r)
Router 2 (R15)	Cisco2911	3 Gigabit Ethernet/IEEE 802.3 interface(s) 2 Serial (sync/async) network interfaces(s) – HWIC-2T	262136 bytes	256483328 bytes	483328K/40960K bytes	15.0(1r)

Для з'єднань між маршрутизаторами використовувалися послідовні інтерфейси стандарту V.35. Топологія досліджуваної мережі приведена на рис. 3.2.

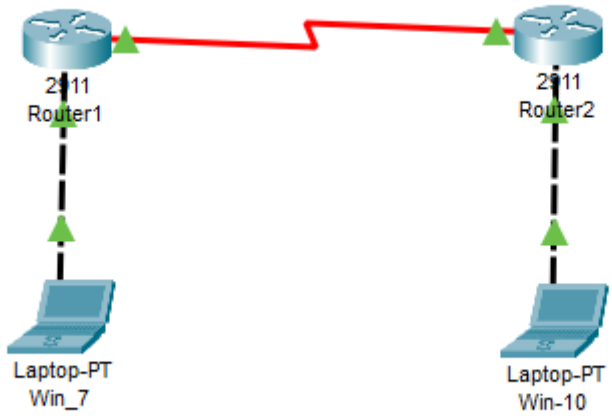

Рисунок 3.2 – Топологія досліджуваної мережі

При налаштуванні пропускної здатності вихідного каналу зв'язку на маршрутизаторах, з ціллю забезпечення потрібної фізичної швидкості передачі пакетів, була встановлена частота синхронізації (clock rate) рівна 2000000 біт/с. Частота синхронізації (clock rate) налаштована на таких інтерфейсах: R1(s0/0/0), R2 (s0/0/0). Пропускна здатність на всіх послідовних інтерфейсах становить 2 Мбіт/с. Інтерфейси Gigabit0/1 на маршрутизаторах R1 та R2 – 1 Гбіт/с.

3.2 Послідовність виконання експериментального дослідження

За допомогою пакета навантажувального тестування D-ITG, встановленого на кінцевих станціях, задавалась кількість потоків трафіку та їх параметри. На рис. 3.3 показаний приклад налаштувань для генерування 4 потоків трафіку (Flow1- Flow4) по протоколах UDP.

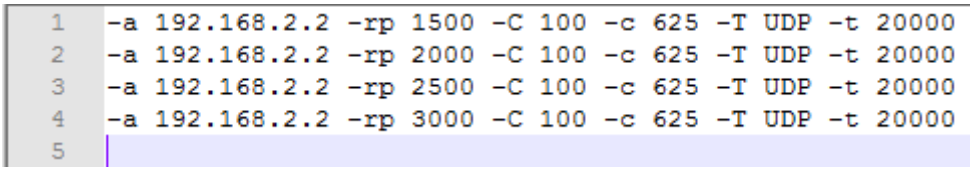

Рисунок 3.3 – приклад налаштувань для генерування 3 потоків трафіку

Потоки передаються по протоколу UDP. Під час виконання експериментального дослідження змінювалася інтенсивність передачі пакетів кожного потоку для встановлення сумарної швидкості передачі чотирьох потоків (B_{sum}) в такі умови:

$$B_{sum} < B_{\max} \quad (3.1)$$

$$B_{sum} \approx B_{\max} \quad (3.2)$$

$$B_{sum} > B_{\max} \quad (3.3)$$

де $B_{\max}$ – максимальна пропускна здатність на інтерфейсі.

Швидкість потоку (B_i) , згенерованого D-ITG, визначалася за формулою:

$$B_i = 8 \times c \times C \quad (3.4)$$

де c – розмір пакету в байтах;

C – кількість переданих пакетів за 1 секунду;

8 – кількість біт в 1 байті.

Сумарна ж швидкість по чотирьом потокам визначалася за формулою:

$$B_{sum} = 4 \times B_i \quad (3.5)$$

В якості механізму управління інтенсивністю був вибраний Traffic Shaping у зв'язку с тим, що він зменшує втрати пакетів, збільшуючи затримку.

Змінюючи B_{sum}, для виконання умов (3.1) – (3.3), були отримані результати по яким оцінювалася якість обслуговування кожного потоку трафіку: швидкість передачі даних, втрати пакетів, середня затримка та джитер.

На першому етапі експериментального дослідження згенерований трафік передавався через досліджувану мережу без використання будь-яких політик якості обслуговування, на всіх інтерфейсах топології використовувався механізм FIFO. Використана статична маршрутизація для зведення працюючих службових протоколів до мінімуму. Налаштування інтерфейсів та IP-адресації представлено нижче.

Маршрутизатор R14:

```
R14(config)#no cdp run
R14(config)#interface GigabitEthernet 0/1
R14(config-if)#ip address 192.168.1.1 255.255.255.0
```

```
R14(config-if)#no keepalive
R14(config-if)#no shutdown
!
R14(config)# interface s 0/0/0
R14(config-if)#ip address 10.1.1.1 255.255.255.252
R14(config-if)#bandwidth 2000
R14(config-if)#no keepalive
R14(config-if)#no shutdown
!
R14(config)#ip route 192.168.2.0 255.255.255.0 10.1.1.2
```

Маршрутизатор R15:

```
R15(config)#no cdp run
R15(config)#interface GigabitEthernet 0/1
R15(config-if)#ip address 192.168.2.1 255.255.255.0
R15(config-if)#no keepalive
R15(config-if)#no shutdown
!
R15(config)# interface s 0/0/0
R15(config-if)#ip address 10.1.1.2 255.255.255.252
R15(config-if)#bandwidth 2000
R15(config-if)#clock rate 2000000
R15(config-if)#no keepalive
R15(config-if)#no shutdown
!
R15(config)# ip route 192.168.1.0 255.255.255.0 10.1.1.1
```

Експеримент був проведений з такими значеннями B_{sum}: 500 Кбіт/с, 2 Мбіт/с, 3 Мбіт/с. Тривалість експерименту була вибрана 20 секунд для кожного трафіку. Адреса кінцевої станції яка генерувала трафік – 192.168.1.2/24, а адреса станції призначення – 192.168.2.2/24. Після проведення експерименту для кожної з умов (3.1) – (3.3) були отримані результати мережних характеристик по кожному потоку трафіка (таблиця 3.2).

Таблиця 3.2 – Результати мережних характеристик для кожного потоку трафіка після першого етапу експериментального дослідження

Bsum, Кбіт/с	Flow1	Flow2	Flow3	Flow4
	B1, Кбіт/с	B2, Кбіт/с	B3, Кбіт/с	B4, Кбіт/с
500	125,2	125,2	125,2	125,2
2000	467,8	471,5	464,1	480,8
3000	61,1	61,4	61,3	61,5
а) Average Bitrate				

Bsum, Кбіт/с	Flow1	Flow2	Flow3	Flow4
	PD, пак.	PD, пак.	PD, пак.	PD, пак.
500	0	0	0	0
2000	88	73	104	36
3000	1561	1171	1042	544
б) Packet dropped				

Bsum, Кбіт/с	Flow1 AD, сек.	Flow2 AD, сек.	Flow3 AD, сек.	Flow4 AD, сек.
500	0,21	0,3	0,23	0,29
2000	0,29	0,34	0,3	0,35
3000	1,19	1,41	1,29	1,82

в) Average delay

Bsum, Кбіт/с	Flow1 AJ, сек.	Flow2 AJ, сек.	Flow3 AJ, сек.	Flow4 AJ, сек.
500	0,001	0,001	0,003	0,002
2000	0,005	0,004	0,005	0,004
3000	0,004	0,004	0,004	0,005

г) Average jitter

Для наглядного відображення результатів отриманих в ході першого етапу експериментального дослідження приведені графіки (рис.3.5), на яких показано як змінюються значення мережних характеристик (швидкість передачі даних, втрати, затримка, та джитер) для кожного потоку трафіка в залежності від виконання умов (3.4) – (3.7).

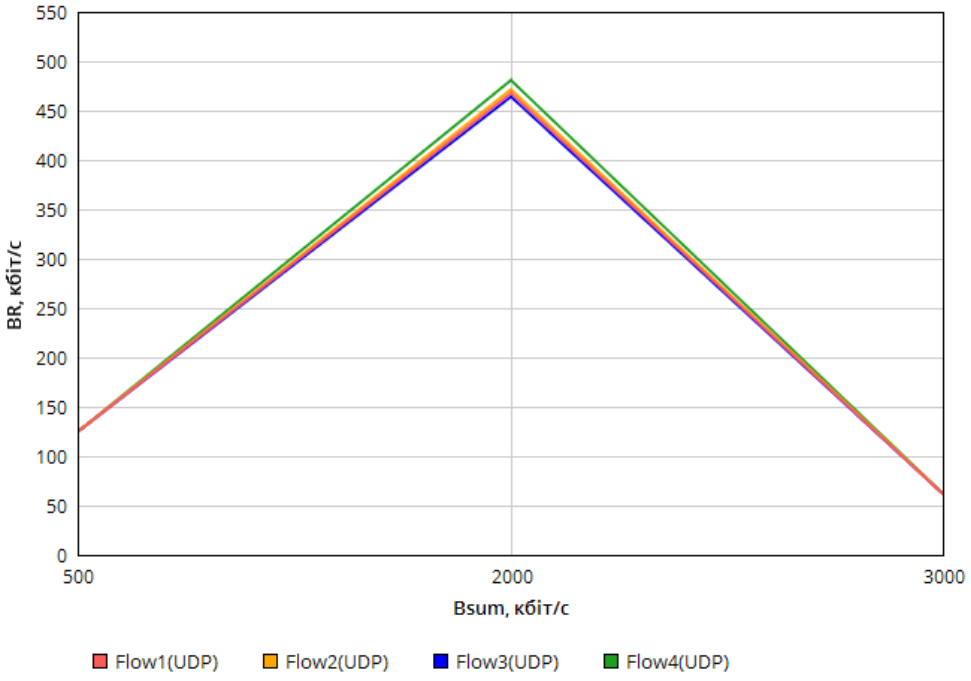

Рисунок 3.4 – Графік залежності швидкості передачі даних від сумарної швидкості потоків

BR (bitrate) – швидкість передачі даних.

Bsum – сумарна швидкість 4-х потоків.

Bmax = 2000 кбіт/с – максимальна пропускна здатність на інтерфейсі.

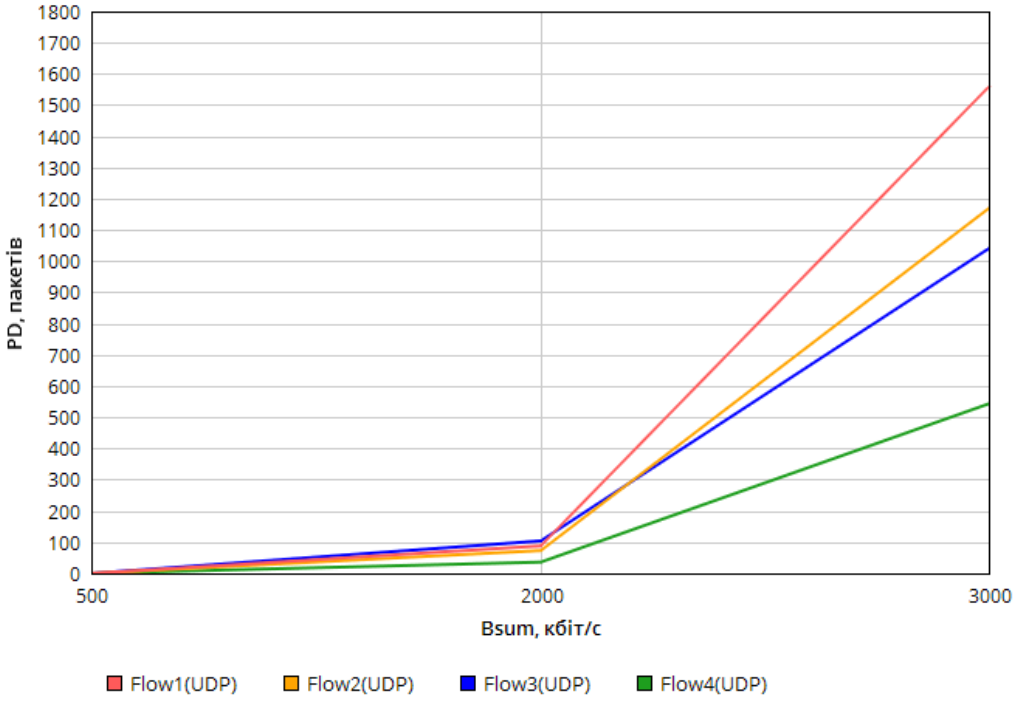

Рисунок 3.5 – Графік залежності відкинутих пакетів від сумарної швидкості потоків

PD (packet dropped) – кількість відкинутих пакетів.

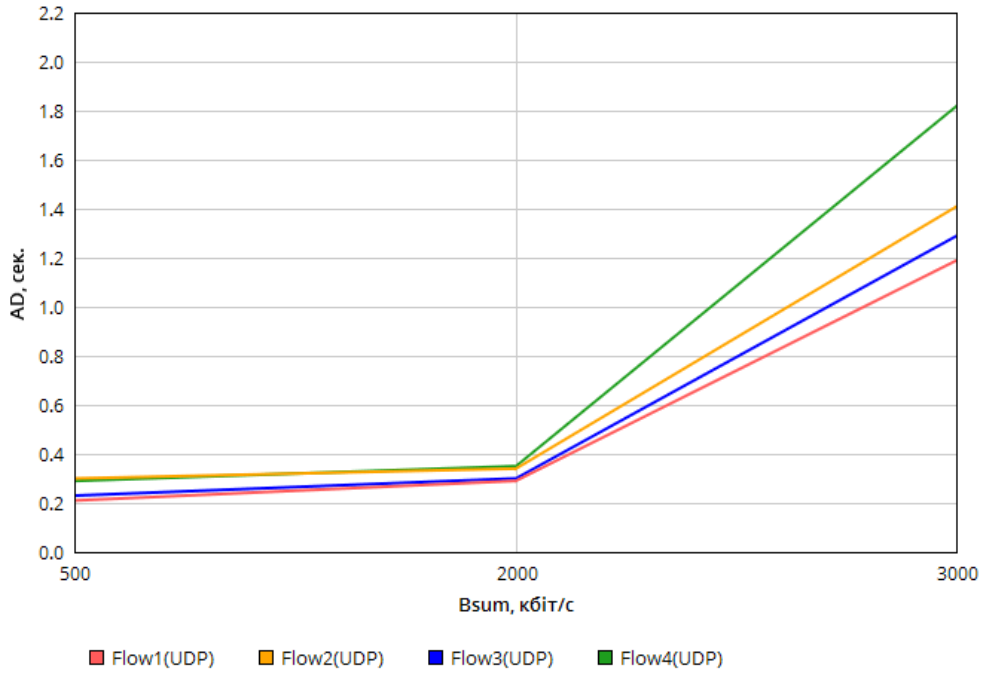

Рисунок 3.6 – Графік залежності середньої затримки від сумарної швидкості потоків

AD (average delay) – середня затримка.

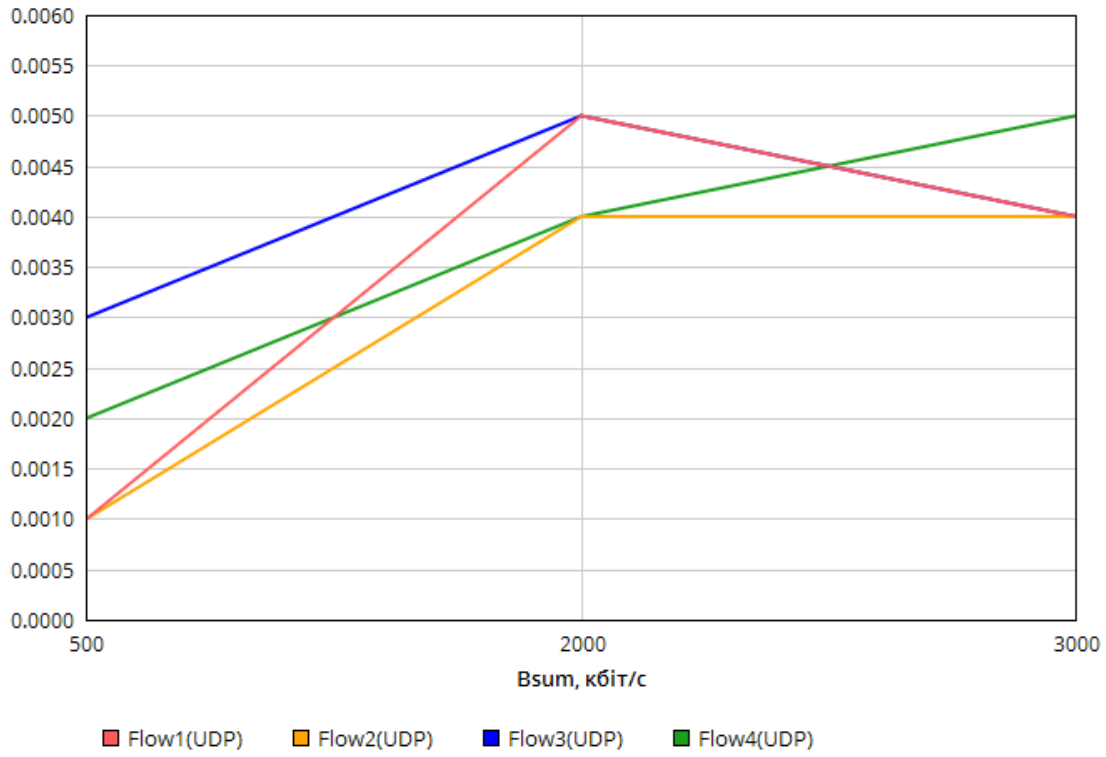

Рисунок 3.7 – Графік залежності середнього значення джитеру від сумарної швидкості потоків

AJ (average jitter) – середня варіація затримки (джитер).

Втрати пакетів при $B_{sum} \approx B_{\max}$ пояснюються тим, що на кінцевих вузлах швидкість обробки пакетів обмежена і тому деякі пакети відкидаються.

На другому етапі експериментального дослідження в досліджуваній мережі на маршрутизаторі R14(s0/0/0) застосований механізм управління інтенсивністю трафіка Traffic Shaping для пріоретизації потоків Flow1 та Flow2. Створено чотири класи (Flow1-Flow4) для кожного з згенерованих потоків трафіку відповідно. Кожному класу виділено інтенсивність трафіка в залежності від типу трафіка – 1000, 600, 300 та 100 відповідно. Класи трафіка визначалися за допомогою списків контролю доступу (ACLs) на основі таких критеріїв як протокол передачі даних та номер порта отримувача. Налаштування списків контролю доступу приведено нижче:

```
R14(config)# ip access-list extended FLOW1_ACL
R14(config-ext-nacl)# permit udp 192.168.1.0 0.0.0.255 any eq 1500
R14(config)# ip access-list extended FLOW2_ACL
R14(config-ext-nacl)# permit udp 192.168.1.0 0.0.0.255 any eq 2000
R14(config)# ip access-list extended FLOW3_ACL
R14(config-ext-nacl)# permit udp 192.168.1.0 0.0.0.255 any eq 2500
```

```
R14(config)# ip access-list extended FLOW4_ACL
R14(config-ext-nacl)# permit udp 192.168.1.0 0.0.0.255 any eq 3000
```

Пакетам, які відповідають критеріям відбору, присвоюється значення заданого класу трафіка.

Розподілення пакетів по класам:

```
R14(config)# class-map match-all CMAP_FLOW1
R14(config-cmap)# match access-group name FLOW1_ACL
R14(config)# class-map match-all CMAP_FLOW2
R14(config-cmap)# match access-group name FLOW2_ACL
R14(config)# class-map match-all CMAP_FLOW3
R14(config-cmap)# match access-group name FLOW3_ACL
R14(config)# class-map match-all CMAP_FLOW4
R14(config-cmap)# match access-group name FLOW4_ACL
```

Описання правил для кожного класу:

```
R14(config)# policy-map SHAPE_PMAP
R14(config-pmap)# class CMAP_FLOW1
R14(config-pmap-c)# shape average 1000000
R14(config-pmap)# class CMAP_FLOW2
R14(config-pmap-c)# shape average 600000
R14(config-pmap)# class CMAP_FLOW3
R14(config-pmap-c)# shape average 300000
R14(config-pmap)# class CMAP_FLOW4
R14(config-pmap-c)# shape average 100000
```

Запуск заданої політики на інтерфейсі:

```
R14(config)#int s0/0/0
R14(config-if)#service-policy output SHAPE_PMAP
```

Після закінчення налаштувань політик якості обслуговування був повторно згенерований трафік з різними номерами портів відправника та отримувача для кожного потоку трафіку (рис.3.3). Після проведення експерименту для кожної з умов (3.1) – (3.3) були отримані результати мережних характеристик по кожному потоку трафіка (таблиця 3.3).

B_{sum}, Кбіт/с	Flow1	Flow2	Flow3	Flow4
	B_1, Кбіт/с	B_2, Кбіт/с	B_3, Кбіт/с	B_4, Кбіт/с
500	125,2	125,2	125,2	125,2
2000	500	500	475	381
3000	749,8	571	286	95,7
а) Average Bitrate				

B_{sum}, Кбіт/с	Flow1	Flow2	Flow3	Flow4
	PD, пак.	PD, пак.	PD, пак.	PD, пак.
500	0	0	0	0
2000	0	0	33	412
3000	0	652	1791	2231
б) Packet dropped				

Bsum, Кбіт/с	Flow1 AD, сек.	Flow2 AD, сек.	Flow3 AD, сек.	Flow4 AD, сек.
500	0,2	0,22	0,3	0,29
2000	0,29	0,3	0,34	0,35
3000	2,9	3,27	8,7	10,1

в) Average delay

Bsum, Кбіт/с	Flow1 AJ, сек.	Flow2 AJ, сек.	Flow3 AJ, сек.	Flow4 AJ, сек.
500	0,002	0,001	0,002	0,001
2000	0,005	0,005	0,007	0,006
3000	0,004	0,008	0,006	0,014

г) Average jitter

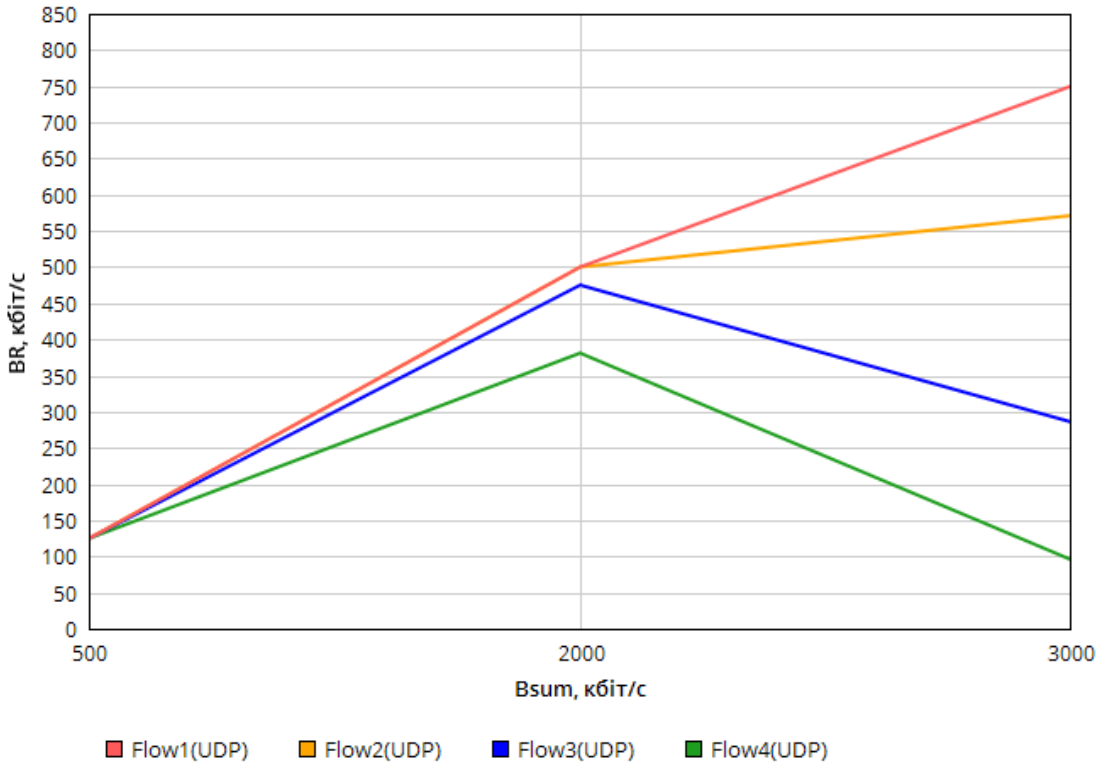

Рисунок 3.8 – Графік залежності швидкості передачі даних від сумарної швидкості потоків

BR (bitrate) – швидкість передачі даних.

Bsum – сумарна швидкість 4-х потоків.

Bmax = 2000 кбіт/с – максимальна пропускна здатність на інтерфейсі.

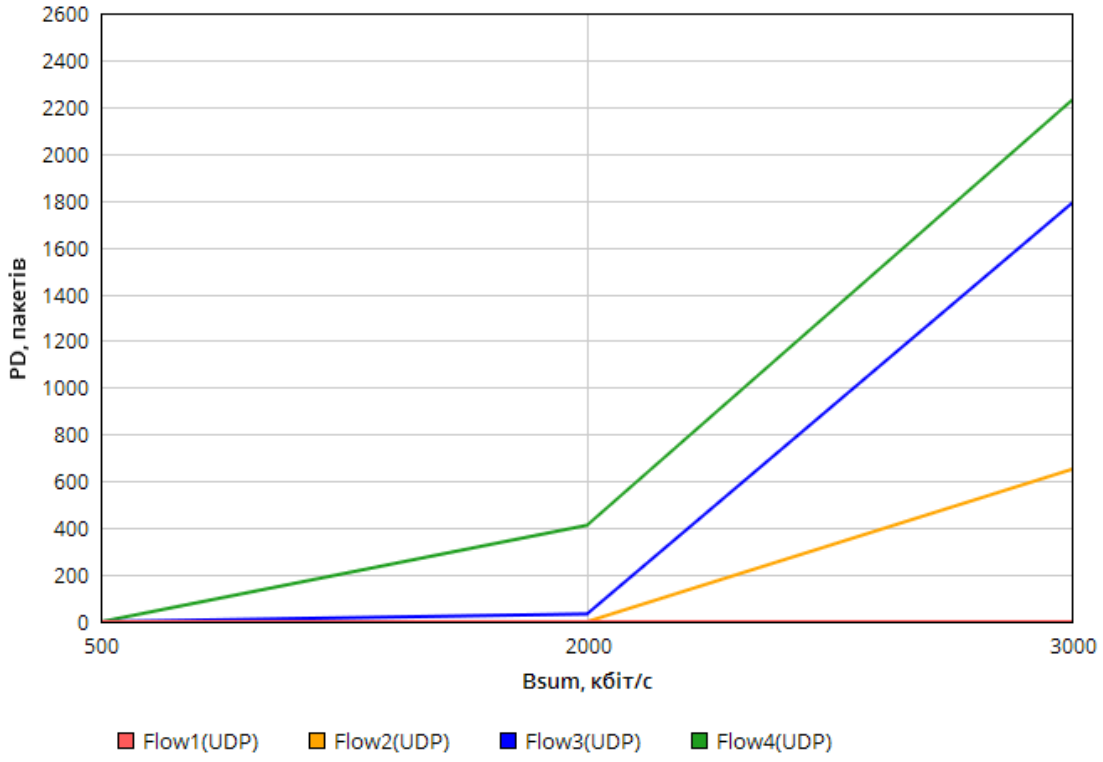

Рисунок 3.9 – Графік залежності відкинутих пакетів від сумарної швидкості потоків.

PD (packet dropped) – кількість відкинутих пакетів.

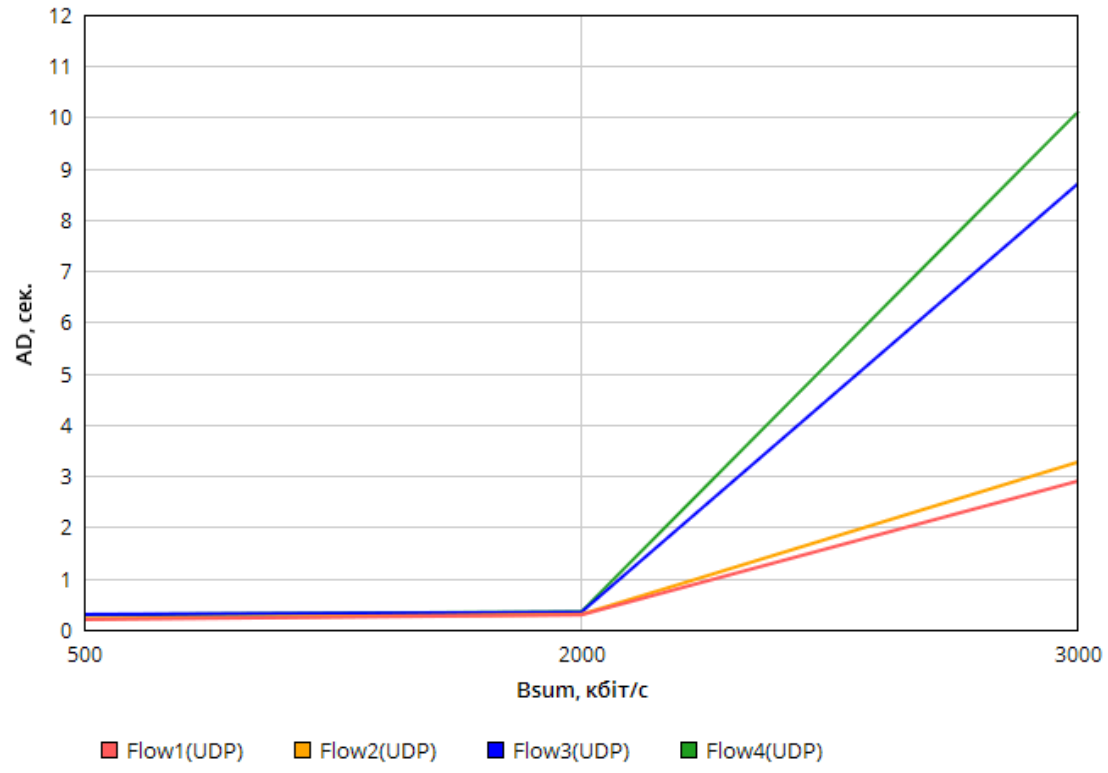

Рисунок 3.10 – Графік залежності середньої затримки від сумарної швидкості потоків

AD (average delay) – середня затримка.

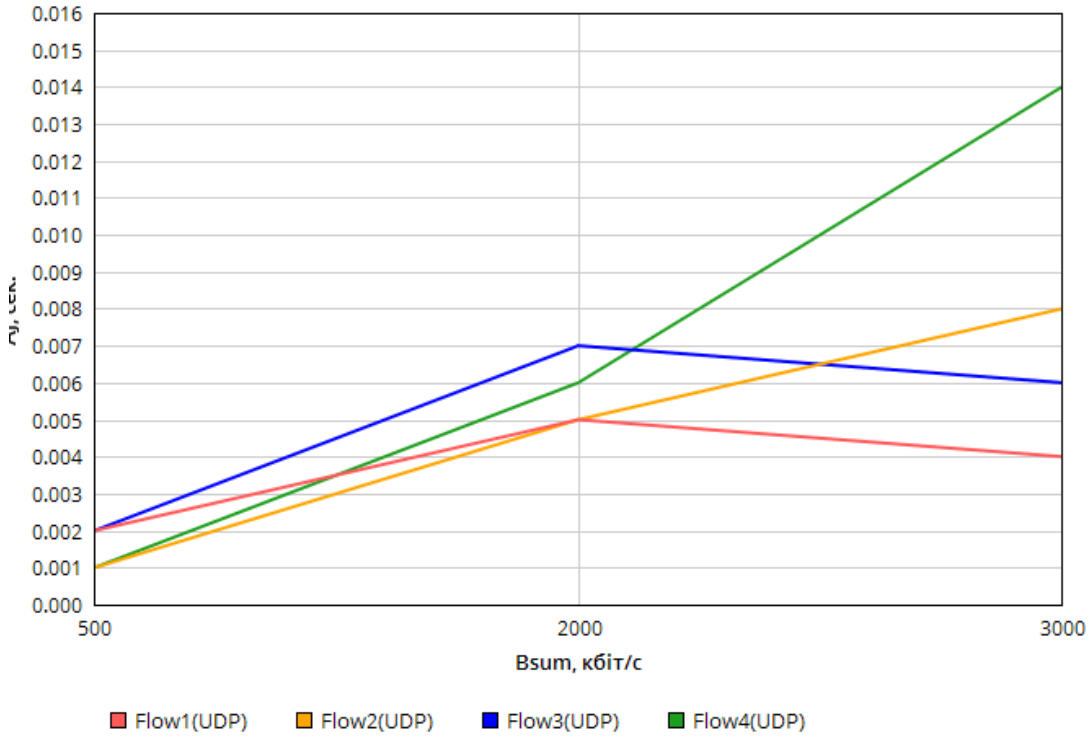

Рисунок 3.11 – Графік залежності середнього значення джитеру від сумарної швидкості потоків

AJ (average jitter) – середня варіація затримки (джитер).

Отримавши результати, бачимо, що перші Flow1 та Flow2 майже не обмежуються і не мають втрат при $B_{sum} \approx B_{\max}$, а у Flow1 не було відкинуто жодного пакету навіть при $B_{sum} > B_{\max}$.

Для подальшої розробки можливе дослідження роботи таких механізмів якості обслуговування, як CQ і WRED, описаних раніше, у комплексі з використаним механізмом управління інтенсивністю трафіка Traffic Shaping.

ВИСНОВКИ

На першому етапі експериментального дослідження згенерований трафік передавався через досліджувану мережу без використання будь-яких політик якості обслуговування, на всіх інтерфейсах топології використовувався механізм FIFO. Результати експерименту показали, що коли значення B_{sum} не перевищували максимальну пропускну здатність каналу $B_{\max,}$ то смуга пропускання порівну розподілялася між 4 потоками трафіку, втрат і затримок пакетів не спостерігалося. У випадку коли B_{sum} перевищувало $B_{\max}$ UDP-потоки ділили смугу пропускання між собою порівну.

На другому етапі експеримент проводився з використанням механізму управління інтенсивністю трафіка Traffic Shaping застосованого на маршрутизаторі R14(s0/0/0) досліджуваної топології. Створено класи для кожного потоку трафіка. Класи трафіка визначалися за допомогою списків контролю доступу (ACLs) на основі таких критеріїв як протокол передачі даних та номер порта отримувача. Результати показали, що при виникненні перевантаження ($B_{sum} > B_{\max}$) в кожному потоку трафіка, в залежності від класу до якого він відносився, була своя інтенсивність.

Загалом для коректного управління трафіком в мультисервісних мережах потрібний індивідуальний підхід, оскільки існує велика кількість факторів які так чи інакше впливають на процес управління трафіком (топологія мережі, пропускна здатність, вид обладнання та ін.). Однак все таки однією з основних особливостей мультисервісних мереж є різнорідність трафіку, тобто знаючи вимоги кожного типу трафіка до характеристик мережі можливо розробити комплексний підхід до управління трафіком, який буде універсальним. В даній дипломній роботі розроблено такий підхід, і для цього використано такі засоби управління інтенсивністю трафіка як Traffic Shaping.

Даний підхід дозволяє забезпечити коректне обслуговування для кожного типу трафіка, шляхом налаштування політик обслуговування для кожного класу, в залежності від вимог будь-якого типу трафіку до мережних характеристик.

ПЕРЕЛІК ПОСИЛАНЬ

1. Е.А. Кучерявый. Управление трафиком и качество обслуживания в сети Интернет//СПб, Наука и Техника. 2004, 121,126 с. , 190-208с.
2. Кульгин М. Технология корпоративных сетей. Часть 1. СПб: Питер, 2000.
3. Cisco Systems Quality of Service // handbook.
4. Холл Э. Приоритезация трафика в сетях IP // Сети и системы связи. 1988. №11 (33).
5. Гольдштейн Б.С., Соколов Н.А., Яновский Г.Г. Сети связи: Учебник для ВУЗов СПб.: БХВ-Петербург,2010.
6. Р. Кох, ГГ. Яновский. Эволюция и конвергенция в электросвязи//М., Радио и связь.2001.
7. Берлин А. Н. Телекоммуникационные сети и устройства: учебное пособие./ Берлин А. Н. – М. : Бином, 2008. – 319 с.
8. Олифер В. Г. Компьютерные сети. Принципы, технологии, протоколы : учебник для вузов. [4-е изд.] / В. Г. Олифер, Н. А. Олифер – СПб. : Питер, 2010. – 944 с.
9. Рональд Бодчер. Программа сетевой академии Cisco CCNA 3 и 4. [3-е изд.]: [пер. с англ.] / Рональд Бодчер, К. Р. Киркендаль. – М. : изд. Дом “Вильямс”, 2007. – 943с.
10. Коньков К. А. Основы операционных систем : курс лекций / К. А. Коньков, К. А. Карпов; ред. Петровичева Е. – М. : ИНТУИТ.РУ, 2009. – 536 с.
11. МСЭ-Т Recommendation Y.1540. IP Packet Transfer and Availability Performance Parameters//December 2002.
12. МСЭ-Т Recommendation Y.1541. Network Performance Objectives for IP-Based Services//May 2002.
13. RFC 1633, Integrated Services in the Internet Architecture: an Overview, JUNE 1994.
14. L. Zhang, R. Braden. Resource reservation Protocol//RFC-2205, September 1997.
15. RFC 2474- Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers,1998
16. S. Blake et al. Architecture for Differentiated Services//RFC-2475, December 1998.
17. RFC 2597 (Assured Forwarding PHB Group), June 1999.
18. RFC 2598 (An Expedited Forwarding PHB), June 1999.
19. RFC 2597 (Assured Forwarding PHB Group), June 1999.
20. Яновский, Г. Г. Качество обслуживания в сетях IP / Г. Г. Яновский // Журнал Вестник связи, №1, 2008.

21. Class-Based Weighted Fair Queuing [Електронний ресур] / Cisco IOS Software Releases 12.0. – Режим доступа: http://www.cisco.com/en/US/docs/ios/12_0t/12_0t5/feature/guide/cbwfq.html