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

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «СИСТЕМА БАГАТОАДРЕСНОГО ОБМІНУ ІНФОРМАЦІЄЮ В КОМП'ЮТЕРНИХ МЕРЕЖАХ»

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

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

_____ *Олександр ЙОСИПЕНКО*
(підпис) *Ім'я, ПРІЗВИЩЕ здобувача*

Виконав: здобувач вищої освіти гр. КІД-42
Олександр ЙОСИПЕНКО
Ім'я, ПРІЗВИЩЕ

Керівник: Ірина КУЗЬМІЧ
науковий ступінь, вчене звання Ім'я, ПРІЗВИЩЕ

Рецензент: _____
науковий ступінь, вчене звання Ім'я, ПРІЗВИЩЕ

Київ 2026

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
Навчально-науковий інститут Інформаційних технологій

Кафедра 123 Комп'ютерної інженерії
Ступінь вищої освіти Бакалавр
Спеціальність 123 Комп'ютерної інженерії
Освітньо-професійна програма Комп'ютерна інженерія

ЗАТВЕРДЖУЮ
Завідувач кафедрою Наталія ЛАЩЕВСЬКА
Ім'я, ПРІЗВИЩЕ

«_____» _____ 2026р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Йосипенко Олександр Сергійович
(прізвище, ім'я, по батькові здобувача)

1. Тема кваліфікаційної роботи: Система багатоадресного обміну інформацією в комп'ютерних мережах
керівник кваліфікаційної роботи Ірина КУЗЬМІЧ,
(Ім'я, ПРІЗВИЩЕ, науковий ступінь, вчене звання)
затверджені наказом Державного університету інформаційно-комунікаційних технологій від «20» 02 2026р. № 51
2. Строк подання кваліфікаційної роботи «08» 05 2026р.
3. Вихідні дані до кваліфікаційної роботи: Науково-технічна література, методи системного аналізу, мережеві протоколи передачі даних, математичний апарат, емпіричні методи
Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Дослідження специфіки та сучасного стану предметної області
2. Теоретико-методологічні основи побудови систем багатоадресного обміну даними
3. Практична реалізація та дослідження
4. Перелік ілюстративного матеріалу: *презентація*
5. Дата видачі завдання «09» 02 2026р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Збір даних	25.02-24.03.2026	Виконано
2	Обробка матеріалу	23.03-05.04.2026	Виконано
3	Розробка теоретичної частини	06.04-10.04.2026	Виконано
4	Розробка практичної частини	11.04-20.04.2026	Виконано
5	Висновки за результатами аналізу	21.04-06.05.2026	Виконано
6	Оформлення пояснювальної записки	07.05-08.05.2026	Виконано
7	Розробка презентації	09.05-11.05.2026	Виконано
8	Передзахист	27.05-29.05.2026	Виконано
9	Здача в деканат	02.06-05.06.2026	Виконано

Здобувач вищої освіти _____ Олександр ЙОСИПЕНКО
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

Керівник кваліфікаційної роботи _____ Ірина КУЗЬМІЧ
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня бакалавра: 61 стор., 19 рис., 1 табл., 22 джерела.

Мета дослідження – розробка та дослідження адаптивного транспортного протоколу багатоадресного зв'язку для забезпечення гарантованої та послідовної доставки великих масивів даних у комп'ютерних мережах..

Об'єкт дослідження – процес керування потоками даних у комп'ютерних мережах під час багатоадресної передачі.

Предмет дослідження – методи, архітектури та протоколи надійної багатоадресної передачі даних.

Короткий зміст роботи: Спектр вимог до сучасних мережевих додатків, що використовують технології надійної багатоадресної передачі (Reliable Multicast, RM), є надзвичайно широким. Існуючі протоколи мають певні обмеження: механізм на основі позитивних підтверджень є неефективним через проблему перевантаження джерела, а такі протоколи як PGM та M/TCP вимагають специфічної апаратної підтримки або погано масштабуються.

Для вирішення цих проблем запропоновано Адаптивний транспортний протокол багатоадресного зв'язку. В основі його роботи лежить ієрархічна деревоподібна модель, де отримувачі поділяються на локальні сегменти з виділеними приймачами (координаторами). Зворотний зв'язок організовано таким чином, що лише координатори контактують з відправником, а локальні вузли взаємодіють зі своїм координатором, який самостійно виконує повторну передачу втрачених даних. Практична реалізація та тестування на віртуальному стенді довели, що запропонований протокол суттєво скорочує обсяг генеруючого трафіку на відправнику порівняно з традиційним стеком TCP.

КЛЮЧОВІ СЛОВА: БАГАТОАДРЕСНА ПЕРЕДАЧА, ПРОТОКОЛ АТПБЗ, МАРШРУТИЗАЦІЯ, RELIABLE MULTICAST, NAK, ACK, PGM, КООРДИНАТОР.

ABSTRACT

Textual part of the qualification work for the bachelor's degree: 61 pages, 19 figures, 1 tables, 22 sources.

The purpose of the work is the development and research of an adaptive transport protocol for multicast communication to ensure the guaranteed and sequential delivery of large amounts of data in computer networks.

The object of research is the process of managing data flows in computer networks during multicast transmission.

The subject of research is methods, architectures, and protocols of reliable multicast transmission.Summary of the work:

The range of requirements for modern network applications using Reliable Multicast technologies is extremely wide. Existing protocols have certain limitations: the positive acknowledgment mechanism is inefficient due to the ACK implosion problem , and protocols like PGM and M/TCP require specific hardware support or scale poorly.

To solve these problems, the Adaptive Transport Protocol for Multicast Communication is proposed. Its operation is based on a hierarchical tree model, where receivers are divided into local segments with designated receivers (coordinators). Feedback is organized so that only coordinators contact the sender, and local nodes interact with their coordinator, which independently performs the retransmission of lost data. Practical implementation and testing on a virtual testbed proved that the proposed protocol significantly reduces the volume of generated traffic on the sender compared to the traditional TCP stack.

KEYWORDS: MULTICAST TRANSMISSION, ATPBZ PROTOCOL, ROUTING, RELIABLE MULTICAST, NAK, ACK, PGM, COORDINATOR.

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут Інформаційних технологій

ПОДАННЯ
ГОЛОВІ ДЕРЖАВНОЇ ЕКЗАМЕНАЦІЙНОЇ КОМІСІЇ
ЩОДО ЗАХИСТУ КВАЛІФІКАЦІЙНОЇ РОБОТИ
на здобуття освітнього ступеня бакалавра

Направляється здобувач Йосипенко О. С. до захисту кваліфікаційної роботи
(прізвище та ініціали)
за спеціальністю 123 Комп'ютерна інженерія
(код, найменування спеціальності)
освітньо-професійної програми Комп'ютерна інженерія
(назва)
на тему: «Система багатоадресного обміну інформацією в комп'ютерних мережах».
Кваліфікаційна робота і рецензія додаються.

Директор ННІ _____ Катерина НЕСТЕРЕНКО
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

Висновок керівника кваліфікаційної роботи

Здобувач Йосипенко О. С. показав відмінну теоретичну підготовку, уміння володіти та розуміння новими комп'ютерними технологіями, користуватися навчальною і науково-технічною літературою. Працюючи над завданнями, які були поставлені керівником, проявив ініціативність, сумлінність та хист до інженерної роботи.
Все це дозволяє оцінити кваліфікаційну роботу студента Йосипенка Олександра Сергійовича на оцінку «відмінно», та присвоїти йому кваліфікацію «бакалавр з комп'ютерної інженерії».

Керівник кваліфікаційної роботи _____ Ірина КУЗЬМІЧ
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

«__» _____ 2026 року

Висновок кафедри про кваліфікаційну роботу

Кваліфікаційна робота розглянута. Здобувач *Йосипенко О. С.* допускається до захисту даної роботи в Екзаменаційній комісії.

Завідувач кафедрою Комп'ютерної інженерії _____ Наталія ЛАЩЕВСЬКА
(назва) *(підпис)* *(Ім'я, ПРІЗВИЩЕ)*

ВІДГУК РЕЦЕНЗЕНТА
на кваліфікаційну бакалаврську роботу

здобувача вищої освіти Йосипенко Олександр Сергійович
(прізвище, ім'я, по батькові)
на тему: «Система багатоадресного обміну інформацією в комп'ютерних мережах»

Актуальність:

Бакалаврська кваліфікаційна робота присвячена актуальній проблемі оптимізації транспортних потоків у сучасних комп'ютерних мережах під час багатоадресної розсилки. В умовах, коли обсяги медіаконтенту, системних оновлень та розподілених баз даних стрімко зростають, розробка ефективних протоколів Reliable Multicast є критично важливою задачею для інфраструктурних проектів.

Позитивні сторони:

1. У роботі наведено глибокий теоретико-методологічний аналіз існуючих протоколів передачі даних. Автор вдало аргументував фундаментальні архітектурні обмеження цих рішень, зокрема проблему перевантаження джерела та складності масштабування.
2. Запропоновано інноваційну архітектуру Адаптивного транспортного протоколу багатоадресного зв'язку з використанням деревоподібної ієрархії та локальних координаторів. Механізм динамічного ковзного вікна та керування перевантаженнями реалізовано на високому інженерному рівні.
3. Практична цінність роботи беззаперечна. Автор створив повноцінний імітаційний стенд на базі VMWare ESXi та власноруч написав програмну реалізацію протоколу мовою Python із використанням raw sockets. Емпіричні дані переконливо доводять перевагу розробленого L2-протоколу над традиційним TCP.
4.

Недоліки:

1. У поточній програмній реалізації протоколу локальний координатор призначається вручну, і кінцеві вузли-отримувачі повинні заздалегідь знати його мережеву адресу. Для сучасних динамічних систем бажано було б глибше опрацювати механізм автоматичного обрання координатора на основі метрик якості каналу.
2. Недостатньо уваги приділено тестуванню системи в умовах асиметричних каналів зв'язку або середовищ із високим рівнем втрати пакетів, де алгоритми прямої корекції помилок могли б доповнити запропоновану NAK-архітектуру.

Висновок: *кваліфікаційна бакалаврська робота заслуговує оцінку " відмінно", а здобувач Йосипенко О. С. заслуговує присвоєння кваліфікації: бакалавра з комп'ютерної інженерії.*

Рецензент:
науковий ступінь, вчене звання _____ *підпис* _____ *Ім'я, ПРІЗВИЩЕ*

ЗМІСТ

ВСТУП.....9
1 ДОСЛІДЖЕННЯ СПЕЦИФІКИ ТА СУЧАСНОГО СТАНУ ПРЕДМЕТНОЇ ОБЛАСТІ.....10
1.1 Базові концепції та сучасні методи керування потоками даних у комп'ютерних мережах.....10
1.2 Аналіз сучасних технологій та протоколів трансляції даних у комп'ютерних мережах.....16
1.3 Комплексний порівняльний аналіз сучасних методів та протоколів передачі даних.....26
2 ТЕОРЕТИКО-МЕТОДОЛОГІЧНІ ОСНОВИ ПОБУДОВИ СИСТЕМ БАГАТОАДРЕСНОГО ОБМІНУ ДАНИМИ.....28
2.1 Архітектурні рішення та базові компоненти мережі.....28
2.2 Архітектура та алгоритмічна база мережевого протоколу.....31
3 ПРАКТИЧНА РЕАЛІЗАЦІЯ ТА ДОСЛІДЖЕННЯ.....42
3.1 Характеристика експериментального середовища та інфраструктури.....42
3.2 Організація та етапи експериментальної перевірки.....45
3.3 Емпіричні результати апробації та їх оцінка.....47
ВИСНОВКИ.....55
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....57
Додаток А.....60

ВСТУП

В наш час комп’ютерні мережі повинні задовольняти надзвичайно широкий спектр вимог, особливо коли йдеться про додатки, що використовують технології надійної багатоадресної передачі. У багатьох сучасних розподілених системах логічно цілісні масиви даних, такі як оновлення програмного забезпечення чи великі медіафайли, повинні синхронно доставлятися тисячам або мільйонам клієнтів.

Забезпечення повністю безпомилкової доставки є складним інженерним завданням. Класичний стандарт TCP є протоколом «точка-точка». Якщо використовувати його для масової розсилки, відправник змушений підтримувати індивідуальні сесії з кожним клієнтом, що неминуче призводить до перевантаження магістрального каналу. З іншого боку, протокол UDP підтримує багатоадресну розсилку, що дозволяє раціонально використовувати пропускну здатність, але він повністю позбавлений механізмів гарантованої доставки та контролю цілісності пакетів. Існуючі спеціалізовані протоколи, такі як PGM та M/TCP, частково вирішують ці проблеми, але мають власні недоліки: вони можуть погано масштабуватися або вимагати глибокої апаратної підтримки на рівні проміжних мережевих вузлів.

Тож постає питання розробки такого архітектурного рішення, яке б поєднувало ефективність багатоадресної розсилки з надійністю гарантованої доставки, не викликаючи при цьому так званого «шторму підтверджень» (ACK implosion) від великої кількості абонентів. Вирішенням цієї проблеми є застосування ієрархічної деревоподібної моделі з делегуванням повноважень на рівень локальних сегментів. Розробка та дослідження такого адаптивного транспортного протоколу дозволить суттєво знизити навантаження на центральні сервери та магістральні канали зв'язку.

1 ДОСЛІДЖЕННЯ СПЕЦИФІКИ ТА СУЧАСНОГО СТАНУ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Базові концепції та сучасні методи керування потоками даних у комп'ютерних мережах

Спектр вимог до сучасних мережевих додатків, що використовують технології надійної багатоадресної передачі (Reliable Multicast, RM), є надзвичайно широким та залежить від конкретної сфери застосування. Проектування ефективного протоколу на основі технології RM вимагає врахування низки критичних факторів. Зокрема, розробникам необхідно визначити, чи потребує система програмного підтвердження отримання даних абсолютно всіма вузлами-приймачами, чи передбачається масштабування додатка для роботи з тисячами або навіть мільйонами клієнтів (наприклад, у сучасних системах Інтернету речей або IPTV). Крім того, важливо розуміти, чи є критичною повністю безпомилкова доставка кожного байта інформації, та чи існують жорсткі часові рамки, порушення яких робить отримані дані неактуальними.

Фундаментальним аспектом є механізм підтвердження доставки інформації. У багатьох сучасних розподілених системах логічно цілісні масиви даних (наприклад, оновлення програмного забезпечення, бази даних або великі медіафайли) повинні синхронно доставлятися багатьом клієнтам. Для цього використовується концепція ADU (Application Data Unit) — логічно неподільного блоку даних, який має практичну цінність для конкретного додатка. Залежно від архітектури, розмір ADU може бути мінімальним і вміщуватися в один стандартний мережевий пакет, або ж значно перевищувати його обсяг, вимагаючи фрагментації [7].

Надійна багатоадресна передача гарантує цілісність інформації за допомогою системи зворотного зв'язку, яка сповіщає відправника про статус доставки. Сучасні протоколи розрізняють два основні рівні таких підтверджень: на рівні цілісного блоку ADU та на рівні окремого мережевого пакета. Підтвердження на рівні ADU дозволяє системі зрозуміти, коли трансляцію конкретного логічного блоку можна успішно завершити. Натомість покетне підтвердження є критично важливим для оптимізації ресурсів сучасного високопродуктивного обладнання — воно сигналізує про те, що певний фрагмент даних успішно досяг адресата, і оперативну пам'ять (буфер) комутатора чи сервера, виділену під його зберігання, можна звільнити [7]. Для низки критичних інфраструктурних додатків життєво необхідним є отримання стовідсоткового підтвердження доставки ADU від кожного учасника мережі, або ж точна ідентифікація вузлів, які зазнали збою під час прийому.

Масштабованість мережевої інфраструктури виступає одним із найжорсткіших обмежувальних факторів під час вибору архітектури підтверджень. Якщо алгоритм передбачає відправлення класичного підтвердження (ACK) кожним клієнтом після кожного отриманого пакета, така система зазнає краху при спробі масштабування. У мережах з великою кількістю абонентів це призводить до так званого «шторму підтверджень» (ACK implosion), коли центральний вузол або маршрутизатор не витримує колосального навантаження від обробки вхідних службових повідомлень. Для уникнення таких ситуацій у сучасних мережах застосовуються оптимізовані механізми контролю, які будуть детально розглянуті нижче.

Щодо повної надійності доставки, варто зазначити, що багато корпоративних додатків не допускають жодних втрат. У разі втрати хоча б одного сегмента всередині ADU, весь блок даних (наприклад, архів або фінансовий транзакційний лог) стає непридатним для використання. На

противагу цьому, існує великий клас сучасних мультимедійних додатків, де 100% надійність не є пріоритетом. Під час трансляції потокового відео високої чіткості або IP-телефонії втрата кількох пакетів призведе лише до незначного, часто непомітного для ока чи вуха, зниження якості, що є прийнятним компромісом заради швидкості.

Ще одним важливим фактором є обмежена за часом доставка даних. Багато сучасних сервісів функціонують у режимі реального часу і мають надзвичайно жорсткі вимоги до затримок (latency) та джитера. Якщо інформація не надходить до буфера клієнта у визначене мілісекундне вікно, її подальша доставка втрачає будь-який сенс і лише марно завантажує канал [7]. Така специфіка яскраво проявляється під час організації стрімів, відеоконференцій або інтерактивного мовлення, рис. 1.1. У подібних архітектурах критично важливо, щоб програмне забезпечення мало інтелектуальні алгоритми пріоритезації трафіку та контролювало часові мітки відправлення.



Рисунок 1.1 — Специфіка використання протоколу TCP порівняно з UDP під час трансляції потокових мультимедійних пакетів даних

На архітектуру протоколу також фундаментально впливають фізичні та логічні властивості самої мережі. Наприклад, наявність односпрямованих каналів. У системах супутникового зв'язку (навіть у сучасних стандартах DVB-S2) або в мережах ефірного цифрового телебачення зворотний канал зв'язку (uplink) може бути суттєво обмеженим у пропускній здатності, мати величезні затримки або взагалі бути відсутнім. У таких асиметричних умовах класичні механізми запиту втрачених пакетів не працюють. Єдиним ефективним інженерним рішенням залишається попереднє кодування даних із застосуванням алгоритмів прямої корекції помилок (FEC).

Ефективність багатоадресної розсилки також залежить від рівня підтримки з боку активного мережевого обладнання. Сучасна концепція надійної багатоадресної передачі охоплює як програмні алгоритми на кінцевих хостах, так і апаратне прискорення на рівні L3-комутаторів та маршрутизаторів. Глобально підходи до розподілу ролей у мережі можна поділити на чотири категорії [8].

Перша категорія — це відсутність спеціалізованої підтримки, коли транзитне обладнання просто пересилає трафік, а вся інтелектуальна робота з контролю доставки лягає на кінцеві вузли (як у класичних overlay-мережах).

Другий підхід — багаторівневий, який активно використовує можливості сучасних протоколів керування групами, таких як IGMPv3 або MLD для IPv6. Дані розподіляються між різними мультикаст-групами, а клієнти динамічно підключаються лише до потрібних потоків, що вимагає апаратної підтримки фільтрації на рівні комутаторів доступу.

Третій підхід — серверний, що передбачає інтеграцію проміжних edge-серверів або вузлів CDN (Content Delivery Network), які беруть на себе локальне кешування, агрегацію статистики та ретрансляцію втрачених сегментів, розвантажуючи магістральні канали.

Четвертий підхід спирається на глибоку інтеграцію з маршрутизаторами та технологіями програмно-конфігурованих мереж (SDN). У цьому сценарії інтелектуальне мережеве обладнання здатне самостійно оптимізувати дерева

маршрутизації, керувати пріоритетами та агрегувати службові повідомлення. Сучасні високопродуктивні маршрутизатори мають достатньо оперативної пам'яті та потужні ASIC-процесори, щоб ефективно реалізовувати такі складні функції керування потоками.

Під час розгортання систем багатоадресного обміну ключовими викликами стають забезпечення стабільно високої пропускної здатності та превентивний контроль перевантажень у каналах зв'язку. Втрата пакетів, спричинена переповненням буферів на портах комутаторів, є головним індикатором деградації мережі. Тому алгоритми швидкої реєстрації втрат та адекватного реагування на них лежать в основі стабільної роботи системи. Для контролю цілісності доставки застосовуються різні концептуальні методи [7, 9].

Класичний механізм на основі позитивних підтверджень (ACK-based) передбачає відправлення квитанції про успішний прийом кожного пакета. Попри свою алгоритмічну простоту, у сучасних умовах він є неефективним через згадану раніше проблему перевантаження джерела (ACK implosion), що робить його придатним лише для невеликих ізольованих груп з кількома клієнтами.

Більш прогресивним є деревоподібний механізм підтверджень (Tree-based ACK). Він спирається на логічну топологію зв'язного ациклічного графа. Клієнтські вузли не звертаються безпосередньо до головного сервера, а відправляють свої підтвердження локальним агрегаційним вузлам (батьківським вершинам). Ті, у свою чергу, збирають статистику зі свого сегмента мережі, формують єдиний зведений звіт і передають його вище по ієрархії. Інформаційне корисне навантаження при цьому транслюється звичайним шляхом. Цей метод демонструє відмінну масштабованість та стійкість до збоїв, проте вимагає наявності ієрархічної мережевої інфраструктури.

Альтернативним та вельми популярним підходом є використання негативних підтверджень (NAK-based). Замість того, щоб звітувати про кожен

успішно отриманий байт, клієнти мовчать доти, доки не виявлять пропуск у нумерації пакетів. Лише тоді вони генерують запит NAK із вимогою повторно надіслати конкретний втрачений фрагмент. Такий підхід має суттєві переваги [9]. По-перше, головному серверу необов'язково знати точну кількість підключених абонентів, що ідеально підходить для відкритого IPTV-мовлення та позбавляє необхідності постійно перераховувати топологію. По-друге, система стає значно стійкішою до відмов, адже достатньо отримати сигнал NAK хоча б від одного вузла, щоб відправник зрозумів проблему та ініціював ретрансляцію для всього проблемного сегмента. Головний недолік цієї архітектури полягає у складності керування пам'яттю сервера: системі важко визначити точний момент, коли всі клієнти гарантовано отримали дані, і буфер передачі можна безпечно очистити.

Для умов, де зворотний зв'язок неможливий або неефективний, використовується пряма корекція помилок (FEC — Forward Error Correction). Це метод превентивного захисту інформації від пошкоджень під час транспортування. На рівні пакетів базова логіка FEC працює за принципом додавання надлишковості за допомогою математичних операцій логічного виключення (XOR). Замість того, щоб чекати запитів на повторну передачу, алгоритм групує вихідні пакети даних і обчислює для них спеціальний додатковий пакет парності. Уся ця група відправляється в мережу разом. Якщо під час передачі клієнт втрачає один з оригінальних пакетів корисного навантаження, але успішно приймає всі інші пакети групи та згенерований пакет парності, його локальне програмне забезпечення здатне повністю відновити втрачену інформацію шляхом виконання зворотних логічних операцій. Це дозволяє гарантувати цілісність даних взагалі без необхідності повторного звернення до сервера.

1.2 Аналіз сучасних технологій та протоколів трансляції даних у комп'ютерних мережах

Сучасні підходи до багатоадресної передачі даних еволюціонували від базових концепцій до складних архітектурних рішень, здатних адаптуватися до різноманітних умов мережевого середовища та вимог конкретних додатків. Детальний аналіз ключових протоколів дозволяє зрозуміти специфіку їхньої роботи та сфери найбільш ефективного застосування.

PGM є спеціалізованим протоколом транспортного рівня, оптимізованим для роботи поверх IP-multicast. Його головна парадигма полягає в гарантуванні того, що кожен клієнт-учасник мультикаст-групи або успішно отримає абсолютно всі надіслані пакети даних без винятку, або ж система своєчасно зафіксує та просигналізує про факт невідновної втрати інформації. Цей протокол класифікується як експериментальний [11], проте його архітектурні принципи лягли в основу багатьох сучасних систем розсилки.

Високий ступінь масштабованості PGM забезпечується завдяки побудові чіткої ієрархічної структури та інтеграції трьох потужних механізмів: прямої корекції помилок (FEC), використання негативних підтверджень (NAK) та розумного алгоритму придушення надлишкових запитів NAK (NAK suppression). Максимальна продуктивність протоколу досягається в мережах, де всі проміжні вузли (зокрема, маршрутизатори та L3-комутатори) апаратно підтримують його логіку. Тим не менш, інженери передбачили здатність PGM функціонувати (хоча і з певним зниженням загальної ефективності) у гетерогенних середовищах, де частина транзитного обладнання не розпізнає PGM-пакети і обробляє їх як стандартний IP-трафік [12].

Особливістю PGM є висока гнучкість управління сесіями: клієнти мають змогу динамічно приєднуватися до трансляції та залишати її в будь-який момент часу. При цьому протокол гарантує надійність доставки виключно в межах так званого поточного «вікна передачі» (transmit window) —

обмеженого буфера актуальних даних. Через цю специфіку PGM ідеально підходить для архітектур додатків, які здатні самостійно компенсувати втрати даних на вищому (прикладному) рівні еталонної моделі OSI, якщо протокол транспортного рівня повідомляє про неможливість відновлення сегмента. Отже, PGM є пріоритетним вибором для систем реального часу та трансакційних сервісів.

Типовими сценаріями використання PGM у сучасних умовах є системи масового розповсюдження фінансової інформації (наприклад, котирувань акцій на фондових біржах) та технології масового розгортання операційних систем (клонування образів дисків по мережі). У випадку з біржовими котируваннями, точність даних є критичною. Проте, якщо інформація про зміну ціни не була доставлена вчасно, а за частки секунди котирування вже оновилося знову, ініціювати повторну передачу застарілого значення немає жодного сенсу — система просто відкидає цю операцію. У сценарії клонування дисків архітектура PGM дозволяє серверу не зупиняти трансляцію основного масиву даних через кілька «повільних» клієнтів з проблемними каналами зв'язку. Такі вузли згодом зможуть дозавантажити відсутні блоки інформації, використовуючи класичну одноадресну клієнт-серверну модель взаємодії (наприклад, по HTTP чи FTP).

Логічна топологія (дерево) PGM формується на базі існуючої інфраструктури IP-multicast, спираючись на мережеві вузли, що мають вбудовану підтримку цього протоколу. Процес обміну даними та відновлення втрат виглядає наступним чином.

Сервер-джерело генерує і відправляє в мультикаст-групу послідовно пронумеровані блоки оригінальних даних (пакети ODATA — Original Data). Клієнтські пристрої (одержувачі) безперервно аналізують вхідний потік. Якщо клієнт виявляє розрив у нумерації пакетів (що свідчить про втрату), він негайно формує запит негативного підтвердження (NAK) і надсилає його вгору по ієрархії до свого безпосереднього «батьківського» вузла (зазвичай це найближчий маршрутизатор доступу).

Отримавши NAK, батьківський маршрутизатор зобов'язаний підтвердити його прийом. Для цього він розсилає вниз по своєму сегменту дерева спеціальний службовий пакет підтвердження (NCF — NAK Confirmation). У відповідь на оригінальний запит NAK система генерує пакет відновлення (RDATA — Repair Data). Важливо зазначити, що RDATA може бути згенерований не обов'язково головним сервером. Залежно від конфігурації мережі, цю роль може взяти на себе спеціально виділений локальний вузол відновлення (DLR — Designated Local Router), який має в кеші копію втрачених даних, що суттєво розвантажує магістральні канали. Формат пакета RDATA залежить від налаштувань сесії: це може бути як пряма копія втраченого пакета ODATA, так і математично обчислений пакет корекції помилок (FEC).

Щоб уникнути перевантаження мережі масовими запитами NAK від сотень клієнтів, які одночасно не отримали той самий пакет (так званий «NAK-шторм»), розробники впровадили механізм придушення. Перед відправкою NAK кожен клієнтський пристрій запускає випадковий таймер затримки. Якщо до завершення відліку цього таймера клієнт отримує від батьківського вузла пакет NCF (який свідчить про те, що хтось інший у цьому сегменті вже запитав втрачений пакет), він скасовує власну відправку NAK. Ця елегантна логіка гарантує, що система згенерує і відправить на відновлення лише один єдиний запит для обслуговування цілої групи одержувачів, які зазнали ідентичної втрати [11]. Детальну візуалізацію цього алгоритму наведено на рис. 1.2.

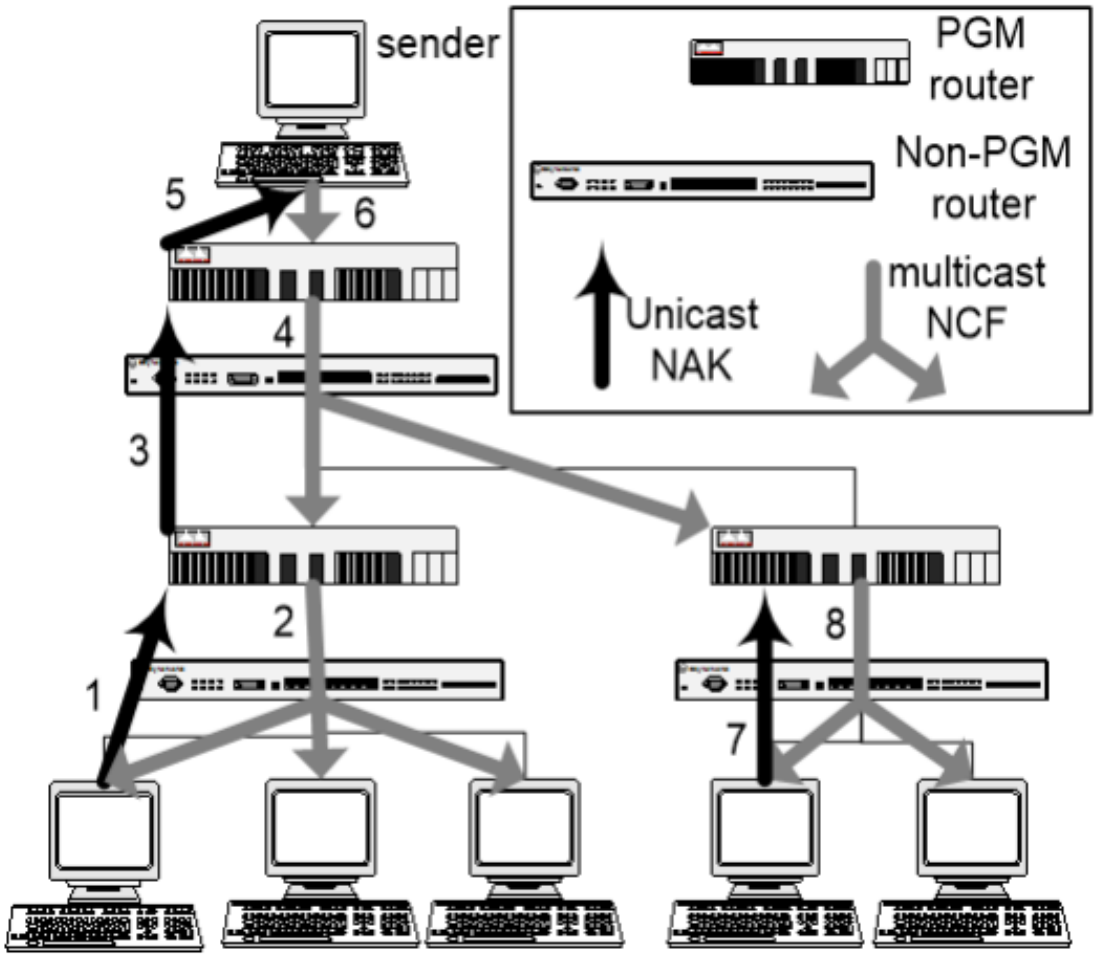


Рисунок 1.2 — Деталізований сценарій взаємодії службових повідомлень NAK та NCF під час процедури відновлення втрачених даних

Пояснення до алгоритму взаємодії (рис. 1.2): Спочатку одержувачі фіксують відсутність пакета і генерують запити NAK. Батьківський маршрутизатор, отримавши перший з них, одразу розсилає пакет підтвердження NCF у свій локальний сегмент. Далі цей маршрутизатор ретранслює запит NAK вище по ієрархії до свого батьківського вузла, який, своєю чергою, також підтверджує його пакетом NCF. Цей процес каскадно триває, доки запит NAK не досягне головного сервера (відправника), який фіналізує ланцюжок розсилкою свого NCF. Якщо через певний час інший, більш «повільний» одержувач у тій самій гілці виявляє відсутність того ж самого пакета, він генерує власний NAK. Однак, його локальний маршрутизатор, вже знаючи про проблему і маючи підтвердження про те, що

запит на відновлення вже відправлено вище, просто заспокоює клієнта локальним пакетом NCF, блокуючи подальшу трансляцію запиту NAK вгору по дереву мережі [11].

Протокол PGM бере на себе завдання побудови та безперервної підтримки в актуальному стані точного дерева маршрутизації — шляхів від джерела розсилки до кожного кінцевого підписника групи. Кожен транзитний вузол у цій структурі чітко знає напрямок (інтерфейс), з якого до нього надходить мультикаст-трафік. Завдяки цьому, коли кінцевий пристрій ініціює запит на відновлення даних, цей запит NAK здатен безпомилково маршрутизуватися у зворотному напрямку — від «листя» дерева до його «кореня».

Проходячи через проміжні вузли, запит NAK залишає за собою «слід» — змінює локальний стан маршрутизатора, позначаючи, що саме в цій конкретній підмережі клієнти очікують на відновлення певного пакета. Коли сервер врешті-решт надсилає пакет RDATA, цей пакет не розсилається сліпо на всі інтерфейси. Маршрутизатори інтелектуально пересилають відновлені дані виключно в ті гілки дерева, де попередньо було виставлено відповідний стан очікування. Гілки, які не сигналізували про втрати, ігнорують цей трафік, економлячи пропускну здатність.

Для підтримки такої складної топології джерело розсилки періодично, через задані інтервали часу, генерує спеціальні службові повідомлення (path messages), які оновлюють таблиці маршрутизації та інформацію про стан дерева на всіх проміжних вузлах [11]. В основі цієї технології лежать концепції ініційованої відправником багатоадресної передачі (SIM) та явної багатоадресної розсилки (Xcast) [9, 10].

Протокол M/TCP пропонує альтернативний підхід, розроблений переважно для додатків, де потік даних жорстко ініціюється та керується центральним сервером-відправником. Типовими прикладами є системи масової передачі файлів (на базі логіки FTP) або розсилки повідомлень (аналоги SMTP). Головною перевагою M/TCP є простота впровадження:

архітектура вимагає інсталяції спеціалізованого програмного забезпечення або модифікації стека протоколів виключно на боці сервера-відправника. Клієнтські пристрої (одержувачі) продовжують працювати зі стандартними мережевими налаштуваннями, оскільки вони сприймають вхідний трафік не як складний мультикаст, а як звичайні одноадресні (unicast) TCP-з'єднання. Це повністю знімає потребу в оновленні клієнтського програмного забезпечення чи операційних систем.

По суті, M/TCP створює міст між потребою в надійній багатоадресній розсилці та бажанням використовувати величезний парк існуючих додатків, написаних під класичний стандарт TCP. Схематична модель обміну даними в середовищі M/TCP наведена на рис. 1.3.

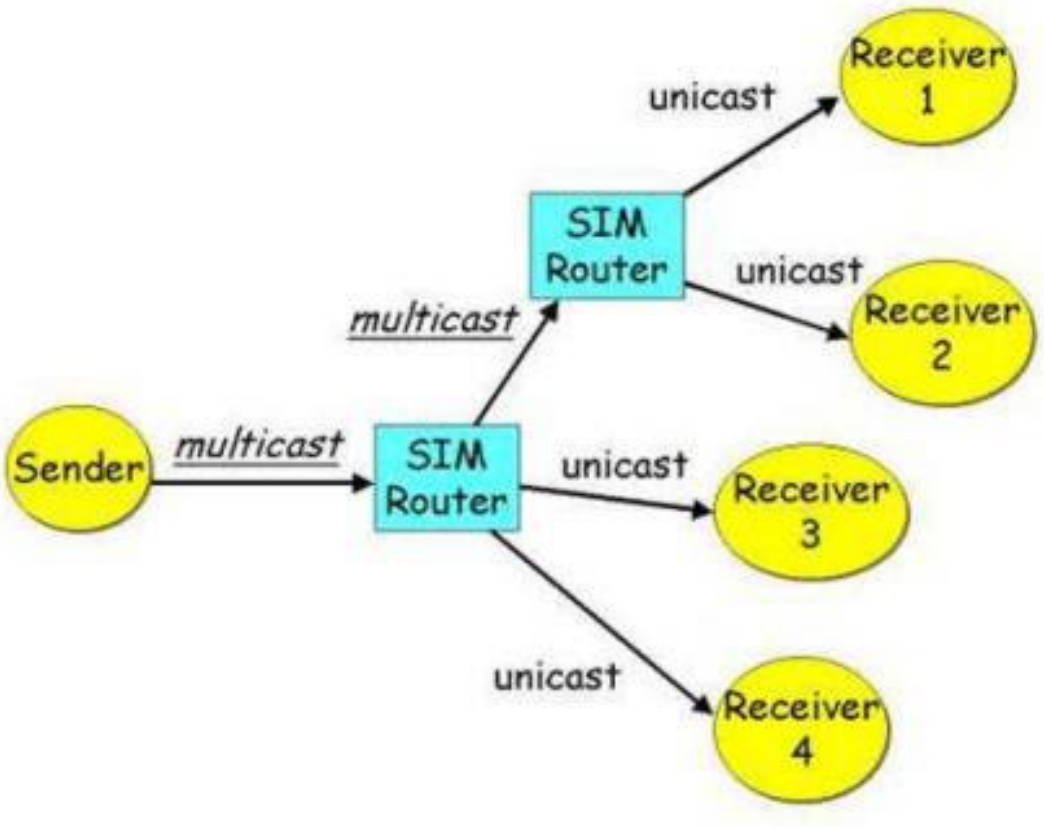


Рисунок 1.3 — Схема організації транспортних потоків та передачі даних в архітектурі M/TCP

Попри свої переваги, M/TCP має ряд суттєвих архітектурних обмежень. Цей протокол глибоко залежить від наявності спеціальної підтримки з боку інфраструктури мережі, він погано масштабується (розрахований на одночасну роботу лише з відносно невеликою кількістю підписників) і підтримує вузький спектр специфічних додатків [10].

Розробники позиціонують M/TCP як рішення для систем, які відповідають трьом ключовим критеріям [10]:

Абсолютна надійність. Оскільки M/TCP успадковує фундаментальні механізми контролю оригінального протоколу TCP, він гарантує безпомилкову доставку кожного байта інформації до кінцевого адресата.

Асиметричний характер трафіку. Протокол оптимізовано для моделі взаємодії «один-до-багатьох», де обсяг вихідного трафіку від головного сервера на порядки перевищує обсяг службової інформації або підтверджень, що генеруються клієнтами. Ініціатива обміну завжди належить відправнику.

Відсутність жорстких вимог до реального часу (non-real-time). Вбудовані в ядро TCP алгоритми контролю перевантажень (TCP congestion control) та плаваючого вікна принципово не сумісні з трансляцією потокового мультимедіа або чутливих до затримок даних. Вони неминуче призводитимуть до критичних затримок та буферизації.

Інтелектуальна складова M/TCP проявляється у здатності аналізувати пропускну здатність та продуктивність кожного індивідуального клієнта. У разі втрати мережевого пакета певним сегментом групи одержувачів, система не призупиняє основну розсилку. Замість цього, сервер відкриває паралельний, незалежний TCP-потік спеціально для доставки втраченого фрагмента. Завдяки такому підходу швидкість основної трансляції не деградує і не орієнтується на найповільнішого учасника мережі, що є величезною проблемою класичних систем [14].

Алгоритм обробки втрат у M/TCP працює за наступною логікою [14]:

Якщо система фіксує втрату одного й того самого пакета одночасно на двох або більше клієнтських пристроях, сервер ініціює повторну доставку, використовуючи багатоадресну розсилку для оптимізації трафіку.

Якщо втрата зафіксована лише на одному єдиному вузлі, сервер все одно використовує механізми багатоадресної передачі для ретрансляції (в окремих реалізаціях може використовуватись оптимізований unicast).

Для клієнтів, які зазнали втрати та потребують відновлення, система динамічно створює окрему тимчасову підгрупу. Нові пакети даних транслюються цій підгрупі ізольовано від основного потоку. Ця ізольована підгрупа знову зливається з основною групою мовлення лише тоді, коли алгоритми протоколу вирівняють швидкості отримання даних і проблемні клієнти повністю синхронізуються з вихідним потоком трансляції сервера.

На противагу складним архітектурам надійних протоколів, стандартний протокол UDP залишається найпопулярнішим, найпростішим та найшвидшим інструментом для транспортування датаграм у сучасних IP-мережах. Його ключова особливість (вона ж і головний недолік для певних задач) полягає в повній відсутності механізмів гарантованої доставки, підтверджень отримання, контролю цілісності послідовності пакетів та управління перевантаженнями [15].

Саме ця «легковажність» і мінімальний розмір заголовка пакета роблять UDP ідеальним, неперевершеним вибором для трансляції чутливого до затримок трафіку, який толерантний до незначних втрат інформації — насамперед, для систем потокового відео, IP-телефонії (VoIP) та онлайн-геймінгу, рис. 1.4.

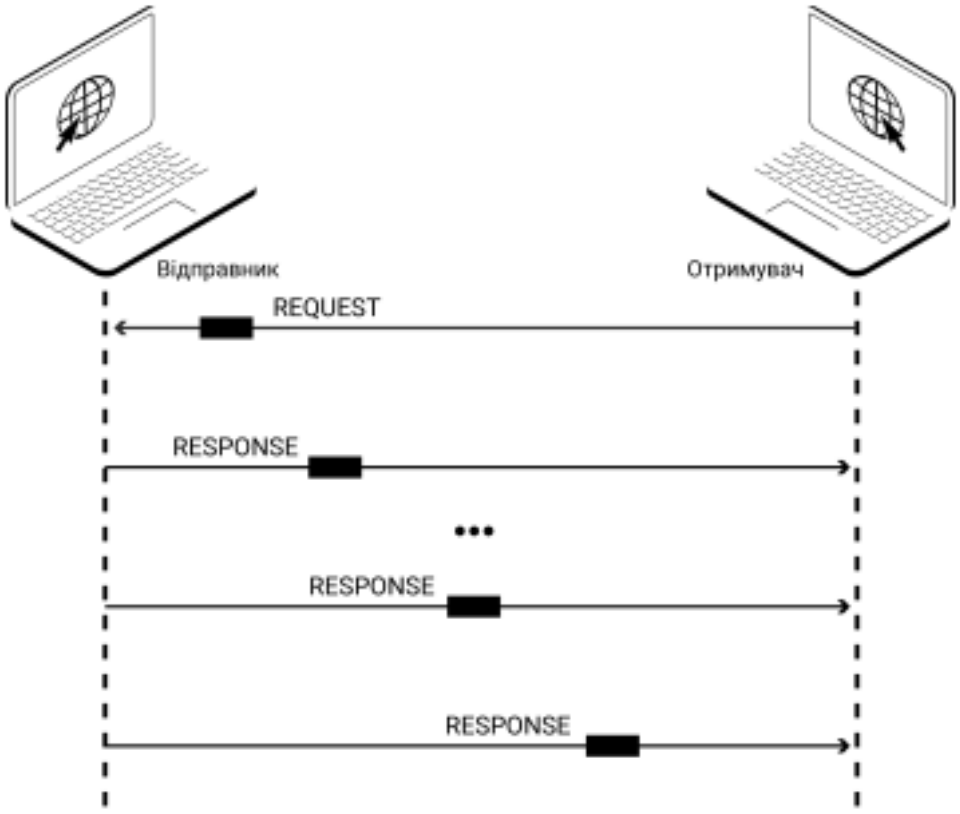


Рисунок 1.4 — Базова архітектура взаємодії мережевих пристроїв та транспортування даних на базі протоколу UDP

Фундаментальне призначення UDP — забезпечити максимально швидку передачу пакетних даних у мережах з комутацією пакетів (як в локальних інфраструктурах, так і в глобальній мережі Інтернет). UDP розглядає базовий протокол мережевого рівня (IP) як транспортну основу нижчого рівня і працює безпосередньо над ним. На відміну від TCP, який є суворо протоколом «точка-точка» (unicast), UDP має унікальну і критично важливу для сучасних мереж здатність підтримувати широкомовні (Broadcast) та багатоадресні (Multicast) розсилки, рис. 1.5.

Технологія Broadcast (широкомовлення) передбачає лавинне надсилання копії пакета абсолютно всім активним пристроям, що знаходяться в межах одного логічного широкомовного домену (зазвичай, в межах однієї підмережі або VLAN). Це вкрай ефективний метод для локальних задач: наприклад, для розсилки службових сповіщень про стан мережевих пристроїв, пошуку

сервісів (як у протоколі DHCP чи ARP) або базових мультимедійних трансляцій у невеликих корпоративних мережах. Однак у великих масштабах Broadcast є неприпустимим, оскільки він створює колосальне сміттєве навантаження на мережу.

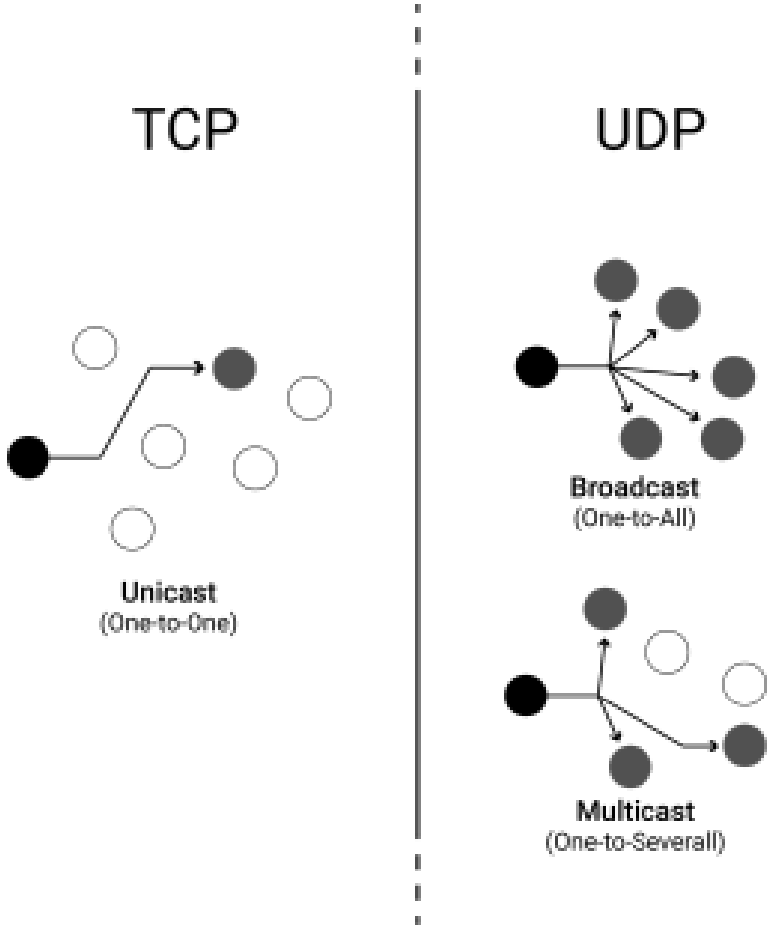


Рисунок 1.5 — Візуальне порівняння топологій маршрутизації: широкомовлення (Broadcast) та багатоадресна розсилка (Multicast) у середовищі UDP на противагу одноадресній (Unicast) передачі TCP

Технологія Multicast UDP (багатоадресна розсилка) вирішує проблему неефективності широкомовлення. Цей механізм дозволяє серверу відправляти єдиний потік даних, який мережеве обладнання (маршрутизатори) динамічно дублює і доставляє виключно тій визначеній групі клієнтських пристроїв, які явно виявили бажання отримувати цей конкретний контент (тобто підписалися

на певну мультикаст-групу). Ця архітектура забезпечує максимально раціональне використання пропускної здатності магістральних каналів зв'язку, оскільки важкий трафік (наприклад, IPTV-канали у форматі 4K) не розсилається сліпо всім підряд, а маршрутизується лише до тих абонентських приставок чи комп'ютерів, які дійсно його запитали [16].

1.3 Комплексний порівняльний аналіз сучасних методів та протоколів передачі даних

Підсумовуючи результати глибокого огляду існуючих протоколів сімейства надійної багатоадресної передачі (Reliable Multicast), можна сформувати чіткі висновки щодо їхньої архітектурної специфіки та сфер найефективнішого застосування в сучасних мережевих інфраструктурах.

Розглядаючи архітектуру протоколу PGM, варто зазначити, що програмні комплекси, розгорнуті на його базі, принципово не орієнтовані на отримання детальної статистики щодо успішної доставки даних кожному окремому клієнту. Замість тотального контролю за кожним вузлом, PGM робить ставку на оптимізацію мережевих ресурсів. Завдяки імплементації інтелектуального механізму негативних підтверджень (NAK), вбудованим алгоритмам придушення зайвих запитів (NAK suppression) та чітко організованій ієрархічній структурі маршрутизації, цей протокол демонструє відмінні показники масштабованості. Він здатний гарантувати високонадійну доставку масивів інформації для тисяч підписників одночасно, не викликаючи перевантаження джерела (ACK-штормів). Водночас інженерам необхідно враховувати його структурні обмеження: PGM абсолютно не пристосований для роботи в умовах суто односпрямованих каналів зв'язку (наприклад, у базових супутникових топологіях без виділеного зворотного каналу) та має складнощі з коректною обробкою високодинамічних, переривчастих потоків трафіку. Крім того, для досягнення максимальної продуктивності протокол

вимагає інтеграції в середовище, де транзитне мережеве обладнання (зокрема, сучасні високопродуктивні L3-комутатори) забезпечує йому хоча б часткову апаратну підтримку. З огляду на ці характеристики, PGM є оптимальним та потужним інструментом для задач масового розповсюдження об'ємних блоків даних, таких як централізоване розгортання образів операційних систем, масова дистрибуція контейнерів програмного забезпечення або синхронізація великих розподілених баз даних.

На противагу PGM, протокол M/TCP реалізує кардинально інший підхід до контролю якості обслуговування. Ця технологія ідеально задовольняє жорсткі вимоги тих корпоративних додатків, для яких критично важливо мати стовідсоткову, підтверджену інформацію про те, що абсолютно всі авторизовані вузли успішно отримали надіслані пакети даних. Такий абсолютний рівень прозорості досягається завдяки використанню класичного механізму позитивних підтверджень (ACK). Проте, саме цей механізм контролю стає головним вузьким місцем системи при спробі її розширення: через необхідність постійної обробки лавини вхідних квитанцій від клієнтів, M/TCP здатен стабільно функціонувати лише в дуже компактних мультикаст-групах з жорстко обмеженою кількістю абонентів. Незаперечною перевагою M/TCP є його виняткова стійкість і гнучкість у роботі з нерівномірними, пульсуючими або переривчастими потоками даних, що робить його незалежним від нестандартних профілів генерації трафіку. Однак розгортання такої інфраструктури є складним інженерним завданням, оскільки M/TCP вимагає повної та глибокої підтримки своєї нестандартної логіки на всіх проміжних мережевих вузлах, висуваючи підвищені вимоги до процесорних потужностей маршрутизаторів. Зважаючи на зазначену специфіку, протокол M/TCP найкраще розкриває свій інноваційний потенціал у вузькоспеціалізованих інформаційних нішах, наприклад, під час організації захищених службових текстових чатів з гарантованою доставкою повідомлень або для забезпечення безперебійного функціонування критичних систем генерації Push-сповіщень в інфраструктурі підприємства [17].

2 ТЕОРЕТИКО-МЕТОДОЛОГІЧНІ ОСНОВИ ПОБУДОВИ СИСТЕМ БАГАТОАДРЕСНОГО ОБМІНУ ДАНИМИ

2.1 Архітектурні рішення та базові компоненти мережі

Основою запропонованої інфраструктури є магістральна мережа, яка забезпечує глобальне з'єднання та обмін даними між усіма учасниками системи. Архітектура передбачає, що відправники та одержувачі інформації підключаються до цього центрального ядра не безпосередньо, а з використанням проміжних ланок — локальних комутаторів доступу. Таке підключення може бути реалізоване як прямим каналом зв'язку, так і опосередковано через додаткові транзитні вузли, що детально відображено на концептуальній схемі базової моделі мережі, рис. 2.1.

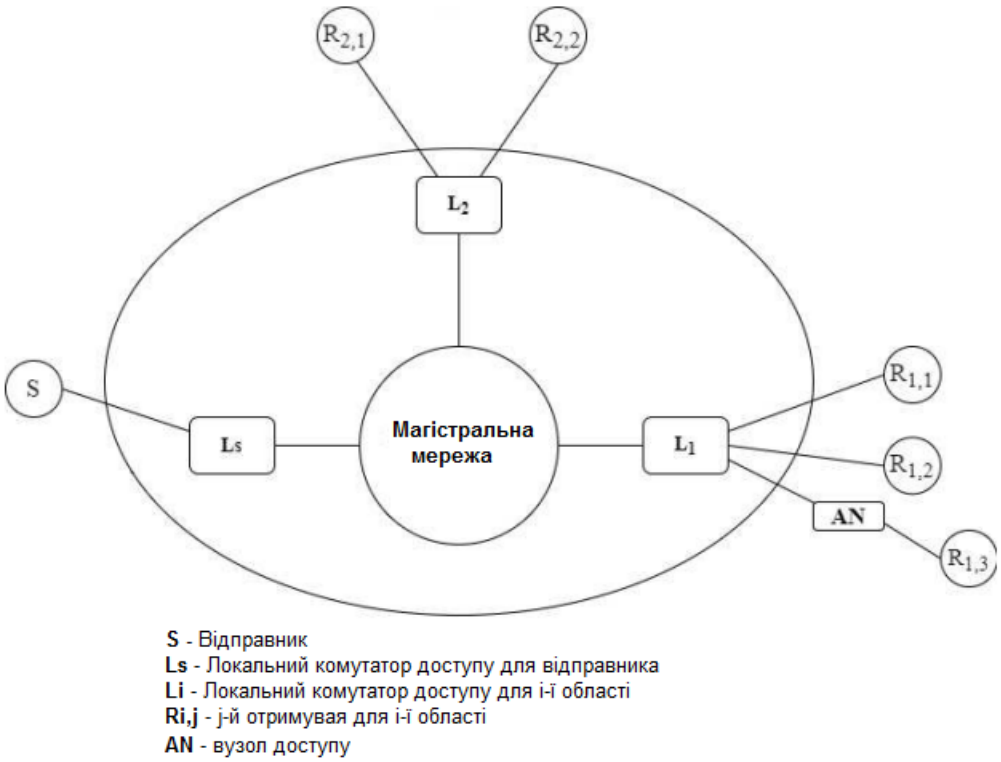


Рисунок 2.1 — Концептуальна модель та базова топологія запропонованої мережевої інфраструктури

Для ефективної маршрутизації трафіку використовується концепція багатоадресних дерев. Безпосередньо на мережевому рівні формується так зване глобальне багатоадресне дерево [18]. У цій логічній структурі вузол-відправник виступає в ролі єдиного кореня, який генерує потік даних, тоді як кінцеві одержувачі утворюють множину вершин (або листків) цього графа. Окрім основної глобальної структури, топологія включає локальні багатоадресні дерева, які логічно функціонують як відокремлені сегменти або гілки загальної системи. Наочне відображення зв'язків глобального багатоадресного дерева, спеціально виділеного суцільними лініями для візуального розмежування, наведено на рис. 2.2.

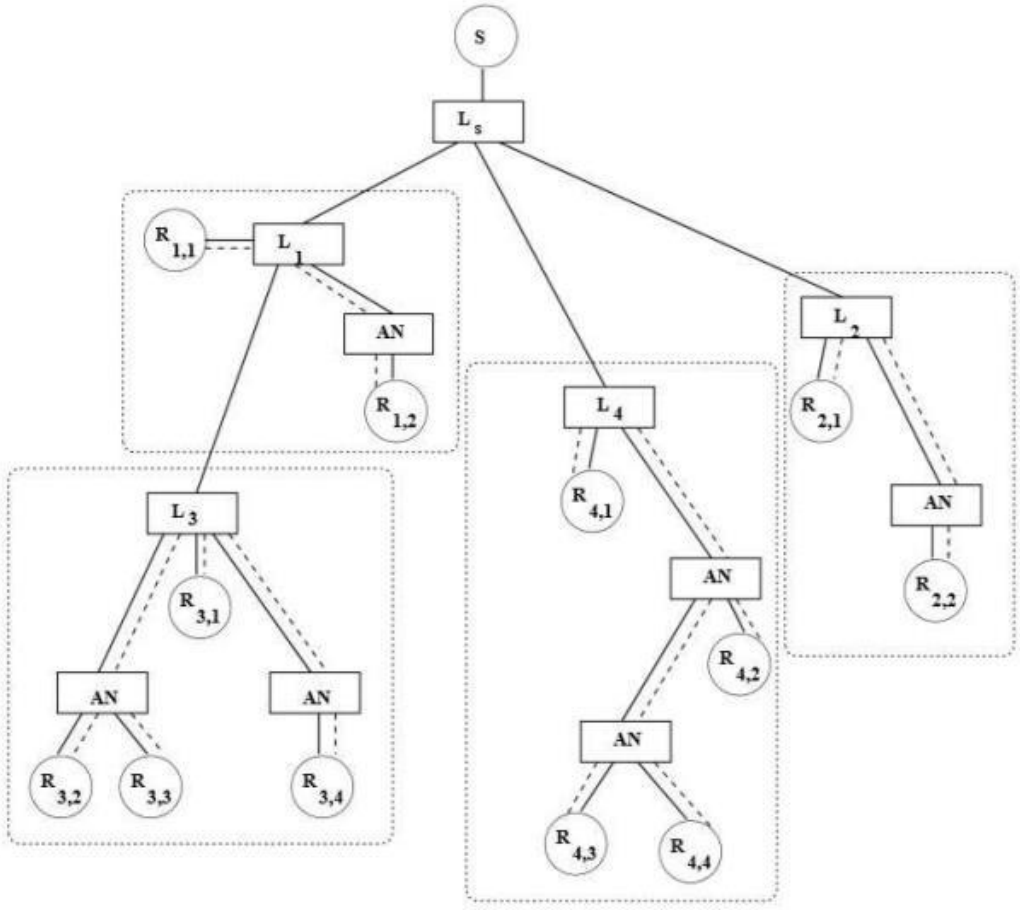


Рисунок 2.2 — Структурна організація глобального та локальних багатоадресних дерев маршрутизації

У межах цієї ієрархічної моделі кінцеві одержувачі, що географічно або топологічно знаходяться у певній локальній зоні та обслуговуються відповідним мережевим обладнанням, мають власну логічну ідентифікацію. Зокрема, виділяється локальний комутатор доступу для кожної окремої області, тоді як множина підключених до нього приймачів розглядається як єдина цільова підгрупа. Важливо підкреслити специфіку їхньої взаємодії: базове позначення локальних зон вказує виключно на комутуюче обладнання, що агрегує трафік, проте в певних сценаріях маршрутизації ці локальні комутатори доступу також здатні виступати в ролі повноцінних транзитних одержувачів даних [19].

Головною вимогою до транспортного протоколу надійної багатоадресної передачі у цій архітектурі є здатність базової мережі успішно встановити та безперервно підтримувати дерево маршрутизації від джерела до множини цільових вузлів. При цьому архітектура вирізняється високою адаптивністю: для коректного та стабільного функціонування протоколу не вимагається жорсткого попереднього резервування пропускної здатності чи виділення додаткових апаратних ресурсів. Основна місія алгоритму зводиться до гарантування суворо послідовної доставки пакетів даних від відправника до кожного одержувача вздовж сформованих шляхів дерева багатоадресної передачі, абсолютно незалежно від того, які саме протоколи нижнього рівня використовувалися для створення цього дерева та розподілу навантаження.

Проектування та розгортання протоколу спирається на глибоку інтеграцію кількох фундаментальних архітектурних концепцій. Насамперед алгоритм базується на принципі просторової кластеризації: мережеві приймачі логічно об'єднуються у специфічні локальні групи, виходячи з їхньої топологічної близькості. Наприклад, за умови використання ієрархічної схеми адресації, групування вузлів може здійснюватися на основі регіональних ідентифікаторів, підмереж або кодів міст у протоколах сімейства IP. Альтернативним і вельми ефективним методом локалізації в IP-мережах є гнучке використання поля часу життя пакета, що дозволяє апаратно

обмежувати радіус поширення багатоадресного трафіку і знижувати загальне навантаження на систему.

Наступним критично важливим аспектом є механізм ініціалізації дерева розсилки. Формування єдиної глобальної структури, яка бере початок від вузла-відправника та охоплює всю множину цільових клієнтів, реалізується глибоко на мережевому рівні. Історично подібний підхід застосовувався на рівні асинхронного режиму передачі в класичних мережах, проте базові принципи побудови глобального дерева (як проілюстровано на рис. 2.2) залишаються незмінними і для сучасних середовищ. Чітке розмежування глобальних і локальних дерев, а також диференціація ролей комутаторів агрегації та кінцевих вузлів-приймачів дозволяє уникнути петель маршрутизації та втрат пакетів на стиках зон відповідальності.

Зрештою, розроблений протокол демонструє значний потенціал до масштабування різних моделей взаємодії. Його базовий функціонал оптимізований для забезпечення гарантованої групової трансляції за односпрямованою топологією від однієї точки до багатьох. Разом з тим, гнучкість архітектури дозволяє адаптувати систему для значно складніших інтерактивних сценаріїв обміну даними між багатьма учасниками одночасно. У такому розширеному форматі надійна багатоадресна розсилка досягається шляхом симетричного налаштування індивідуальних, незалежних дерев маршрутизації для кожного активного вузла-відправника в мережі, що запобігає колізіям та мінімізує затримки при масовому обміні інформацією.

2.2 Архітектура та алгоритмічна база мережевого протоколу

Адаптивний транспортний протокол багатоадресного зв'язку (АТПБЗ) спроектований для забезпечення гарантованої та послідовної доставки великих масивів даних від одного джерела до широкого кола споживачів без

жодних втрат інформації. В основі його роботи лежить механізм вибіркової повторної передачі тих фрагментів, які не були вчасно отримані вузлами-адресатами. Такий підхід дозволяє оптимізувати трафік, оскільки повторно надсилаються лише втрачені пакети, а не весь масив даних.

Однією з ключових проблем класичних схем багатоадресної розсилки є так звана проблема «активного перевантаження підтвердженнями», коли велика кількість одночасних статусних повідомлень від отримувачів фактично блокує роботу відправника [20]. Крім того, наявність значної відстані між джерелом та деякими клієнтами призводить до зростання затримок та критичного падіння пропускної здатності. Для подолання цих недоліків у АТПБЗ реалізовано ієрархічну модель, що базується на деревоподібній структурі.

Основним архітектурним рішенням є поділ отримувачів на локальні сегменти, у кожному з яких призначається відповідальний вузол — виділений приймач. Незважаючи на те, що первинна розсилка здійснюється джерелом для всієї глобальної мережі одночасно, зворотний зв’язок організовано інакше. Тільки виділені приймачі мають право безпосередньо контактувати з відправником, інформуючи його про стан успішності отримання пакетів. Усі інші вузли в межах своєї локальної області взаємодіють виключно зі своїм призначеним локальним координатором. Виділений приймач не просто збирає статуси, а самостійно виконує повторну передачу втрачених даних у межах своєї зони відповідальності, що дозволяє значно скоротити час очікування. За такої схеми відправник «бачить» лише обмежену кількість координаторів, а кожен координатор опікується лише своєю невеликою групою клієнтів.

Ця ієрархія візуалізована на схемі взаємодії вузлів у межах глобального та локальних сегментів розсилки, рис. 2.2, де певні вузли обрані як координатори для своїх підгруп. Локальні дерева маршрутизації, корінням яких є ці координатори, логічно вкладені в загальну структуру мережі.

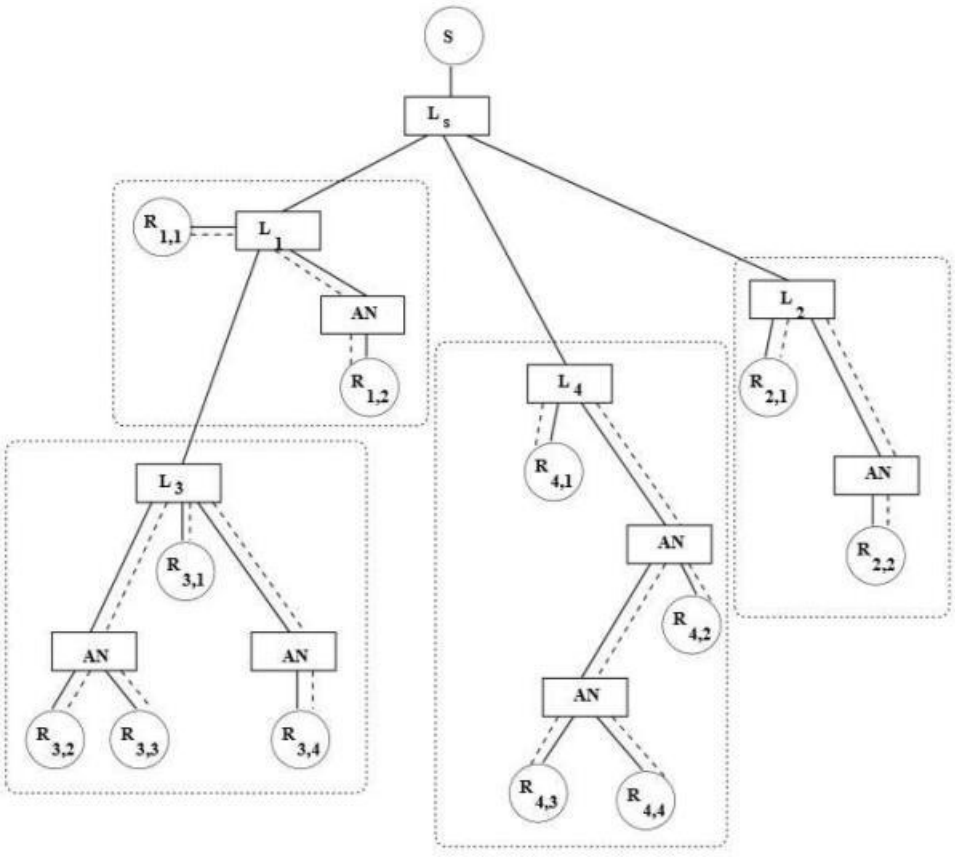


Рисунок 2.2 — Схема ієрархічної взаємодії вузлів у межах глобального та локальних сегментів розсилки

Функціонування протоколу спирається на регулярний обмін пакетами стану. Кожен виділений координатор з певною періодичністю надсилає джерелу звіт, де детально вказано, які фрагменти даних були успішно верифіковані. На основі цих звітів відправник приймає рішення: якщо кількість запитів на повторну передачу певного пакета перевищує встановлений критичний поріг, він знову транслює його на всю мережу; якщо ж запитів мало — пакет надсилається лише конкретним координаторам. Аналогічна логіка працює і на локальному рівні: координатор розсилає дані своїй групі або багатоадресно, або індивідуально, залежно від масштабу втрат.

Відправник ініціює передачу нових блоків даних лише за умови наявності вільного простору у вікні передачі. Весь масив інформації розбивається на пакети однакового розміру, за винятком фінального фрагмента, який маркується спеціальним ідентифікатором завершення потоку.

Кожному пакету присвоюється унікальний порядковий номер, починаючи з нульового значення.

Кожен учасник сеансу періодично формує пакети підтвердження. Таке повідомлення містить інформацію про нижню межу вікна прийому та спеціальний бітовий масив фіксованого розміру, який виконує роль карти отриманих даних: кожен біт у ньому відповідає певному слоту в буфері, де позитивне значення сигналізує про коректне прийняття пакета. Параметри з’єднання визначаються мережевими адресами та портами обох сторін [21]. Управління сеансом, включаючи налаштування часових інтервалів та лімітів вікна, покладається на спеціальний диспетчер сеансів.

У АТПБЗ реалізована динамічна модель участі: відправнику не потрібно знати точний перелік чи кількість отримувачів. Вузли можуть вільно приєднуватися до групи або залишати її під час трансляції [22]. Через відсутність централізованого обліку учасників, завершення сеансу відбувається за часовим принципом. Після відправлення останнього фрагмента даних запускається таймер очікування. Якщо протягом визначеного часу не надходить нових підтверджень, відправник вважає сеанс успішно завершеним і видаляє інформацію про з’єднання.

У разі виникнення критичних помилок або передчасного завершення роботи програми-джерела, протокол використовує спеціальний механізм скидання з’єднання, що інформує всіх учасників про примусову зупинку процесу [23].

Логіка роботи протоколу розподілена між трьома основними функціональними компонентами: відправником, виділеним координатором та отримувачем, рис. 2.3.

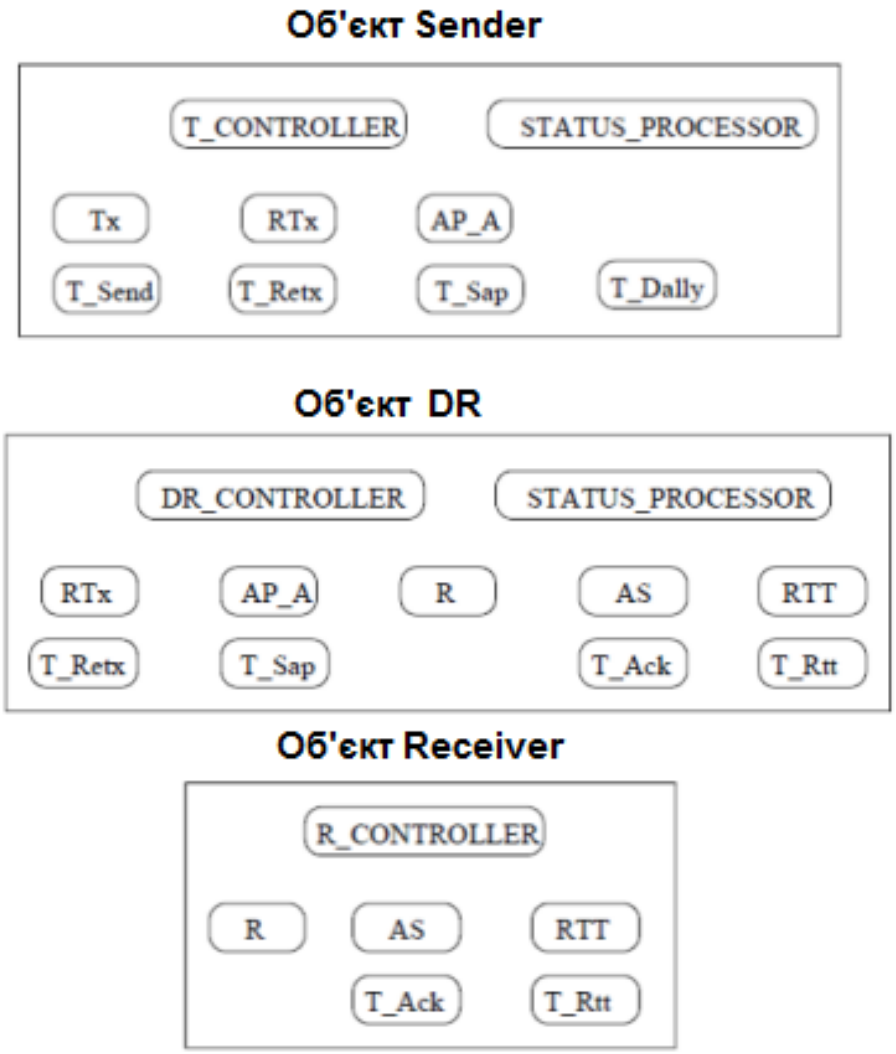


Рисунок 2.3 — Функціональна модель взаємодії основних компонентів системи та їх внутрішніх контролерів

Компонент відправника містить керуючий модуль, який на основі внутрішніх таймерів визначає пріоритетність дій: трансляція нових даних, повторна передача втрачених фрагментів або розсилка сервісної інформації про активність джерела. Окремий аналітичний модуль обробляє вхідні підтвердження та оновлює карту стану вікна передачі.

Компонент отримувача фокусується на передачі верифікованих даних кінцевому додатку, регулярному формуванні звітів про стан прийому та динамічному обчисленні затримок у мережі для адаптації таймерів.

Виділений координатор функціонально поєднує можливості обох попередніх об’єктів, виступаючи одночасно як отримувач для джерела та як локальне джерело для своєї групи.

Для забезпечення безперебійної роботи використовуються механізми буферизації. Наприклад, стан вікна прийому можна уявити як набір комірок, рис. 2.4. Якщо в буфері зафіксовано отримання пакетів, наприклад, з шістнадцятого по сімнадцятий та дев’ятнадцятого, але відсутній вісімнадцятий, бітовий масив точно вкаже на цю прогалину. При цьому протокол суворо дотримується черговості: дані передаються додатку лише в послідовному порядку. Якщо виявлено розрив у нумерації, передача вищих за номером пакетів призупиняється до моменту успішного отримання та вставки відсутньої ланки в ланцюжок.

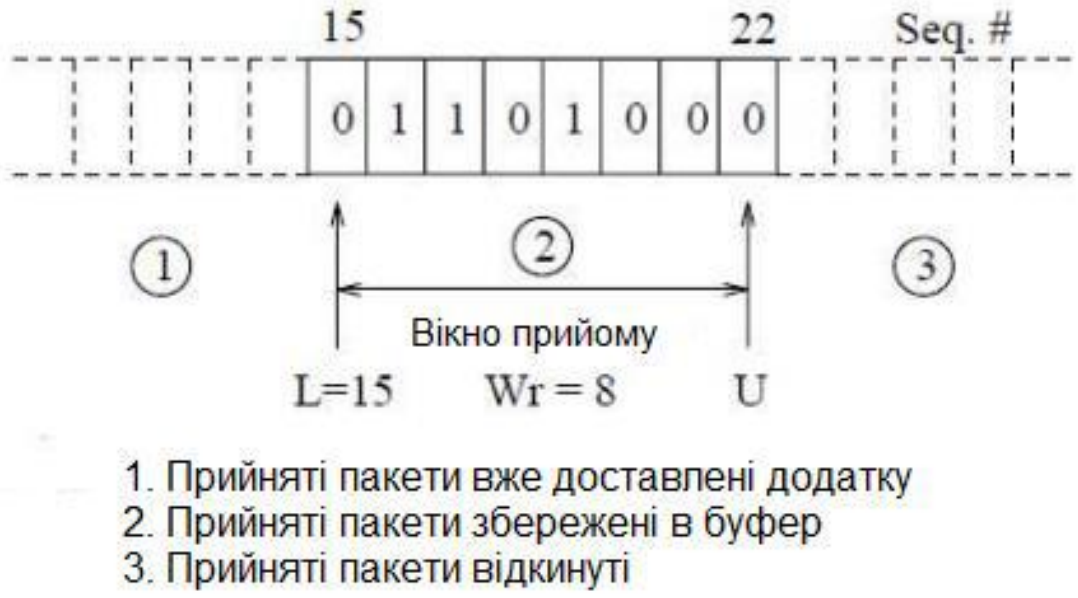


Рисунок 2.4 — Візуалізація механізму ковзного вікна прийому та стану заповнення буфера даних

Однією з ключових особливостей протоколу є механізм регулярного сповіщення джерела про стан прийому даних. Приймачі генерують статусні повідомлення з певною періодичністю. Проте, якщо інтенсивність таких сповіщень буде занадто високою, вузол, відповідальний за їх обробку (джерело

або локальний координатор), може опинитися в ситуації невизначеності. Він ризикує безперервно здійснювати повторну передачу одного й того ж фрагмента даних, просто не встигаючи отримати підтвердження про його успішну доставку від усіх адресатів.

Для запобігання такому марнотратству ресурсів та надлишковому трафіку, у системі реалізовано алгоритм динамічного вимірювання затримок. Кожен приймач самостійно розраховує час, необхідний для здійснення повного циклу прийому-передачі (Round-Trip Time) між собою та своїм безпосереднім координатором. Цей процес ініціюється відразу після встановлення з'єднання шляхом відправлення спеціального тестового пакета, до якого приймач додає власну локальну часову мітку. Надалі такі вимірювальні пакети генеруються із суворо фіксованим інтервалом.

Коли координатор отримує тестовий пакет, він миттєво змінює його службовий ідентифікатор і віддзеркалює назад до відправника. Повернувши свій пакет, приймач фіксує поточний час і, порівнюючи його зі збереженою часовою міткою, обчислює фактичну тривалість проходження сигналу в обох напрямках. Саме на основі цього динамічного показника приймач математично розраховує оптимальний інтервал паузи перед відправленням наступного статусного підтвердження, адаптуючись до поточного стану мережі.

Процес відновлення втрачених даних курується або головним відправником, або призначеним локальним координатором. Аналізуючи вхідні підтвердження, обробник формує точний перелік пропущених фрагментів. Оскільки в умовах багатоадресної розсилки один і той самий пакет може бути втрачений одночасно кількома різними одержувачами, алгоритм повинен прийняти оптимальне рішення щодо способу його повторної доставки.

Для цього використовується спеціальна черга очікування та система граничних значень. Коли надходить запит на відновлення, ідентифікатор відповідного пакета потрапляє до черги. Кожен запис у цій черзі містить детальну інформацію: номер втраченого фрагмента, динамічний лічильник

загальної кількості вузлів, які його потребують, таблицю з їхніми точними адресами та системні вказівники для управління пам'яттю.

Рішення приймається після завершення спеціального таймауту очікування. Якщо до моменту спрацьовування таймера лічильник запитів на конкретний пакет перевищує встановлений системою поріг масовості, координатор визнає втрату глобальною (або масштабною для своєї локальної зони) і здійснює повторну багатоадресну розсилку цього пакета всій групі. Якщо ж кількість запитів виявляється меншою за критичний поріг, інцидент кваліфікується як локальний збій, і координатор використовує збережену таблицю адрес для точкової, індивідуальної відправки пакета лише тим вузлам, які про нього просили.

Для контролю інтенсивності вихідного потоку даних відправник оперує концепцією динамічного вікна відправки. Цей механізм базується на відстеженні трьох ключових параметрів:

1. Поточної нижньої межі вікна, що позначає найстаріший пакет, який ще очікує підтвердження;
2. Вказівника на наступний вільний ідентифікатор для маркування нових даних;
3. Фактичного обсягу доступного простору (квоти) для генерації нового трафіку.

Кожна відправка нової порції інформації зміщує вказівник уперед і відповідно зменшує доступну квоту. Коли джерело отримує підтвердження про успішну доставку найстаріших пакетів (тих, що знаходяться на нижній межі), ця межа зсувається вперед, пропорційно вивільняючи нове місце у вікні та відновлюючи квоту для передачі. Візуалізацію цього процесу та структуру самого вікна наведено на рис. 2.5.

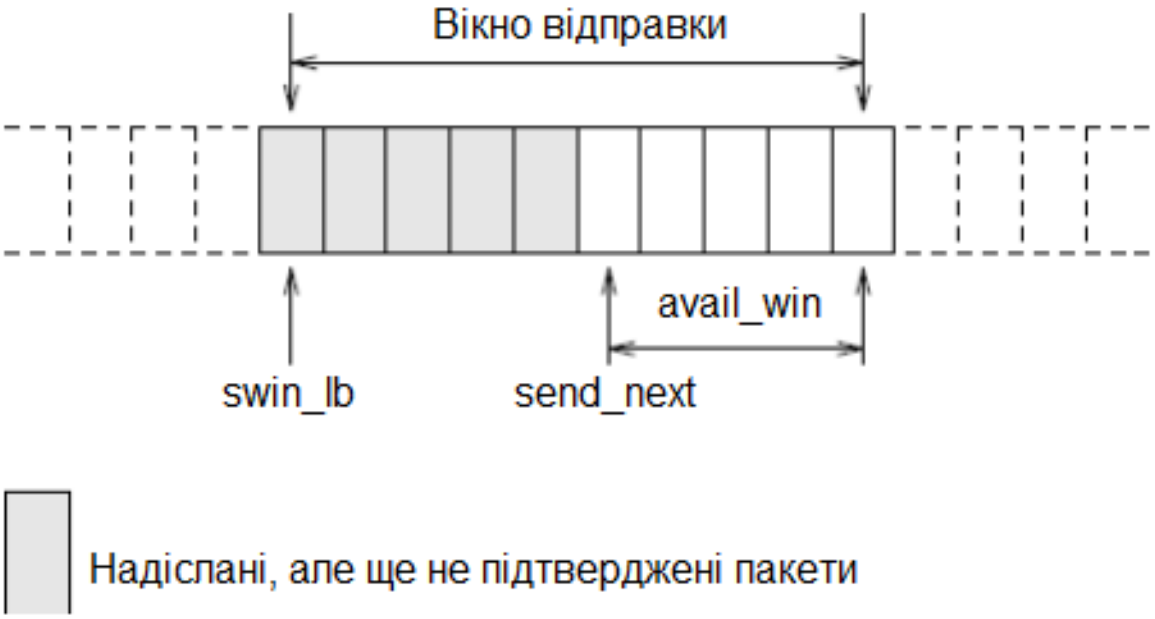


Рисунок 2.5 — Механізм роботи ковзного вікна відправлення та розподіл його параметрів

Щоб адаптивно розрахувати, скільки саме нових блоків можна згенерувати в наступному такті, відправник аналізує масив отриманих підтверджень за останній період. Він знаходить мінімальне значення підтвердженої межі серед усіх відповідей. Якщо цей мінімум перевищує поточну системну межу вікна відправника, вікно успішно зміщується вперед на різницю цих значень, а доступна квота пропорційно зростає. Важливим правилом безпеки протоколу є те, що нижня межа вікна відправки може рухатися лише вперед; будь-які запізнілі статусні повідомлення, що вказують на вже пройдені етапи, системою ігноруються.

АТПБЗ розроблений з урахуванням високої динаміки мережевих сеансів, дозволяючи новим учасникам вільно приєднуватися до трансляції на будь-якому її етапі. Крім того, протокол враховує ситуації, коли існуючі приймачі можуть тимчасово «відставати» від загального потоку через локальні перевантаження маршрутизаторів. Для забезпечення цілісності даних у таких нестандартних сценаріях використовуються два взаємопов'язані механізми: експрес-запит на синхронізацію та глибоке дворівневе кешування.

Коли новий вузол підключається до вже активного сеансу, він одразу починає отримувати поточний багатоадресний трафік. Проаналізувавши порядкові номери вхідних пакетів, приймач миттєво усвідомлює розрив і розуміє, що пропустив попередню частину масиву. У цей момент він формує спеціальний пріоритетний запит до свого локального координатора. Цей запит структурно ідентичний звичайному підтвердженню, але маркований особливим ідентифікатором терміновості. Отримавши такий сигнал, координатор позачергово аналізує прикріплену карту пропущених пакетів і негайно ініціює їх індивідуальну (одноадресну) передачу новому учаснику.

Щоб ця функція працювала безвідмовно, система спирається на стратегію тотального збереження інформації. Відправник та всі виділені координатори зобов'язані буферизувати весь масив даних від самого початку і до завершення сеансу трансляції. Оптимізація ресурсів досягається за рахунок дворівневої архітектури пам'яті: найбільш актуальні (останні) пакети утримуються у швидкій оперативній пам'яті для миттєвого доступу, тоді як старіші сегменти архівуються на жорсткі диски.

Формування ієрархічної структури відбувається автоматично. Протокол здійснює первинний аналіз топології (наприклад, оцінюючи IP-адреси в підмережах) і призначає найбільш оптимальні вузли на роль локальних координаторів. Далі ці координатори починають періодично розсилати довкола себе спеціальні маякові пакети із заданим обмеженням радіусу дії (TTL). Звичайний приймач, отримавши кілька таких сигналів від різних координаторів, оцінює їх і логічно «прив'язується» до того вузла, чий сигнал виявився топологічно найближчим.

Зважаючи на непередбачуваність інтернет-середовища, де вузли постійно з'являються та зникають без попередження [24], класичні жорсткі алгоритми керування потоком (які вимагають підтвердження від абсолютно всіх учасників перед продовженням передачі) тут неефективні. Джерело фізично не може знати точну кількість активних координаторів у поточний

момент. Тому в АТПБЗ реалізовано адаптивне керування на основі швидкості та циклічності.

Відправник працює ітераціями. Замість того, щоб чекати підтвердження від кожного існуючого вузла, він орієнтується лише на тих активних координаторів, які встигли надіслати звіти протягом визначеного часового вікна. Переконавшись, що ці активні учасники отримали свої дані, відправник зсуває вікно вперед. При цьому старі дані не видаляються, а залишаються в кеші для обслуговування можливих запізнілих запитів. Максимальна інтенсивність генерації трафіку жорстко лімітована, що не дозволяє джерелу монополізувати канал.

Для захисту магістральних каналів від колапсу, протокол використовує кількість запитів на повторну передачу як прямий індикатор перевантаження мережі. Відправник оперує додатковим лімітом — «вікном перевантаження». Протягом кожного циклу він підраховує кількість повідомлень про втрати. Якщо цей показник перевищує критичну межу, система діагностує серйозний затор і миттєво зменшує швидкість передачі до абсолютного мінімуму (буквально до одного пакета за такт). Якщо ж ситуація стабілізується і кількість помилок падає нижче порогу, відправник починає обережно, лінійно збільшувати пропускну здатність, доки не досягне нормальних робочих показників. Цей механізм, відомий як «м'який старт» (Slow Start), є класичним високоефективним рішенням, запозиченим із еталонних реалізацій транспортних протоколів.

3 ПРАКТИЧНА РЕАЛІЗАЦІЯ ТА ДОСЛІДЖЕННЯ

3.1 Характеристика експериментального середовища та інфраструктури

Для проведення всебічного аналізу та перевірки працездатності розробленого протоколу було сформовано спеціалізований імітаційний стенд. Його архітектура базується на чотирьох функціональних вузлах, об’єднаних у єдину мережеву структуру за чітко визначеною топологією. Детальне просторове розміщення компонентів та логічні зв’язки між ними наведено на схемі моделі віртуального стенду, рис. 3.1.

Програмно-апаратна база стенду реалізована на платформі віртуалізації промислового рівня VMWare ESXi. Кожен окремий вузол системи функціонує як ізольована віртуальна машина під управлінням операційної системи сімейства Linux, а саме дистрибутива Ubuntu версії 12.04.5 LTS. Такий вибір ОС забезпечує необхідну стабільність та сумісність із системними інструментами мережевого адміністрування.

З точки зору мережевої архітектури, вузли, що виконують ролі джерела даних та кінцевих отримувачів, оснащені одним мережевим інтерфейсом для взаємодії з інфраструктурою, рис. 3.1.

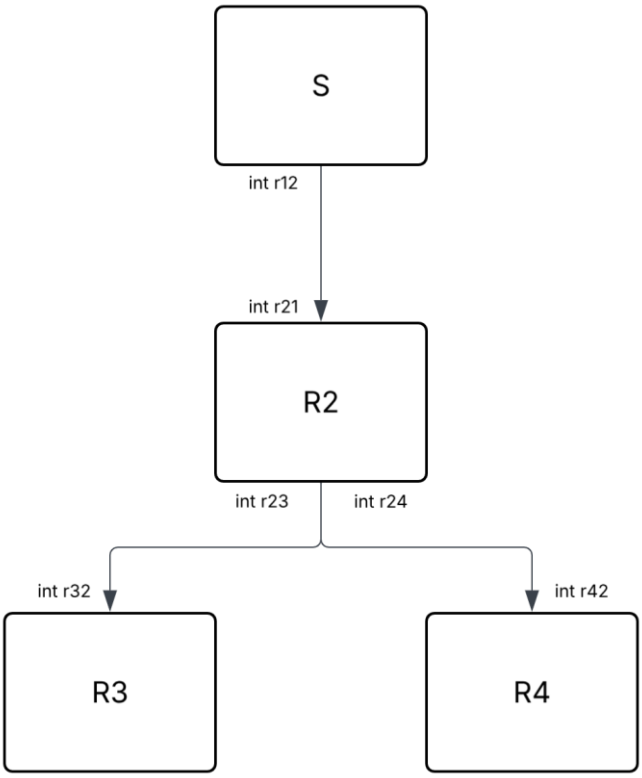


Рисунок 3.1 — Концептуальна схема та логічна топологія віртуального випробувального стенду

Для контролю коректності функціонування кожного сегмента системи проводиться моніторинг активності інтерфейсів; прикладом такого аналізу є стан мережевого адаптера на одному з вузлів-отримувачів, дані про який зафіксовано у системному журналі, рис. 3.2.

```
r32       Link encap:Ethernet  HWaddr 2a:1b:3c:4d:5e:6f
          inet6 addr: fe80::281b:3cff:fe4d:5e6f/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:45 errors:0 dropped:0 overruns:0 frame:0
          TX packets:38 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:4120 (4.1 KB)  TX bytes:2840 (2.8 KB)
```

Рисунок 3.2 — Моніторинг поточного стану та параметрів мережевого інтерфейсу на вузлі-отримувачі

Фізичним рівнем передачі даних у моделі виступає комутована мережа стандарту Ethernet, що забезпечує високу швидкість обміну інформацією. Особливу роль у цій структурі відіграє вузол, призначений на роль локального координатора (виділеного приймача). Для виконання своїх специфічних функцій він оснащений трьома мережевими інтерфейсами. За допомогою інструментарію управління мережевими стеками ці інтерфейси логічно об’єднані в єдиний мережевий міст.

Така конфігурація координатора є критично важливою, оскільки вона дозволяє вузлу ефективно виконувати роль ретранслятора, здійснюючи широкомовну передачу даних до підпорядкованих йому отримувачів у локальному сегменті. Детальні відомості про розподіл адрес та статус усіх активних каналів зв’язку на вузлі-координаторі представлені на рис. 3.3.

```
my_bridge Link encap:Ethernet  HWaddr 00:11:22:aa:bb:cc
          inet6 addr: fe80::0211:22ff:feaa:bbcc/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:4 errors:0 dropped:0 overruns:0 frame:0
          TX packets:28 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:320 (320.0 B)  TX bytes:2410 (2.4 KB)

r21       Link encap:Ethernet  HWaddr 12:34:56:78:90:ab
          inet6 addr: fe80::1034:56ff:fe78:90ab/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:18402 errors:0 dropped:2 overruns:0 frame:0
          TX packets:3104 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:18824100 (18.8 MB)  TX bytes:56210 (56.2 KB)

r23       Link encap:Ethernet  HWaddr aa:bb:cc:dd:ee:ff
          inet6 addr: fe80::a8bb:ccff:fedd:eeff/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:24 errors:0 dropped:0 overruns:0 frame:0
          TX packets:24 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1200 (1.2 KB)  TX bytes:1944 (1.9 KB)

r24       Link encap:Ethernet  HWaddr 00:11:22:aa:bb:cc
          inet6 addr: fe80::0211:22ff:feaa:bbcc/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:12 errors:0 dropped:0 overruns:0 frame:0
          TX packets:24 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:972 (972.0 B)  TX bytes:1944 (1.9 KB)
```

Рисунок 3.3 — Аналіз конфігурації мережевого моста та активності інтерфейсів на вузлі-координаторі

Програмна логіка самого протоколу реалізована за допомогою мови програмування Python. З огляду на це, на кожній віртуальній машині було розгорнуто середовище виконання Python версії 2.3.7, що гарантує ідентичність обробки алгоритмів на всіх рівнях ієрархії. Для забезпечення максимально низьких затримок та високої пропускної здатності під час експериментів, взаємодія між віртуальними машинами здійснюється через високошвидкісні віртуальні канали Gigabit Ethernet, що дозволяє максимально наблизити умови тестування до реальних високопродуктивних мереж.

3.2 Організація та етапи експериментальної перевірки.

Практична перевірка працездатності розробленого протоколу була здійснена шляхом серії контрольних запусків у сформованому віртуальному середовищі. Програмна реалізація алгоритмів виконана мовою Python з використанням механізму низькорівневого програмування — сирих сокетів (raw sockets). Вибір цього інструментарію зумовлений необхідністю повного контролю над процесом формування мережевих пакетів у просторі користувача, що дозволяє реалізовувати нестандартні протоколи стека IPv4 без обмежень, які накладають високорівневі системні бібліотеки.

Процес тестування розпочинається з одночасного запуску керуючого програмного модуля на всіх вузлах віртуального стенду. Логіка головного файлу програми побудована таким чином, що залежно від переданих аргументів командного рядка, кожна віртуальна машина автоматично активує один із трьох функціональних режимів:

— Відправник: ініціює передачу даних та контролює стан глобального вікна;

— Звичайний приймач: фокусується на отриманні інформації та формуванні звітів для локального координатора;

— Призначений координатор (DR): виконує роль проміжної ланки, агрегуючи статуси та забезпечуючи надійність у локальному сегменті.

Програмний код точки входу в систему автоматично сканує доступні мережеві інтерфейси операційної системи, створює необхідну кількість сокетів для кожного каналу зв'язку та прив'язує їх до відповідних адрес. Після цього програма переходить у стан очікування команд або безпосереднього виконання функцій згідно з обраною роллю.

Основним завданням експерименту є гарантована передача файлу від центрального джерела до групи отримувачів. У межах обраної топології ключову роль відіграє вузол-координатор. Його програмний модуль реалізує основний механізм забезпечення надійності: він безперервно прослуховує мережевий канал, приймає вхідні кадри даних, проводить їх розбір (парсинг) для перевірки цілісності та ідентифікації типу пакета, а також веде облік отриманих сегментів. Окрім стандартного прийому, цей вузол накопичує інформацію про успішність доставки від інших приймачів у своїй зоні, що дозволяє йому оперативно виконувати повторну трансляцію у разі виявлення втрат, не навантажуючи центральний канал зв'язку з відправником.

Звичайні приймачі в системі працюють за спрощеною, але критично важливою схемою. Оскільки в поточній реалізації координатор призначається вручну, вузли-отримувачі заздалегідь володіють інформацією про його мережеву адресу. При старті вони відправляють координатору спеціальне службове повідомлення, сигналізуючи про своє входження до його локальної області. Після реєстрації координатор починає очікувати від них періодичні підтвердження успішного отримання кожного сегмента даних.

Вузол-відправник бере на себе функції підготовки та сегментації контенту. Програма відкриває цільовий файл у режимі читання бінарних даних і розбиває його на пакети фіксованого розміру, враховуючи необхідне місце для службового заголовка протоколу. Після формування черги пакетів починається їх послідовна широкомовна трансляція в мережу.

Паралельно з відправкою нових даних, модуль джерела безперервно аналізує вхідний потік на наявність підтверджень від координаторів. Якщо алгоритм виявляє, що певний сегмент не був підтверджений протягом встановленого часу або був отриманий запит на його відновлення, система ініціює повторну передачу відповідного фрагмента. Такий зациклений процес триває до моменту отримання остаточного підтвердження про успішну верифікацію всього обсягу переданої інформації всіма сегментами мережі.

3.3 Емпіричні результати апробації та їх оцінка

Після успішної ініціалізації всіх вузлів та завершення етапу реєстрації клієнтів у локальній зоні (факт якої фіксувався у системних журналах координатора), головний відправник розпочав трансляцію даних. У результаті контрольного запуску тестовий файл обсягом 3 кілобайти був успішно, без порушення цілісності та послідовності, доставлений усім цільовим вузлам у мережі. Аналіз системного виводу підтвердив, що протокол коректно відпрацював цикли отримання датаграм, відправки підтверджень та ініціалізації наступних транзакцій, рис. 3.4 – 3.8.

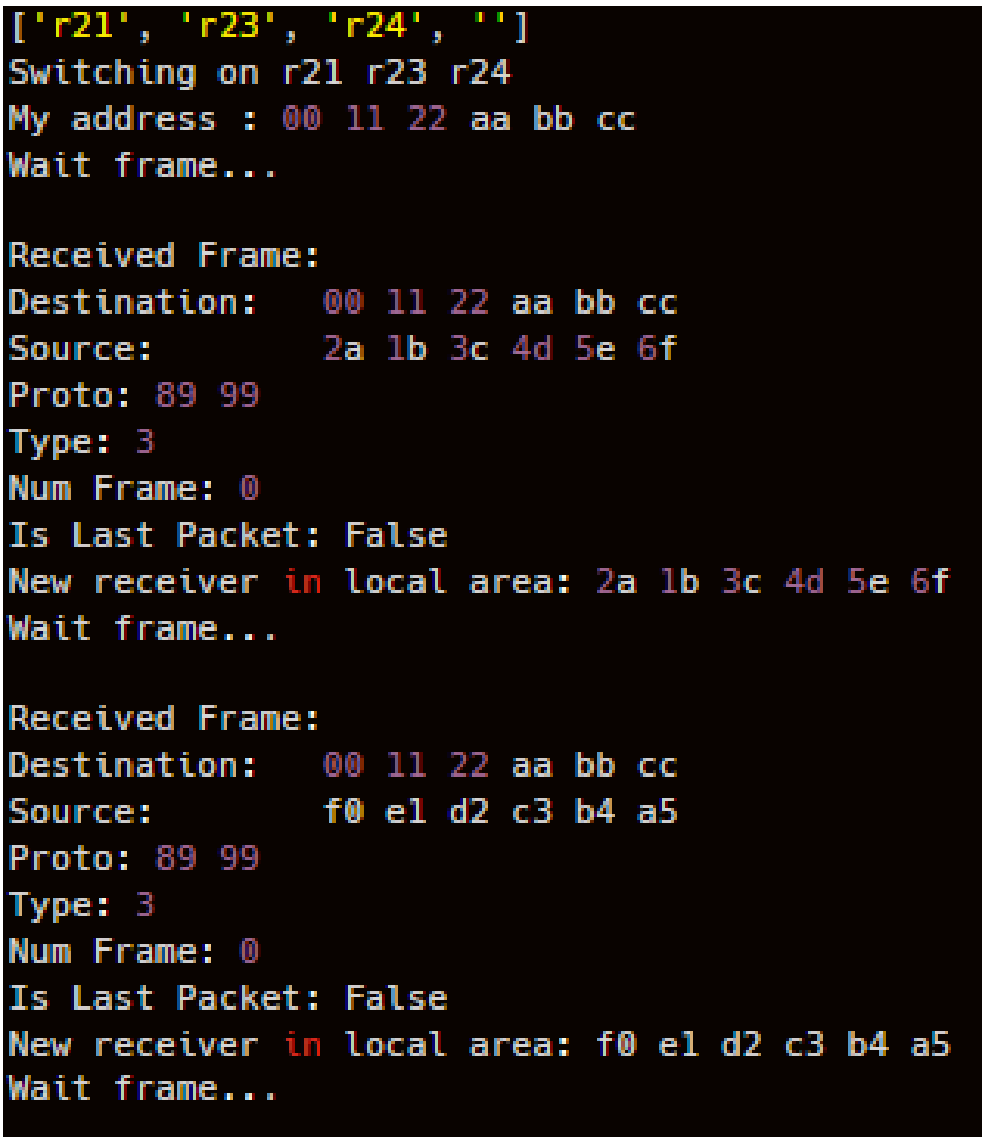


Рисунок 3.4 — Фіксація реєстраційних запитів від кінцевих вузлів на боці координатора

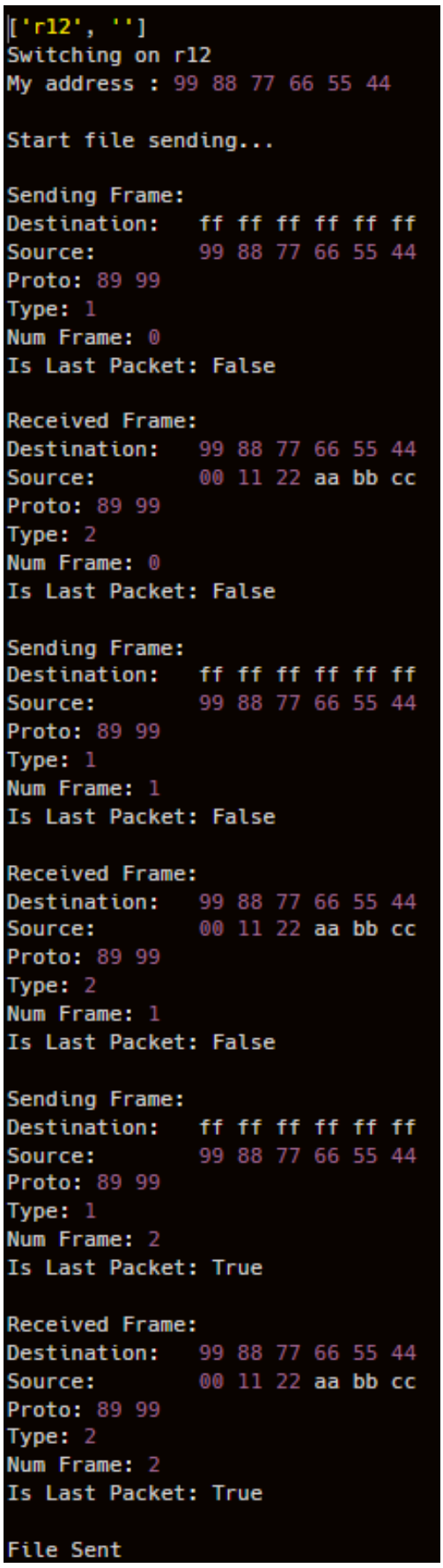


Рисунок 3.5 — Журналювання процесу сегментації та трансляції даних відправником

```
Received Frame:
Destination:   ff ff ff ff ff ff
Source:        99 88 77 66 55 44
Proto: 89 99
Type: 1
Num Frame: 0
Is Last Packet: False
Wait frame...

Received Frame:
Destination:   00 11 22 aa bb cc
Source:        f0 e1 d2 c3 b4 a5
Proto: 89 99
Type: 2
Num Frame: 0
Is Last Packet: False
Wait frame...

Received Frame:
Destination:   00 11 22 aa bb cc
Source:        2a 1b 3c 4d 5e 6f
Proto: 89 99
Type: 2
Num Frame: 0
Is Last Packet: False

Sending Frame:
Destination:   99 88 77 66 55 44
Source:        00 11 22 aa bb cc
Proto: 89 99
Type: 2
Num Frame: 0
Is Last Packet: False
Wait frame...
```

Рисунок 3.6 — Аналіз логів отримання даних та генерації підтверджень на R2

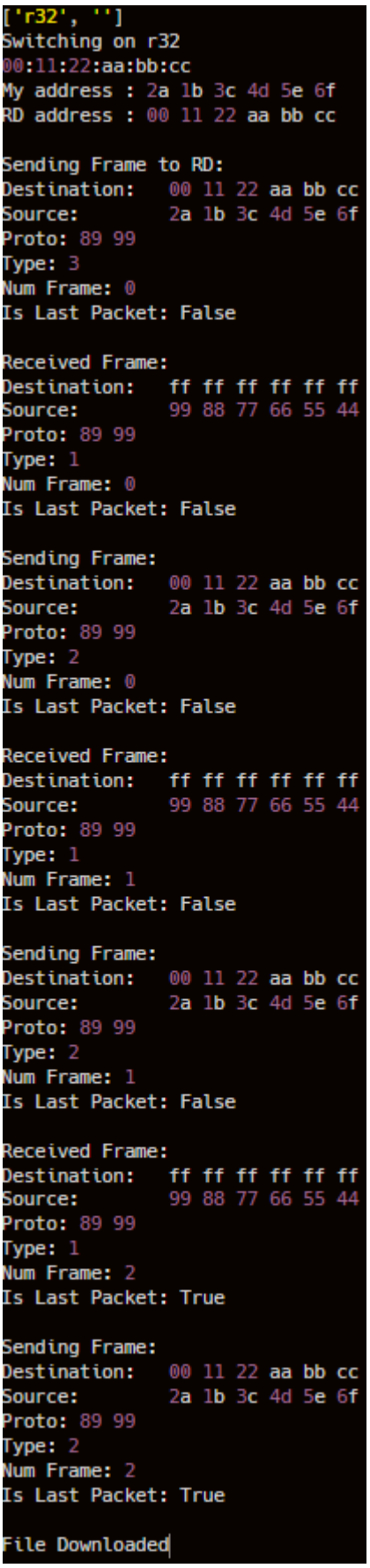


Рисунок 3.7 — Аналіз логів отримання даних та генерації підтверджень на R3

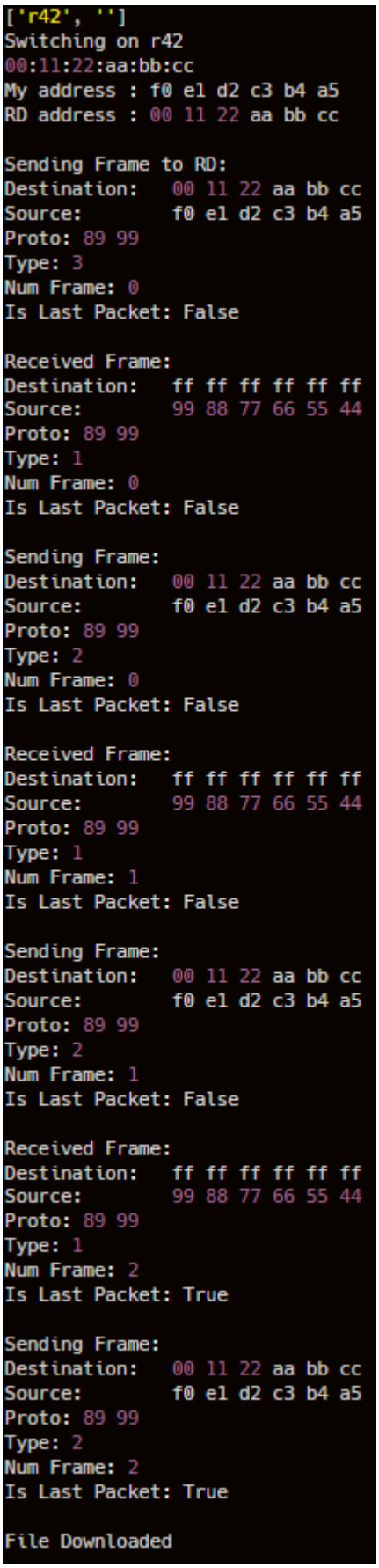


Рисунок 3.8 — Аналіз логів отримання даних та генерації підтверджень на R4

Для об’єктивної оцінки продуктивності розробленого алгоритму (АТПБЗ) було проведено порівняльний аналіз обсягів згенерованого службового та корисного трафіку з еталонним протоколом гарантованої доставки TCP. Ключовою метрикою виступала загальна кількість пакетів, надісланих та прийнятих кожним вузлом під час повної передачі ідентичного файлу. Зведені результати моніторингу мережевої активності наведено в таблиці 3.1.

Таблиця 3.1 — Порівняльний аналіз кількості надісланих та прийнятих пакетів у розрізі мережевих вузлів

Роль у топології (Вузол)	Традиційний стек (TCP Unicast)		Пропонований L2-протокол	
	Генерація (Tx)	Обробка (Rx)	Генерація (Tx)	Обробка (Rx)
S	165	142	50	50
R2	48	62	50	152
R3	48	55	51	50
R4	48	55	51	50

Емпіричні дані переконливо демонструють архітектурні відмінності протоколів. При використанні класичного TCP відправник змушений підтримувати індивідуальні сесії з кожним клієнтом, що призводить до відправки 165 пакетів та обробки 142 підтверджень на боці джерела. Відповідно, при масштабуванні мережі обсяг трафіку зростатиме суворо лінійно, що в умовах високонавантажених систем неминуче призведе до перевантаження магістрального каналу.

Натомість запропонований протокол АТПБЗ кардинально оптимізує навантаження на кореневий сегмент мережі. Завдяки ієрархічній архітектурі та широкомовній природі передачі, головному вузлу знадобилося надіслати лише 50 пакетів і обробити 50 зведених підтверджень. Основне навантаження з

агрегації зворотного зв'язку (152 отримані пакети) було зміщено на рівень локального координатора. Таким чином, результати підтверджують, що розроблений протокол забезпечує рівномірний розподіл навантаження, суттєво скорочує загальний обсяг трафіку в магістралі, а його пропускна здатність залежить не від загальної кількості клієнтів, а виключно від кількості локальних зон та ефективності їхніх транзитних вузлів.

ВИСНОВКИ

Проведене дослідження специфіки та сучасного стану предметної області багатоадресного обміну інформацією в комп’ютерних мережах дозволило зробити наступні висновки:

1. Аналіз існуючих технологій показав, що класичні механізми на основі позитивних підтверджень є неефективними для великих мереж через проблему перевантаження джерела. Більш перспективними є деревоподібні механізми та використання негативних підтверджень, які позбавляють головний сервер необхідності постійно перераховувати топологію та обробляти лавину квитанцій.

2. Розроблено архітектуру Адаптивного транспортного протоколу багатоадресного зв'язку, який забезпечує гарантовану доставку даних завдяки поділу отримувачів на локальні сегменти. Впровадження виділених приймачів (координаторів), які самостійно агрегують статистику та виконують повторну передачу втрачених фрагментів у своїй зоні, вирішує проблему активного перевантаження підтвердженнями.

3. Для тестування запропонованої архітектури було розгорнуто імітаційний стенд на платформі віртуалізації VMWare ESXi. Програмна логіка протоколу була успішно реалізована мовою Python з використанням сирих сокетів (raw sockets), що дозволило забезпечити повний контроль над формуванням мережевих пакетів.

4. Результати порівняльного експерименту довели високу ефективність розробленого рішення. Під час передачі ідентичного тестового файлу класичний стек TCP згенерував 165 пакетів на боці відправника, тоді як запропонований протокол АТПБЗ потребував відправки лише 50 пакетів. Основне навантаження з обробки підтверджень було успішно зміщено на рівень локального координатора.

5. Отже, запропонована система забезпечує рівномірний розподіл навантаження та суттєво скорочує обсяг службового трафіку в магістралі, роблячи її пропускну здатність незалежною від загальної кількості кінцевих клієнтів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Kurose J. F., Ross K. W. Computer Networking: A Top-Down Approach. 8th ed. Pearson, 2021. 832 p.
2. Tanenbaum A. S., Feamster N., Wetherall D. J. Computer Networks. 6th ed. Pearson, 2021. 944 p.
3. Оліфер В. Г., Оліфер Н. А. Комп'ютерні мережі. Принципи, технології, протоколи. 5-те вид. СПб.: Пітер, 2016. 992 с.
4. Peterson L. L., Davie B. S. Computer Networks: A Systems Approach. 5th ed. Morgan Kaufmann, 2011. 920 p.
5. Fall K. R., Stevens W. R. TCP/IP Illustrated, Volume 1: The Protocols. 2nd ed. Addison-Wesley Professional, 2011. 1056 p.
6. Кліменко В. О. Комп'ютерні мережі: Навчальний посібник. Київ: Центр учбової літератури, 2012. 352 с.
7. Postel J. RFC 793. Transmission Control Protocol. DARPA Internet Program Protocol Specification. IETF, 1981. URL: https://datatracker.ietf.org/doc/html/rfc793
8. Postel J. RFC 768. User Datagram Protocol. IETF, 1980. URL: https://datatracker.ietf.org/doc/html/rfc768
9. Deering S. RFC 1112. Host Extensions for IP Multicasting. IETF, 1989. URL: https://datatracker.ietf.org/doc/html/rfc1112
10. Mankin A., Romanow A., Bradner S., Paxson V. RFC 2357. IETF Criteria for Evaluating Reliable Multicast Transport and Application Protocols. IETF, 1998. URL: https://datatracker.ietf.org/doc/html/rfc2357
11. Speakman T., Crowcroft J., Gemmell J. et al. RFC 3208. PGM Reliable Transport Protocol Specification. IETF, 2001. URL: https://datatracker.ietf.org/doc/html/rfc3208

12. Whetten B., Vicisano L., Kermode R. et al. RFC 3048. Reliable Multicast Transport Building Blocks for One-to-Many Bulk-Data Transfer. IETF, 2001. URL: https://datatracker.ietf.org/doc/html/rfc3048

13. Handley M., Floyd S., Whetten B. et al. RFC 2887. The Reliable Multicast Design Space for Bulk Data Transfer. IETF, 2000. URL: https://datatracker.ietf.org/doc/html/rfc2887

14. Adamson B., Bormann C., Handley M., Macker J. RFC 3940. Negative-acknowledgment (NACK)-Oriented Reliable Multicast (NORM) Protocol. IETF, 2004. URL: https://datatracker.ietf.org/doc/html/rfc3940

15. Luby M., Vicisano L., Gemmell J. et al. RFC 5052. Forward Error Correction (FEC) Building Block. IETF, 2007. URL: https://datatracker.ietf.org/doc/html/rfc5052

16. Floyd S., Jacobson V., Liu C., McCanne S., Zhang L. A Reliable Multicast Framework for Light-weight Sessions and Application Level Framing. *IEEE/ACM Transactions on Networking*, 1997. Vol. 5, No. 6. P. 784-803.

17. Pingali S., Kurose J., Towsley D. A comparison of sender-initiated and receiver-initiated reliable multicast protocols. *IEEE Journal on Selected Areas in Communications*, 1997. Vol. 15, No. 3. P. 398-406.

18. Paul S., Sabnani K. K., Lin J. C.-H., Bhattacharyya S. Reliable multicast transport protocol (RMTP). *IEEE Journal on Selected Areas in Communications*, 1997. Vol. 15, No. 3. P. 407-421.

19. Levine B. N., Garcia-Luna-Aceves J. J. A comparison of reliable multicast protocols. *Multimedia Systems*, 1998. Vol. 6, No. 5. P. 334-346.

20. Rhodes B., Goerzen J. Foundations of Python Network Programming. 3rd ed. Apress, 2014. 360 p. (Джерело, що обґрунтовує роботу з сокетами в Python).

21. Donahoo M. J., Calvert K. L. TCP/IP Sockets in C: Practical Guide for Programmers (Concepts applicable to raw sockets). 2nd ed. Morgan Kaufmann, 2009. 224 p.

22. VMware vSphere Documentation. Virtual Networking Concepts. VMware, Inc. URL: https://docs.vmware.com/en/VMware-vSphere/index.html (Для обґрунтування побудови віртуального стенду на базі VMWare ESXi).

Додаток А

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

(Презентація)