Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет комп'ютерної інженерії та управління
(повна назва)

Кафедра електронних обчислювальних машин
(повна назва)

АТЕСТАЦІЙНА РОБОТА
Пояснювальна записка

Рівень вищої освіти другий (магістерський)

Методи та алгоритми оптимізації мережевого трафіку у Cloud–інфраструктурі

(тема)

Виконав:
студент II курсу, групи КСМм-19–1
Євтушенко А.В.
(прізвище, ініціали)

Спеціальність
123 – Комп'ютерна інженерія
(код і повна назва спеціальності)

Тип програми освітньо-професійна
(освітньо–професійна або освітньо–наукова)

Освітня програма
Комп'ютерні системи та мережі
(повна назва освітньої програми)

Керівник: доц. Іванісенко І.М.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри ЕОМ _____ Коваленко А.А.
(підпис) (прізвище, ініціали)

2020 р.

Харківський національний університет радіоелектроніки

Факультет комп’ютерної інженерії та управління

Кафедра електронних обчислювальних машин

Рівень вищої освіти другий (магістерський)

Спеціальність 123 – Комп’ютерна інженерія
(код і повна назва)

Тип програми освітньо-професійна
(освітньо–професійна або освітньо–наукова)

Освітня програма Комп’ютерні системи та мережі
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(Підпис)
"____" _____ 2020 р.

ЗАВДАННЯ

НА АТЕСТАЦІЙНУ РОБОТУ

студентові Євтушенку Антону Вячеславовичу
(Прізвище, ім'я, по батькові)

1. Тема роботи Методи та алгоритми оптимізації мережевого трафіку у Cloud–інфраструктурі.

затверджена наказом по університету від " 30 " жовтня 2020 р. № 1487 Ст

2. Термін подання студентом роботи 14 грудня 2020 р.

3. Вхідні дані до роботи

Загальна структура обчислювальних систем, хмарна інфраструктура, структура WAN мережі, базова модель зв'язку відкритих систем OSI, принципи роботи систем моніторингу комп'ютерних мереж, ПЗ GNS 3, маршрутизатор Cisco 2811, комутатори Cisco 3750, 3650, мережеве програмне забезпечення Riverbed Steelhead, Wanos, Traffic Squsee. Загальна структура та принципи роботи мережевих протоколів, система моніторингу Nagios різновиди топологій міських мереж, особливості їх побудови,

.

4. Перелік питань, що потрібно опрацювати в роботі

Вступ.

Аналіз предметної області, літератури та постановка задачі.

Обґрунтування оптимізації та модернізації WAN мережі, як компоненту хмарної інфраструктури.

Експериментальна частина модернізації WAN мережі.

Розрахункова частина.

Висновки. Додаток.

.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів)

Презентація Powerpoint 15 слайдів.

6. Консультанти розділів роботи (заповнюється за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначку консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№ п./ п.	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз літератури за темою роботи	03.11.20–06.11.20	
2	Постановка мети та задач	07.11.20–11.11.20	
3	Аналіз засобів оптимізації WAN мережі	12.11.20–18.11.20	
4	Обгрунтування оптимізації		
	та модернізації мережі	19.11.20–24.11.20	
5	Експериментальна частина	25.11.20–29.11.20	
6	Розрахункова частина	30.11.20–02.12.20	
7	Підготовка пояснювальної записки	03.12.20–06.12.20	
8	Розробка презентації та доповіді	07.12.20–13.12.20	
9	Подача роботи у ЕК	14.12.2020	

Дата видачі завдання 02.11. 2020 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Іванісенко І.М.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка атестаційної роботи: 116 с., 55 рисунків, 30 таблиць, 22 джерела.

СИСТЕМА МОНІТОРИНГУ МЕРЕЖІ, ХМАРНІ ТЕХНОЛОГІЇ, мережевий ДІАГНОСТИКА, TRAFFIC ANALYZER, КОЛІЗІЇ, WAN МЕРЕЖІ, ПРОГРАМИ МОНГІТОРІНГУ ТРАФІКА, МОНІТОРИНГ ПРОТОКОЛІВ

Метою атестаційної роботи є розгляд модернізації комп’ютерної мережі у результаті оптимізації трафіка WAN мереж, як важливого компоненту хмарної інфраструктури (cloud architecture) й отримання практично повної відмовостійкості. У вступі розкривається актуальність обраної тими атестаційної роботи, також ставиться мета й визначаються головні задачі.

У ході виконання атестаційної роботи було розглянуто 3 сценарії з оптимізації трафіка WAN мереж, із зазначенням переваг і недоліків кожному. В експериментальній частині були зроблені повні налаштування та запуск кожного з цих сценаріїв із зазначенням повного списку дій і прикладів. Після чого було проведено тестування з кожного окремо і вибрано оптимальне мережеве програмне забезпечення з оптимізації трафіка.

ABSTRACT

Master's thesis: 116 pages, 55 figures, 30 tables, 22 sources.

NETWORK MONITORING SYSTEM, CLOUD TECHNOLOGIES, NETWORK DIAGNOSTICS, NETWORK TRAFFIC ANALYZER, COLLISION, WAN, TRAFFIC MONITORING SOFTWARE, MONITORING PROTOCOLS

The goal of the attestation work is to consider the modernization of the computer network as a result of optimizing the traffic of WAN networks, as an important component of cloud infrastructure, and complete fault tolerance. The introduction reveals the relevance of the chosen theme of the graduation project also sets goals and objectives are defined to achieve these goals.

During the attestation work, 3 scenarios for optimizing the traffic of WAN networks were considered, indicating the advantages and disadvantages of each. In the experimental part were produced complete installation and configuration of each of these projects with the full list of actions and examples. After which it was produced for each test separately and selected the best network software to optimize traffic.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

WAN – глобальна обчислювальна мережа (англ., Wide Area Network)

VLAN – віртуальна локальна обчислювальна мережа (англ., Virtual Local Area Network)

CDN – географічно розподілена мережева інфраструктура (англ., Content Delivery Network)

SaaS – програмне забезпечення як послуга (англ., Software as a Service)

IaaS – інфраструктура як послуга (англ., Infrastructure as a Service)

PaaS – платформа як послуга (англ., Platform as a Service)

NAT – перетворення мережевих адрес (англ., Network Address Translation)

QoS – якість обслуговування (англ., Quality of Sevice)

ЦОД – центр обробки даних

ОС – Операційна система

Ethernet – сімейство протоколів стандарту IEEE 802.3

IEEE – Інститут інженерів з електротехніки та електроніки (англ., Institute of Electrical and Electronic Engineers)

ПЗ – програмне забезпечення

СУБД – система управління базами даних

Утилізація – завантаження каналу або сегменту

ЕОМ – електронно–обчислювальна машина

ВСТУП

Сьогодні світ неможливо уявити без постійно зростаючих хмарних мереж. Все хмарні сервіси, які використовуються як споживачами, так і організаціями, управляються за допомогою мереж центрів обробки даних (ЦОД), розкиданих по всьому світу. ЦОДи і оператори хмарних послуг працюють з вендорами з метою створення більш масштабованих рішень, які можуть відповідати постійно зростаючим потребам численних систем зберігання даних і серверів.

Незважаючи на те, що «хмарні обчислення» як термін набули розголосу лише в 2007 році, вони мають досить довгу історію. Практично всі технології, які сьогодні входять до складу хмарної парадигми, існували і раніше, однак на ринку не було пропозицій, які б об'єднували перспективні технології в єдиному комерційно привабливому вирішенні. І тільки в останні чотири–п'ять років з'явилися публічні хмарні сервіси, завдяки яким ці технології стали, з одного боку, гранично доступні розробнику, а з іншого – гранично зрозумілі для бізнесу.

Хмарні обчислення (англ. Cloudcomputing), – це модель надання повсюдного і зручного мережевого доступу, в міру необхідності, до загальної сукупності конфігуруються обчислювальних ресурсів (наприклад, мереж, серверів, систем зберігання, додатків і сервісів), які можуть бути швидко надані і звільнені з мінімальними зусиллями з управління і необхідністю взаємодії з провайдером послуг (сервіс–провайдером) [1–2].

Хмарні обчислення (англ. Cloud computing), – це модель забезпечення повсюдного та зручного мережевого доступу на вимогу до загального пулу (англ. Pool) конфігуруються обчислювальних ресурсів (наприклад, мереж передачі даних, серверів, пристроїв зберігання даних, додатків і сервісів – як разом , так і окремо), які можуть бути оперативно надані та звільнені з міні-

мальними експлуатаційними витратами і / або зверненнями до провайдера [3].

На сьогоднішній день, як маленькі, так і великі компанії прагнуть підвищити свій прибуток. Перш за все, це полягає в якісно підібраному персоналі, а найголовніше в підході до тієї чи іншої задачі. У плані підходу керівники прагнуть максимально поліпшити якість сфери своєї діяльності.

У атестаційній роботі необхідно здійснити модернізацію WAN мережі (як найважливішого компонента хмари) і виконати оптимізацію трафіка в ній за допомогою мережевого програмного забезпечення.

На даний момент ця тема дуже актуальна, в зв'язку з глобальним прогресом у сфері інформаційних технологій і динамічним зростанням самого підприємства. Оскільки, чим більше підприємство відповідно підвищуються витрати на телекомунікації. Як таким вибором модернізації мережі, стало встановлення нового додаткового мережевого обладнання Cisco, щоб максимально досягти повної відмовостійкості мережі в зв'язку з розширенням компанії та навантаження на наявну мережу. Також планується виконати оптимізацію трафіка наявних мереж між головним офісом і іншими філіями по Україні, оскільки нинішніх швидкостей вже не досить для повноцінної швидкісної роботи.

Все вище викладене показує, що обрана тема атестаційної роботи актуальна й важлива.

1 АНАЛІЗ ЛІТЕРАТУРИ І ПОСТАНОВКА ЗАДАЧІ

1.1 Структура і сутність хмарних обчислень

Хмарні обчислення (англ. Cloud computing) – технологія розподіленої обробки даних, в якій комп'ютерні ресурси і потужності надаються користувачеві як інтернет–сервіс. Хмарний сервіс являє собою особливу клієнт–серверну технологію – використання клієнтом ресурсів (процесорний час, оперативна пам'ять, дисковий простір, мережеві канали, спеціалізовані контролери, програмне забезпечення і т. д.) Групи серверів в мережі, взаємодіючих таким чином, що:

- для клієнта вся група виглядає як єдиний віртуальний сервер;
- клієнт може прозоро і з високою гнучкістю змінювати обсяги споживаних ресурсів в разі зміни своїх потреб (збільшувати / зменшувати потужність сервера з відповідною зміною оплати за нього).

При цьому наявність кількох джерел використовуваних ресурсів, з одного боку, дозволяє підвищувати доступність системи клієнт–сервер за рахунок можливості масштабування при підвищенні навантаження (збільшення кількості використовуваних джерел даного ресурсу паралельно зі збільшенням потреби в ньому). А з іншого – знижує ризик непрацездатності віртуального сервера в разі виходу з ладу будь–якого з серверів, що входять в групу, яка обслуговує даного клієнта, так як замість вийшов з ладу сервера можливо автоматичне перепідключення віртуального сервера до ресурсів іншого (резервного) сервера.

1.1.1 Технології, які підготували грунт для «хмар»

Віртуалізація – хоча і важлива, але не єдина технологія, що лежить в основі сучасної хмарної парадигми. Серед інших технологічних трендів, які

послужили прелюдією до сучасних хмарних обчислень, можна назвати сервіс–орієнтовану архітектуру (Service–Oriented Architecture, SOA), надання додатків в режимі послуг (Application Service Provider, ASP), ITIL / ITSM і ін. Деякі з цих концепцій укладали в собі цілком конкурують ні технологічні підходи, в той час як інші переважно вживалися в маркетингових цілях (наприклад, ASP). На рисунку 1.1 хмарні обчислення зображені, як результат синтезу цілого ряду технологій і підходів.

Рисунок 1.1 – Хмарні обчислення – результат синтезу цілого ряду технологій та підходів [3]

Повсюдне поширення високошвидкісних каналів інтернет–зв'язку зробило можливим інтенсивний обмін даними з комп'ютерами, що знаходяться в «хмарі». Дозрівання технологій Web 2.0 дозволило виконувати функціонально насичені веб–додатки безпосередньо у вікні веб–браузера, а не запускати їх на локальному комп'ютері або в локальній мережі. В якійсь мірі успіху хмарних обчислень сприяло також розвиток інтернет–сервісів, які надають

доступ до своїх даних за допомогою спеціальних програмних інтерфейсів (API). Дійсно, коли розробник створює додаток, яке обслуговує віддалених користувачів на основі даних з віддаленого джерела (наприклад, з Facebook), то цілком логічно, що і проміжний етап – обробка даних – також може здійснюватися на віддаленій хмарної майданчику.

Хмарні обчислення увібрали в себе багато ідей з попередніх концепцій, і тому спочатку вони носять більш різносторонній характер: їх можна розуміти і як технічну парадигму, і як маркетинговий термін, і як перспективний напрямок для НДДКР і академічних досліджень. По суті, в хмарні обчислення вклали всі ті ідеї, які накопичувалися в галузі протягом попередніх півтора десятиліть.

Одним з найбільш значущих подій в даній області була поява Salesforce.com в 1999 році. Дана компанія стала першою компанією надала доступ до свого додатком через сайт, по суті, дана компанія стала першою компанією надала своє програмне забезпечення за принципом – програмне забезпечення як сервіс (SaaS).

Наступним кроком стала розробка хмарного веб–сервісу компанією Amazon в 2002 році. Даний сервіс дозволяв зберігати, інформацію і робити обчислення.

У 2006, Amazon запустила сервіс під назвою Elastic Computecloud (EC2), як веб–сервіс який дозволяв його користувачам запускати свої власні додатки. Сервіси Amazon EC2 і Amazon S3 стали першими доступними сервісами хмарних обчислень.

Інша віха в розвиток хмарних обчислень сталася після створення компанією Google, платформи Google Apps для веб–додатків в бізнес секторі.

Розвиток апаратного забезпечення сприяло не стільки швидкому зростанню хмарних технологій, скільки доступності даної технології для малого бізнесу і індивідуальних осіб. Що стосується технічного прогресу, то значну роль в цьому зіграло створення багатоядерних процесорів і збільшення ємності накопичувачів інформації.

Датою відліку сучасної історії Cloud Computing став 2006 рік, коли компанія Amazon, яка вже на той момент була однією з найбільших, презентувала світу свою інфраструктуру Web–сервісів, яка була здатна забезпечити користувачеві не тільки хостинг, але і надати віддалені обчислювальні потужності клієнту. З тих пір розвиток «хмар» проходило стрімко, багато компаній перейшли на них при першій нагоді, а незабаром з'явилися і сервіси, що надають послуги розподілених обчислень своїм клієнтам.

На рисунку 1.2 представлені основні характеристики хмарних обчислень.

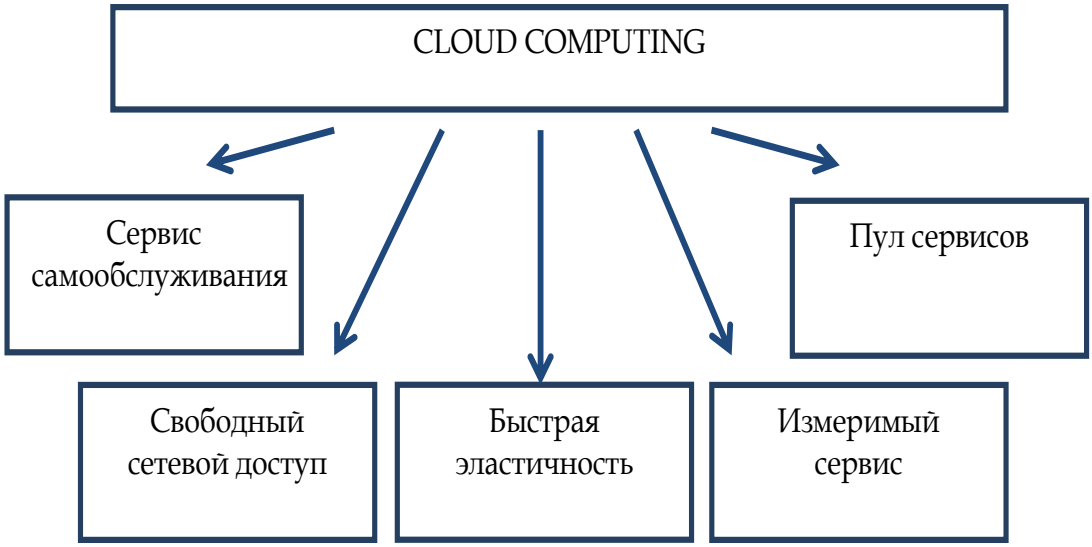

Рисунок 1.2 – Характеристики хмарних обчислень [3]

1.1.2 Класифікація хмарних ресурсів і їх основні характеристики

Виділяють кілька видів хмарних ресурсів (далі "хмар") (див. Рисунок 1.3):

1) Публічне хмара – доступ до ресурсів здійснюється будь–яким користувачем, що має підписку, з будь–якого місця, за умови наявності доступу в мережу Інтернет.

2) Приватне хмара – ресурси доступні лише обмеженому числу осіб (наприклад, співробітникам однієї компанії).

3) Громадське хмара – ресурси доступні кільком організаціям, що мають однакові потреби з точки зору інформаційних ресурсів.

4) Гібридний хмара – хмара, що складається з двох і більше хмар різних видів, наприклад, публічного і приватного. Кінцеві домашні користувачі або малий бізнес в основному використовують публічні хмари.

Рисунок 1.3 – Види хмарних ресурсів [4]

Моделі сервісу, що надається в хмарі Залежно від обраної підписки, користувач отримує той чи інший набір послуг. Виділяють три основні моделі обслуговування:

1. Програмне забезпечення як сервіс (Software as a Service, SaaS) – користувачеві надається доступ як до апаратних ресурсів, так і до додатка, що знаходиться на цих ресурсах. В даному випадку користувач позбувається необхідності зберігати програмне забезпечення на своїх ресурсах і робити їх резервну копію. Більш того, з'являється можливість отримати доступ до програми з будь–якого пристрою без попереднього встановлення. Потрібно лише доступ в Інтернет. При цьому користувач має мінімум прав для управління додатком.

2. Платформа як сервіс (Platform as a Service, PaaS) – даний вид підписки стоїть на рівень нижче попереднього. В даному випадку користувачеві надаються всі необхідні компоненти з хмари для розробки і експлуатації програмного забезпечення через Інтернет.

3. Інфраструктура як сервіс (Infrastructure as a Service, IaaS) – при даному виді угоди користувач отримує набір апаратних ресурсів, які він може використовувати відповідно до своїх потреб.

1.2 Транспортні мережі WAN – найважливіший компонент хмари

Сучасні організації стрімко впроваджують віртуалізацію, відео, хмарні обчислення і інші новітні технології для зниження капіталовкладень, підвищення операційної ефективності та забезпечення конкурентоспроможності. Пропозиції щодо оптимізації мережі WAN повинні розвиватися з урахуванням цих та інших тенденцій та ініціатив в сфері IT для бізнесу. В результаті з'явилися нові критерії і можливості, що змінюють уявлення про ідеальне рішення для оптимізації глобальної мережі.

Для чого б не використовувалося хмара – для пошуку, соцмереж або додатків корпоративного класу – воно являє собою розподілену обчислювальну модель на базі інфраструктури, що складається з безлічі серверів і ЦОДов, і в наш час вона затребувана повсюдно. Насправді, організації бачать у використанні хмарних сервісів величезну вигоду: їм набагато дешевше орендувати хмарний сервер, ніж будувати власний. Зростання попиту на хмарні додатки привів до експоненційному росту хмарних інфраструктур, який, в свою чергу суттєво змінив IT і мережеві архітектури. Розвиток технологій віртуалізації обчислювальних систем і СГД зіграло основну роль у розвитку хмари, але саме мережу, в кінцевому підсумку, формує його таким, яким ми звикли його сприймати.

Зростання хмари обумовлює потужний попит на мережі високої пропускної здатності, які з'єднують ЦОДи один з одним. Наприклад, один–єдиний

запит в соцмережах розподіляється на кілька сотень серверів, що знаходяться як всередині одного ЦОД а, так і в географічно розподілених ЦОДах (рисунок 1.4). Один з великих провайдерів послуг визначив, що один запит на 1 Кбайт споживає близько 930 Кбайт (майже 1 Мбайт) пропускної здатності з'єднання всередині ЦОДу.

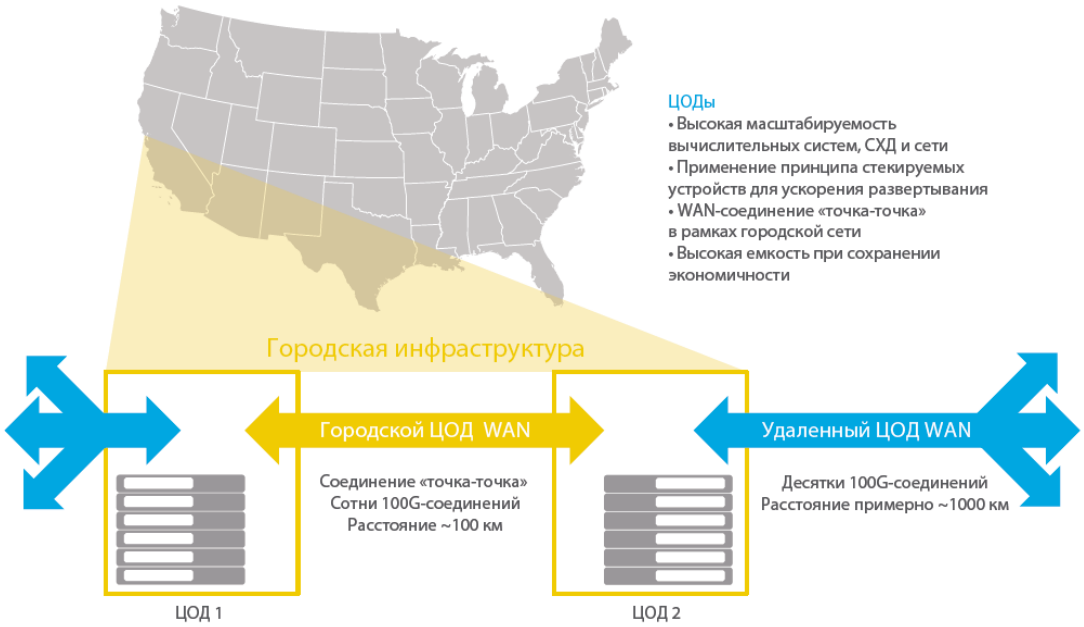

Рисунок 1.4 – Схема взаємодії 2–х ЦОДів [7]

Інший провайдер хмарних сервісів підрахував, що один пошуковий запит в середньому передається на відстань близько 2400 км, Притому велика частина цього маршруту доводиться на WAN–мережу, що з'єднує ЦОДи в хмарі, якщо порівнювати з публічної WAN–мережею, обробної запити користувача до хмари.

Щоб з'єднати між собою ці точки мережі, необхідна транспортна мережа WAN – життєво необхідний елемент будь–хмарної мережі. Провайдерам послуг необхідна можливість швидко масштабувати інфраструктуру, щоб відповідати зростаючим потребам до смуги пропускання, а також з'єднувати сервера, забезпечивши запас ємності для майбутніх потреб в пропускної здатності.

По–друге, самі по собі ЦОДи представляють собою величезні за площею об'єкти, що нагадують складські приміщення, і зазвичай знаходяться у віддаленій від великих міст місцевості, де власники ЦОД а можуть заощадити на вартості земельного фонду, нежитлової нерухомості і електроенергії. Ці центри потрібно підключити до густонаселених містах, які зазвичай знаходяться в декількох сотнях кілометрів від Цода. Також постачальники хмарних послуг будують ЦОДи в міській місцевості на відстані десятків кілометрів один від одного, щоб боротися з нестачею приміщень – і таким ЦОД ам потрібно рішення для організації широкосмугового з'єднання «точка–точка» з метою забезпечити відновлення після збоїв і стихійних лих, синхронізацію і балансування навантаження.

У той час як інфраструктуру обчислювальних систем і СГД давно будують відповідно до принципу стекирования пристроїв в стійках для забезпечення зручності при масштабуванні, простоти обслуговування і високої швидкості, фахівці з архітектури мереж тільки починають сповідувати цю філософію.

1.2.1 Прискорення і підключення хмарних сервісів

Хмарні моделі доставки (загальнодоступні, приватні та гібридні) перетворять сучасні компанії, забезпечуючи гнучку, масштабовану, стійку і економічну доставку IT–послуг. Оптимізація мережі WAN може допомогти двома способами: забезпечуючи безпечний оптимізований міст для передачі робочих навантажень і даних між головним центром обробки даних і хмарою (або допоміжним центром обробки даних), і шляхом оптимізації доставки розміщених в хмарі додатків і послуг співробітникам, де б вони не знаходилися . Функції і можливості, необхідні для виконання даних завдань, включають:

- можливість безпечного підключення до хмар, при якому передача всіх даних;
- здійснюється по надійно зашифрованому каналу;

- можливість прозорого підключення до хмар, при якому хмарна мережа стає природним розширенням мережі центру обробки даних підприємства (тим самим зводячи до мінімуму потребу в зміні конфігурації мережі і додатків);
- попередню інтеграцію з хмарами, що спрощує конфігурування підключень до популярних постачальникам хмарних сервісів, наприклад Amazon Web Services і Windows Azure, і розгортання кількох примірників оптимізаторів глобальної мережі.

Рішення для оптимізації глобальної мережі, що володіють такими можливостями, допомагають повною мірою використовувати загальнодоступний хмарний сервіс, дозволяючи підприємствам доповнити свої центри обробки даних революційною моделлю доставки з необмеженою пропускною спроможністю і гнучкою ефективністю.

1.3 Огляд мережевого ПЗ

1.3.1 Мережеве програмне забезпечення Wanos

Wanos є мережевим програмним забезпеченням, щоб скоротити витрати мережевого трафіка і прискорення швидкості мережі. Це досягається за рахунок підвищення ефективності передачі даних в результаті стиснення, дедуплікації, якість обслуговування і методик кешування. Віртуальні і фізичні пристрої Wanos розміщуються на шляху потоку мережевого трафіка. Трафік з інтернету або центр обробки даних зменшується, в результаті стиснення і видалення дублікатів закономірності в даних. Це дозволяє витрачати менше пропускної здатності, яка буде використовуватися протягом багатьох каналів глобальної мережі або VPN і зменшує витрати на трафік.

Висока доступність це метод проектування мережі реалізує отримувати високий рівень доступності мережі протягом будь–якого проміжку часу. Ви-

сока доступність або надмірність може бути досягнута шляхом запуску резервного копіювання мережевого кабелю йде паралельно пристрої Wanos.

При запуску RSTP пристрій Wanos з функцією Ядро розміщується між двома комутаторами, підключених через інтерфейси wanO із зовнішнім комутатором і lanO до внутрішнього комутатора. RSTP має обрати первинний і вторинний шлях, заснований на Port ID. Переконайтеся, що з'єднання Wanos знаходиться на більш низькому номері порту або змінити налаштування комутатора, щоб зробити з'єднанню основний шлях. RSTP повинен розмістити з'єднання резервного копіювання в альтернативному статус шлях і перемикається на резервний канал протягом декількох мілісекунд, після виявлення втрати. При запуску Per –VLAN RSTP висока доступність може бути досягнута за допомогою одного перемикача. У цьому випадку створюються два VLAN–а. Зовнішній VLAN з'єднує сегмент WAN і внутрішній VLAN з'єднує сегмент локальної мережі. На рисунку 1.5, маршрутизатор, Wanos wanO і резервний канал підключений до мережі VLAN 10. З боку LAN lanO Wanos, сегмент локальної мережі даних та резервний канал підключений до VLAN 20.

При нормальних умовах трафік проходить в режимі моста, пристрій Wanos розташоване між двома сегментами і резервний канал пасивно на холостому ходу. Коли перемикач виявляє збій на первинному шляху, резервний канал включається і відразу ж поміщає в стан з'єднання ..

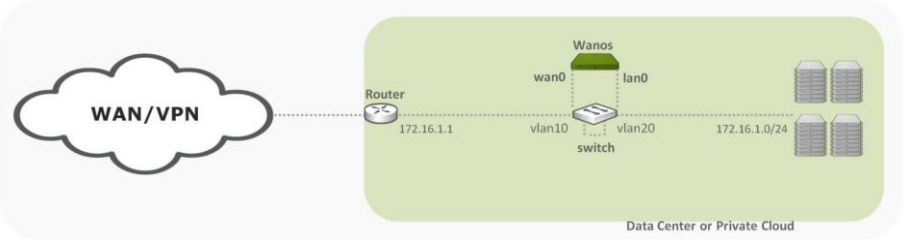

Рисунок 1.5– Приклад схеми з налаштуванням надмірності з одним комутатором

MultiSite дозволяє оптимізувати до кількох віддаленим офісам з пристроєм Wanos з функцією Край. Налаштування буде, однакова для 2х офісів. На головному пристрої є необхідною умовою для включення цієї функції.

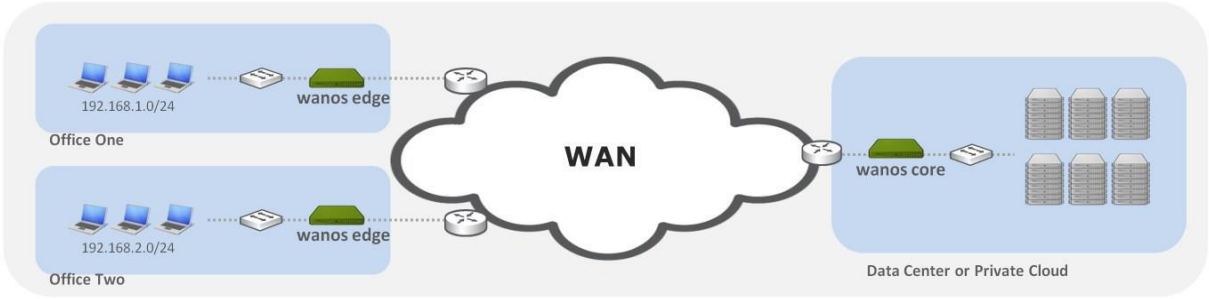

Рисунок 1.6 – Схема з двома віддаленими офісами

При використанні MultiSite, також забезпечити Wanos пристрій з функцією Ядро, повинно мати як найменш 2 Гб вільного місця.

Використання веб–інтерфейсу. Веб–інтерфейс буде відображати стан фізичних приладів і гіпервізора, які імітують фізичний стан мережі, також буде відображати стан порту Ethernet. Зелений порт вказує інтерфейс виявив мережевий кабель і з'єднаний. Чорний порт вказує, що кабель не підключений або стан невідомий. Ці показники можуть бути використані для ідентифікації ролі портів і корисні у визначенні того, якщо необхідно поміняти місцями кабелі.

Virtual Appliance Cabling. Після того, як віртуальні пристрої були створені наступним кроком є забезпечення правильного з'єднання кабелів. Перший Ethernet підключається до маршрутизатора WAN. Цей інтерфейс буде інтерфейсом WAN0. Другий Ethernet підключається до іншої частини мережі і називається інтерфейс LAN0. Разом ці два інтерфейси утворюють пару інтерфейс Inpath. Вони налаштовуються для проходження трафіка через міст і не мають IP– адреси При встановленні тестів можна також використовувати внутрішню мережу, гипервизор, сегменти локальної мережі або віртуальні мережі для створення необхідної топології мережі. Для виробничої роботи,

необхідний фізична мережева карта, мережевий інтерфейс для кожного інтерфейсу LAN0 WAN0.

Комутатори. Трафік між LAN0 і WAN0 з'єднується, і тому важливо, щоб прилад міст розгорнуто між стороною WAN мережі і на стороні LAN мережі. Коли міжмережеві екрани є частиною мережі, необхідно ставити прилад міст між WAN маршрутизатора і міжмережевого екрану. Зверніть увагу, що підключення обидва інтерфейсу до одного комутатора (рисунок 1.7) може викликати перемикач для блокування один з портів або гірше створити цикл моста, якщо STP відключена. Можна безпечно підключатися обидва порту до одного комутатора, тільки якщо порти комутатора налаштовані в різних VLAN 'S і працює Per VLAN Spanning – Tree Protocol (PVSTP)

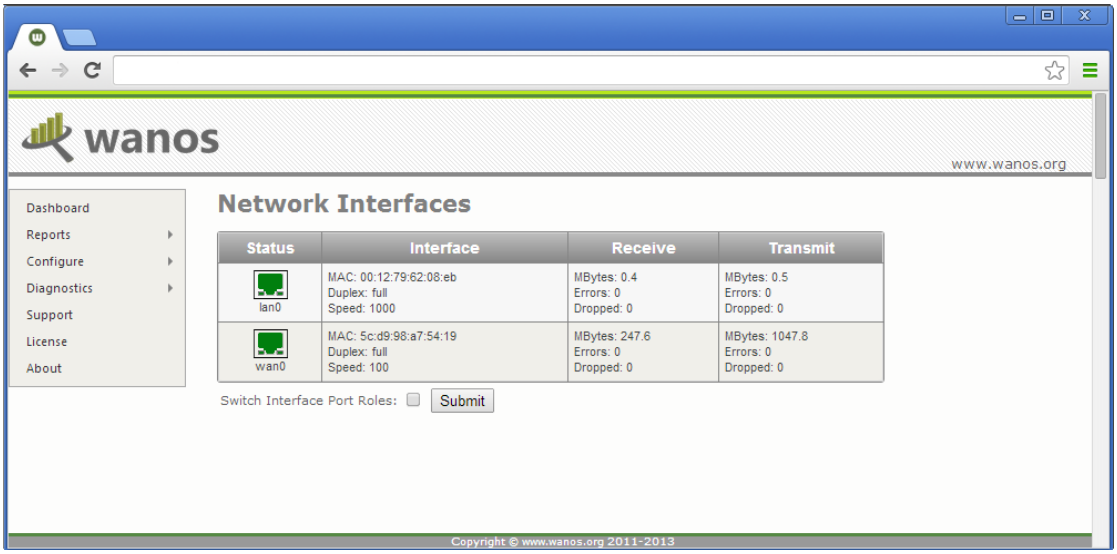

Рисунок 1.7 – Мережеві інтерфейси в веб інтерфейсі

З ряду еталонних тестів з реальними і реалістичних даних, wanos поставляє в середньому в 1.5 рази в холодні перекладів. Холодні переклади є першою спробою зменшити пропускну здатність через дедуплікації і стиснення. Гарячі трансфери є другими або наступні переклади одного й того ж або аналогічних даних і тут ми бачимо оптимізацію 4–10X. Наступний графік

ілюструє пропускну LAN від трьох різних випробувань передачі одного і того ж файлу 1.6 GB.

Перший контрольний тест без оптимізації (рисунок 1.8), як і очікувалося, максимальна швидкість дорівнює швидкості WAN на 1 Мбіт.

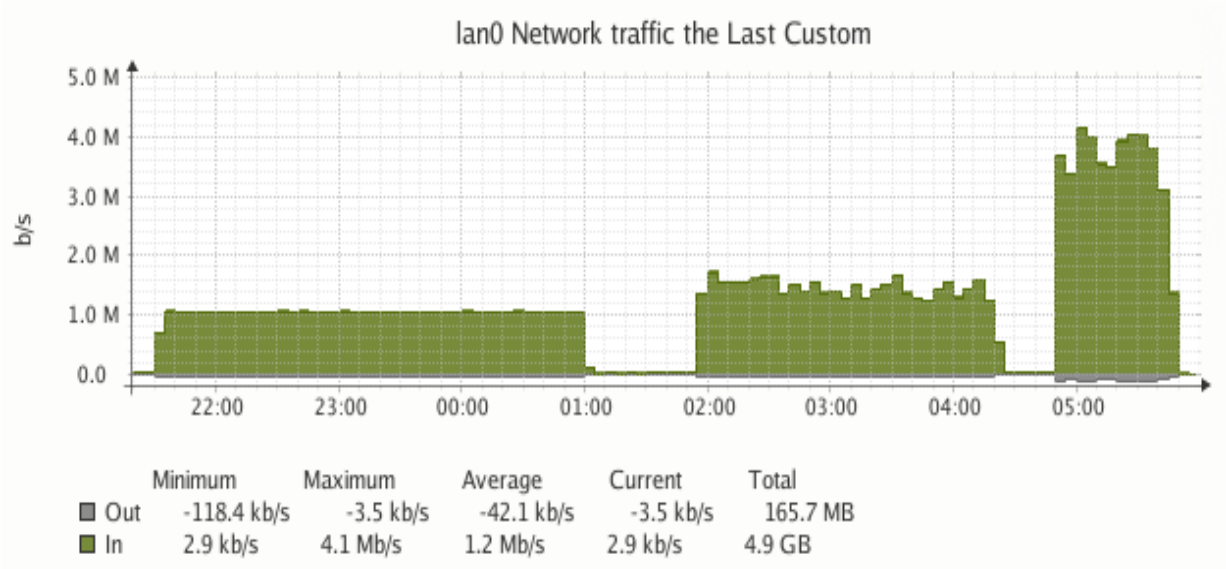

Рисунок 1.8 – Графік проходження трафіка без оптимізації

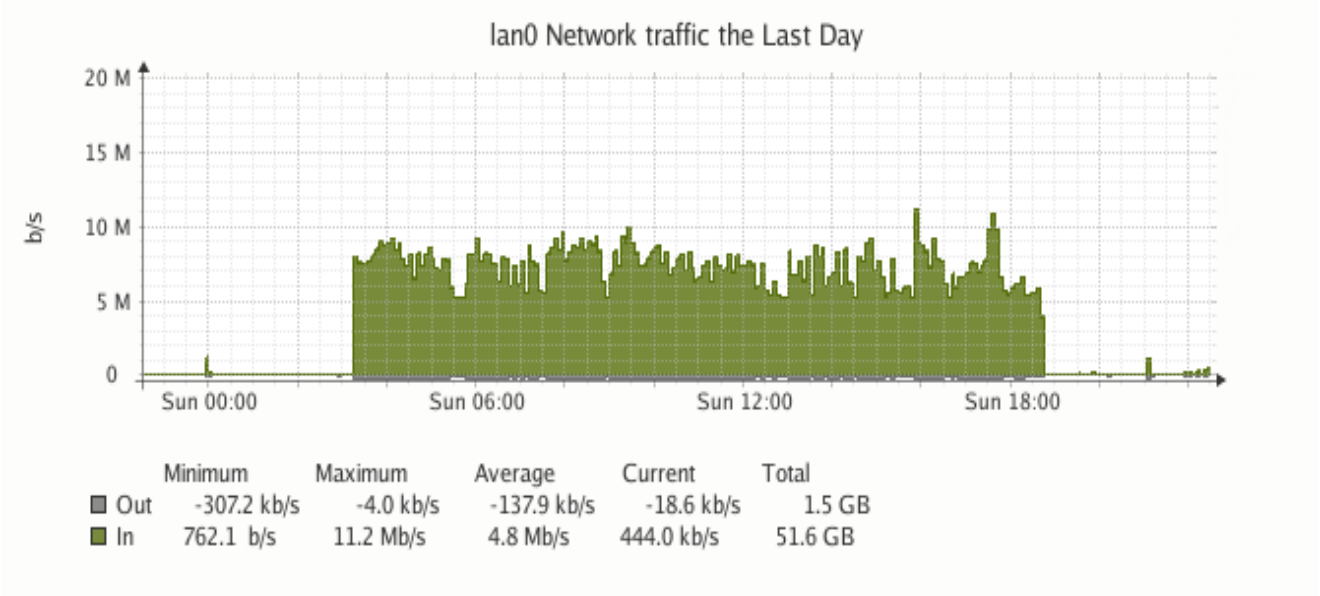

Рисунок 1.9 – Графік проходження трафіка в "холодному режимі"

Другий тест (рисунок 1.9) холодний передачі з оптимізації включений і, починаючи з чистого сховища. Це являє собою нижню межу, де сховище даних ще повинен бути не заповнено. Тут показана оптимізація 1.4–1.6x в середньому третій тест (рисунок 1.10) на графіку гарячої передачі тут показана оптимізація 4–10X.

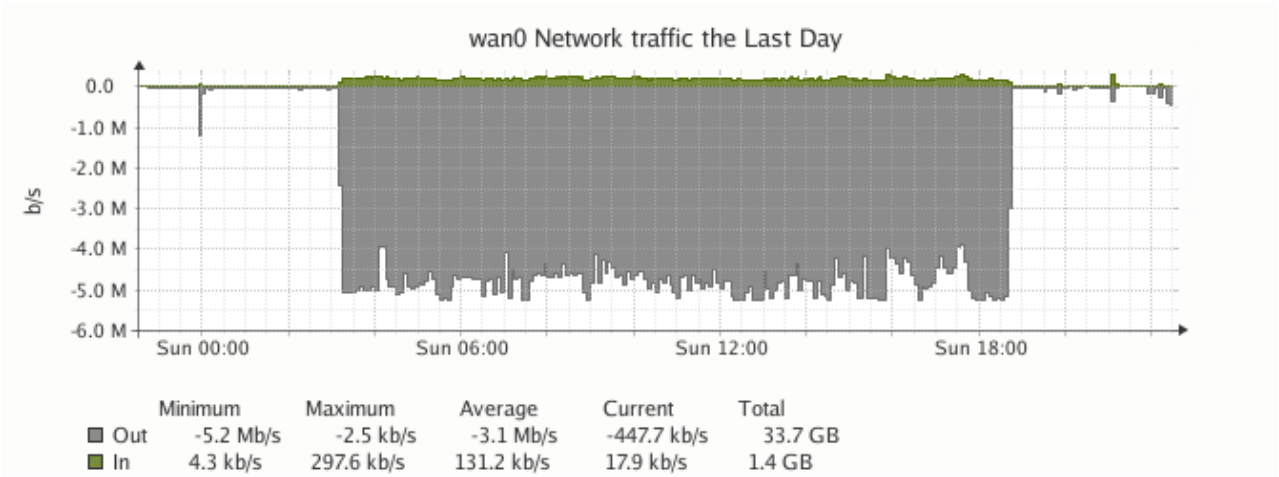

Рисунок 1.10 – Графік проходження трафіка в "гарячому режимі"

1.3.2 Мережеве програмне забезпечення TrafficSqueezer

Первісна версія прототипу написана в просторі користувача, але пізніше в версії TrafficSqueezer Альфа, дійсний механізм ядро буде підтримуватися в просторі ядра. Для управління ж, тонкими додатками користувача будуть надані, які забезпечує як командний рядок – (CLI) і Веб–графічний–Інтерфейс (GUI). Користувачі можуть налаштовувати операції в основних модулях TrafficSqueezer ядра. За допомогою цієї програми користувача також, можуть контролювати роботу і отримати статистику часу виконання, діаграми / графіки роботи TrafficSqueezer з відповідним трафіком, який вона оптимізує і відправляє через нього.

TrafficSqueezer може працювати практично у всіх можливих і часто використовуваних апаратних платформах. Проте, ось деякі спеціальні інструк-

ції і розуміння про картину можливих мінімальних вимог, які TrafficSqueezer типова установка може знадобитися. Мінімальні вимоги до апаратного: TrafficSqueezer є апаратно незалежним ядро Linux Kernel. Або, іншими словами він схожий на Linux Kernel IPv4, IPv6, подолання, і інших модулів основних мереж. Так що в цілому він не має будь–якого конкретного процесора або апаратного залежність. сумісність процесора CPU:

– Сумісність з Intel ® Celeron, Pentium, Xeon на основі i386, x64 процесорів.

Пропускна здатність в порівнянні з вибором процесора:

- 10 Мбіт – будь–яка Intel / AMD одиночний / соло процесор, процесор Intel Atom, двоядерний процесор, застаріли сервери і ПК можна використовувати;

- 20, 50 Mbps – будь–який процесор Intel / AMD Dual, процесори Quad Core, Intel Atom двоядерних процесорів, застаріли серверів і ПК можна використовувати.

TrafflcSqueezer є ядром Linux без платформи, залежить мережевий стек ядра. (Іншими словами, TrafficSqueezer є майже таким же, як Linux Kernel IPv4 мережевий стек, IPv6 мережевого стека і так далі. Таким чином TrafficSqueezer повинен підтримувати практично будь–які основі Linux дистрибутиви і були вибірково перевірені в нижче перерахованих дистрибутивів:

- Fedora;
- Ubuntu;
- Red Hat і Red Hat Enterprise Linux;
- Linux Mint;
- CentOS;
- Open SuSE;
- Knoppix;
- Debian GNU / Linux;

CDN – Content розподільні мережі (також згадується як мережу доставки контенту). Є інфраструктурою (може бути підмножина технологій Cloud Computing), яка дозволяє ділитися в першу чергу інтернет ресурсами (наприклад, веб–сторінки, файли, потоки аудіо / відео і т.д.) на різних точки і так далі до найближчих файлообмінних серверах CDN які розгорнуті в декількох регіонах. Користувач замість звернення до веб–сайту першоджерела, через первинний веб–сервер отримає прозоро що відображається інформацію через ці вузли CDN.

CDN є великий стратегією, щоб впоратися з будь–яким видом навантаження на мережу (в основному в Інтернеті), забезпечує високу продуктивність завантаження сторінки, незалежно від числа користувачів, які отримують доступ той же сайт одночасно. CDN технології (і постачальники послуг CDN) використовує більшість з відкритим вихідним кодом і фірмовим компонентом для побудови своєї інфраструктури. Іноді в складі оптимізації CDN, постачальники послуг також надають призначені для користувача оптимізовані маршрутизації, відображення DNS, також можуть використовувати P2P на основі технологій і так далі. Для побудови оптимізованих мереж CDN, постачальники послуг можуть використовувати TrafficSqueezer в рамках своєї інфраструктури в залежності від їх унікальних сценаріїв використання.

Банківські мережі, є унікальний набір вимог і вузьких місць. Як можна зрозуміти, що вони очікують високого рівня безпеки мережі передачі даних. Але в той же час їм потрібно, і необхідні високооптимізовані тунелі передачі даних. Зі збільшенням транзакцій електронної торгівлі, а також широким поширенням кредитних карт, операцій і розгортання з банкомату, вимоги WAN Optimization разом з (частіше в поєднанні з) вимоги безпеки різко зростають з року в рік.

Більшість банків матимуть свої безпечні виділені WAN мережі, але це є дуже дорогим варіантом. Але у них немає вибору, продовжують підписуватися на ці дорогі виділені канали глобальної мережі. Разом з тим вони мо-

жуть також використовувати в якості альтернативи захищені канали VPN для менш інтенсивних і менш часто використовувані віддалені мережі. Що означає, іноді є виділені WAN мережі, і для оптимізації техніка розгорнута і готова, а іноді вони потребують програмного рішення Acceleration WAN.

Це унікальний сценарій, при якому TrafficSqueezer оптимізації WAN заповнює обидва пробілу, вони можуть побудувати WAN пристрою і розгорнути їх, і так само в своїх існуючих end–machines / servers (кінцевих машинах / серверах), вони можуть встановити TrafficSqueezer. Вони можуть легко використовувати розширення хмари і GUI, що надаються через Doublefish Solutions акваріум. З акваріум, вони отримують GUI не тільки для TrafficSqueezer, вони отримують веб–графічний інтерфейс для всіх своїх систем Linux, їх систем кешування Linux Squid, і так далі. Хоча акваріум в даний час доступний в якості загального версії SaaS GUI, у випадку, подібному банківських і інших величезних розгортання вони можуть мати свої власні місцеві розгортання акваріум.

Citrix є одним з найпопулярніших і перевірених рішень для віртуалізації настільних систем. Організація (в основному середніх і великих підприємств) може розгорнути висококваліфіковані ефективні робочі столи віртуалізації для своїх співробітників. З якої він дає свободу, що співробітники тепер можуть здійснювати свій робочий стіл в будь–якому місці, так як він розгортається в серверах Citrix в їх Cloud Networks. Доведено, що це знижує часті витрати на інфраструктуру обслуговування, витрати на обладнання, а також проблеми з покупкою ліцензій на програмне забезпечення та установкою його в кожній системі співробітників.

Citrix також ефективний у разі частих оновлень програмного забезпечення / додатків і наявності дорогих додатків, і так далі. Оскільки з Citrix все це може бути зроблено десь віддалено в Citrix Server, і користувачі підключаються до цих серверів за допомогою своїх тонких клієнтів Citrix (в основному клієнтську програму Citrix встановлений в ноутбуці, або це може бути виділеним Citrix обладнанням клієнта, такі як WYSE , і так далі). Великим

недоліком цієї технології є те, що наявність хорошою пропускною спроможністю. Як можна зрозуміти, Citrix більше схожий на презентації шару, а також прикладного рівня. Звідси дуже інтерактивний, і швидкісний мережевий трафік. В основному кожен з ваших кліків миші, клавіш на клавіатурі, і багатьох інших аспектів GUI, перетворюються в пакети протоколу ICA і передається по мережі. Звідси інтерактивний трафік. Якщо у вас є низька швидкість WAN, то, як правило, користувачів може дуже дратувати використання цього середовища, оскільки кожен раз, при введенні тексту на клавіатурі користувач, це дія займає кілька секунд, щоб вступили в силу на екрані, і те ж саме відбувається з мишею інтерактивності. Всі ці основні інтерактивні можливості графічного інтерфейсу користувача, який прекрасно працює в простому локальному комп'ютері, тепер буде більш недбало і повільно на столі Citrix. щоб вступили в силу на екрані, і те ж саме відбувається з мишею інтерактивності. Всі ці основні інтерактивні можливості графічного інтерфейсу користувача, який прекрасно працює в простому локальному комп'ютері, тепер буде більш недбало і повільно на столі Citrix. щоб вступили в силу на екрані, і те ж саме відбувається з мишею інтерактивності. Всі ці основні інтерактивні можливості графічного інтерфейсу користувача, який прекрасно працює в простому локальному комп'ютері, тепер буде більш недбало і повільно на столі Citrix.

Отже з TrafficSqueezer можна досить легко оптимізувати цей інтерактивний протокол трафік ICA Citrix. Ми повинні розуміти, той факт, що МКА вже оптимізований у власному контексті, а також ефективно стиснутий в залежності від ступеня, пріоритету і так далі. Але роль TrafficSqueezer в Citrix Application інтенсивної глобальної мережі може змінити це рівняння оптимізації мережі в іншому контексті. В основному, якщо у вас є більш ніж один робочий стіл Citrix в віддаленому філії, підключеного через WAN ^ допомогою TrafficSqueezer можуть зливатися пакети різних дискретних МКА сесій пакетів, а також оптимізувати і мінімізувати кількість пакетів, що передаються через WAN. Зрозуміло, що не всі пакети Citrix дійсно стискаються з будь–

яким алгоритмом стиснення TrafficSqueezer. Але це, тут присутній особливість шаблонів. З TrafficSqueezer можете зробити свою частину оптимізації трафіка. Отже через Citrix інтенсивних каналах глобальної мережі, функції оптимізації TrafficSqueezer такі як: пакет використання шаблонів і Packet зливаються, так можна ще більше оптимізувати цей трафік. Пакет Coalescing зменшить непотрібні запити для каналів глобальної мережі, за рахунок зменшення числа або кількість пакетів, шляхом злиття або мультиплексування проходження пакетів через той же пункт призначення. Це ізолює ці нові мультиплексовані пакети, тобто згубні наслідки МГФ міжкадрових зазорів і сценаріїв затримки між кадрами. так можна ще більше оптимізувати цей трафік. Пакет Coalescing зменшить непотрібні запити для каналів глобальної мережі, за рахунок зменшення числа або кількість пакетів, шляхом злиття або мультиплексування проходження пакетів через той же пункт призначення. Це ізолює ці нові мультиплексовані пакети, тобто згубні наслідки МГФ міжкадрових зазорів і сценаріїв затримки між кадрами. так можна ще більше оптимізувати цей трафік. Пакет Coalescing зменшить непотрібні запити для каналів глобальної мережі, за рахунок зменшення числа або кількість пакетів, шляхом злиття або мультиплексування проходження пакетів через той же пункт призначення. Це ізолює ці нові мультиплексовані пакети, тобто згубні наслідки МГФ міжкадрових зазорів і сценаріїв затримки між кадрами.

TrafficSqueezer пропонує як TCP Acceleration (сенс змінити параметри конкретного управління Протокол TCP) і TCP оптимізація (можливо замінити використання протоколу і т.д.), щоб зменшити передавання пакетів в дуже великій кількості по низькошвидкісних WAN мереж. Відомо TCP надійний зв'язок і надаючи підключення орієнтованих послуги зв'язку точка–точка. TCP з точки зору контексту протоколу не обізнані про network–connection / bandwidth доступності та обмежень. Звідси повені, відправивши дані по максимуму, поки не помітите будь–який пакет втрачає. Це може виявитися дорогим по низькій швидкості зв'язку або WAN такому середовищі. Тому це є

причиною випадково TCP називають "балакучий" протоколу. TCP оптимізація – з TCP на UDP туннелювання.

Це дуже важливо для низькошвидкісної мережі. У разі, якщо пакет втрачений додаток з відправляє UDP може вибірково надавати надійність, просто повторно передачі пакетів. Так ще можна домогтися надійності від тунелями на основі UDP. Оскільки в цьому методі ми тунелювання TCP передачі через UDP, це насправді один з методів TCP оптимізації в проекті TrafficSqueezer. І як тільки пакети перетворяться в UDP він може бути переданий на компресію в циліндрах двигуна в TrafficSqueezer.

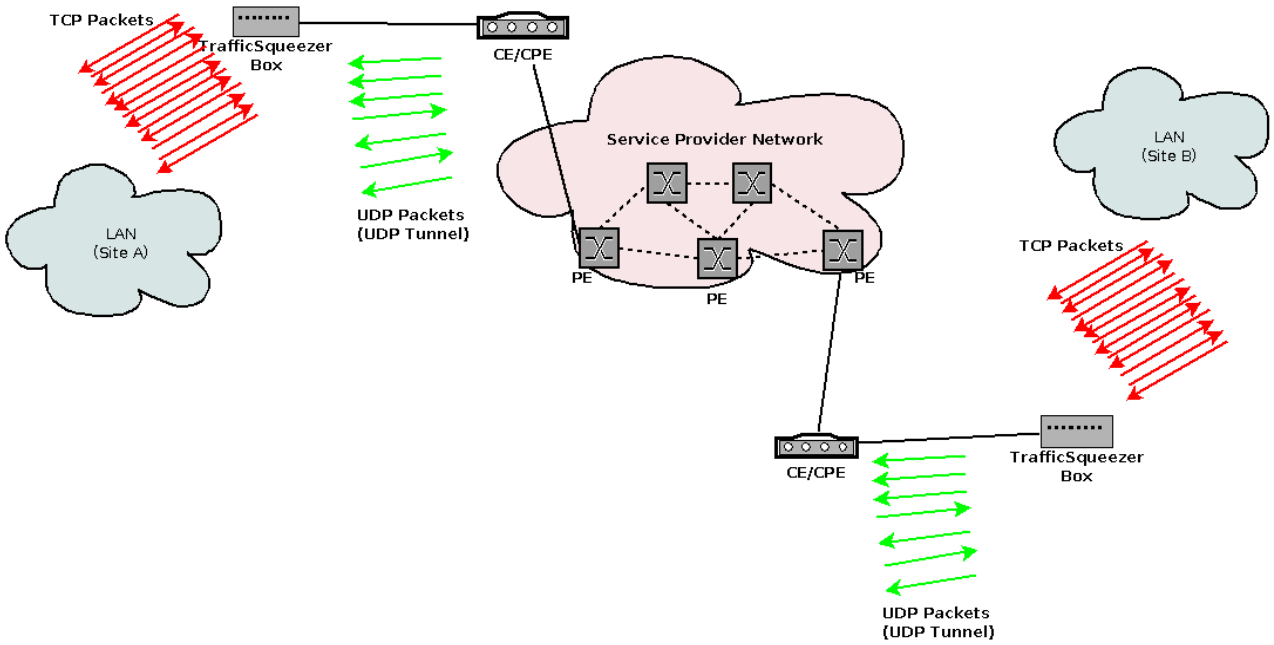

Рисунок 1.11– TrafflcSqueezer, TCP оптимізація з UDP тунелюванням

З метою досягнення ще більш ефективного WAN прискорення, мережевого трафіка використовується якість обслуговування (QoS) і можуть бути включені в русі відкат. Неабияка частка пропускної здатності зарезервований для кожного з службового трафіка, який проходить по з'єднанню WAN. Також на основі пріоритетів вимоги руху можуть бути включені.

Якість обслуговування є здатністю забезпечити свій пріоритет в різних додатках, користувачів або потоків даних, або гарантувати певний рівень продуктивності до потоку даних. Наприклад, потрібно швидкість передачі бітів, затримка, тремтіння, пакет ймовірність або частоти появи помилкових бітів може бути гарантована. Якість обслуговування гарантій важливі, якщо ємність мережі недостатня, особливо в режимі реального часу потокове мультимедіа–додатків, таких як передача голосу по IP і IP–TV, так як вони часто вимагають фіксованої швидкості передачі даних і затримки чутливі, і в мережах, де ємність обмеженим ресурсом, наприклад, в зв'язку стільникового даних. За відсутності перевантаження мережі, механізми QoS не потрібні.

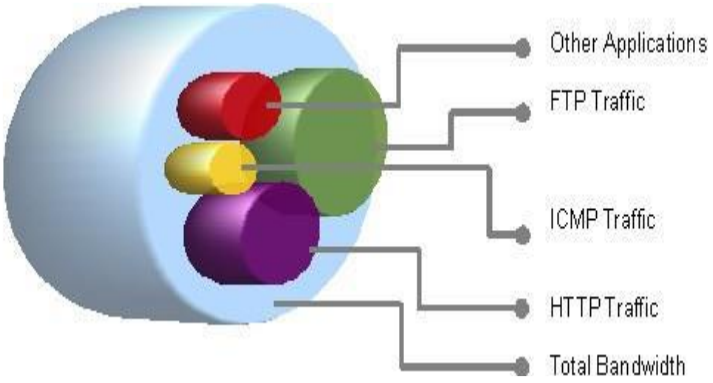

Рисунок 1.12 – Графічне представлення об'єднаної політики–приклади QoS [4]

1.3.3 Мережеве програмне забезпечення Riverbed

Продуктивність приватної, громадської, і інтернет–мережі має вирішальне значення для кожної глобальної системи зв'язку підприємства. Мережі Wide Area (WAN) – і тепер гібридні мережі – що з'єднують людей, додатки і дані, щоб зробити бізнес на відстані можливим. З приладами оптимізацією Riverbed Steelhead для гібридних мереж, організації стануть працювати швидше і більш ефективно з підвищенням продуктивності додатків і надання послуг.

Причини повільної пропускної здатності в глобальних мережах, добре відомі: високі затримки (час прийому–передачі або затримки), обмежена пропускна здатність, і балакучі прикладні протоколи. Великі підприємства витрачають значну частину своїх інформаційно–технологічних бюджетів на утримання мереж, багато з яких провели для компенсації малої пропускної здатності шляхом розгортання резервних серверів і систем зберігання, а також необхідність резервного обладнання. Техніка Steelhead дозволяє консолідувати і централізувати ключові IT–ресурси, щоб заощадити гроші, спростити ключові бізнес–процеси, і підвищити загальну продуктивність мережі.

RiOS це програма, якою оснащений прилад Steelhead і Steelhead Mobile. З RiOS, можна вирішити цілий ряд проблем, що впливають на глобальні мережі і продуктивності додатків, в тому числі:

- нехватку пропускної здатності WAN;
- неефективні транспортні протоколи з високою затримкою;
- неефективні прикладні протоколи з високою затримкою.

RiOS перехоплює з'єднання клієнт / сервер, не заважаючи нормальному клієнт / сервер взаємодії. Всі клієнтські запити надходять на сервер як зазвичай, хоча даний трафік вже оптимізований для підвищення продуктивності.

З упорядкуванням даних прилади Steelhead і Steelhead Mobile можуть зменшити навантаження каналів глобальної мережі від 65% до 98% для додатків на TCP основі.

Масштабованість даних. На додаток до традиційних методів, як стиснення даних, RiOS також використовує власний алгоритм під назвою Scalable реферування даних (SDR). RiOS SDR розбиває дані TCP потоки в унікальні частини даних, які зберігаються на жорстких дисках (Rios зберігання даних) пристрої під управлінням RiOS (прилад Steelhead або система Steelhead Mobile). Кожному блоку даних присвоюється унікальний цілий ярлик (посилання) перед відправкою на інші пристрої RiOS пристрою через WAN мережі. Коли в наступних предачі даних відбувається послідовність байтів клієнтів або серверів, з'єднання відправляється через глобальну мережу. Сусіднє

RiOS пристрій (прилад Steelhead або система Steelhead Mobile) використовує це з'єднання, щоб знайти вихідні дані про сховище даних RiOS і відтворити початковий потік даних TCP.

Файли та інші структури даних можуть бути прискорені шляхом упорядкування даних, навіть якщо вони передані з використанням різних додатків. Наприклад, файл, який спочатку був переданий через CIFS прискорюється, коли він знову передається через FTP.

Додатки, які кодують дані в іншому форматі, коли вони передають їх через глобальну мережу також можуть бути прискорені шляхом оптимізації даних. Наприклад, Microsoft Exchange використовує протокол MAPI для кодування вкладених файлів перед відправкою їх клієнтам Microsoft Outlook. В рамках своїх MAPI конкретних оптимізованих з'єднань, RiOS декодує дані перед застосуванням SDR. Це дозволяє приладу Steelhead дізнатися послідовності байтів в файлових вкладень, коли файл згодом передається через FTP або копіюється на CIFS папку.

Двонаправлене синхронне RiOS сховище даних Дані підтримуються в постійній пам'яті в сховища даних в межах кожного пристрою RiOS і стабільні навіть після перезавантаження та оновлення системи. Для забезпечення подальших довгої служби та безпеки, локальні пари приладів Steelhead необов'язково повинні знаходитися в сховищ даних, по скільки повністю синхронізовані в обох напрямках весь час. Двунаправленна синхронізація гарантує, що відмова з одного пристрою Steelhead НЕ появліяеет на інші віддалені пристрої Steelhead при відправленні раніше переданих даних. Ця функція особливо корисна, коли локальні прилади Steelhead розгорнуті в мережному кластері, таких як майстер і резервне копіювання розгортання, серійний кластер або кластер WCCP.

Steelhead техніка використовує метод оптимізації званий транспортний впорядкування.

Транспортне впорядкування використовує набір стандартів і власних методів для оптимізації TCP трафіка між Steelhead пристроями. Ефективні властивості цих методів:

- гарантують, що ефективні методи ретрансляції використовуються (наприклад, TCP селективних підтверджень);
- оптимальні розміри вікон TCP, щоб мінімізувати вплив латентності на пропускну здатність;
- максимально збільшують пропускну здатність в широкому діапазоні каналу глобальної мережі.

За замовчуванням, техніка Steelhead використовує стандартний протокол TCP, як це визначено в RFC 793, для зв'язку між пристроями. У Steelhead приладах також можна включити високошвидкісний протокол TCP (HS–TCP), як це визначено в RFC 3649, для досягнення високої пропускної для зв'язку з високою пропускною спроможністю і високу латентність.

Можна також вибірково використовувати максимальну швидкість протоколу TCP (MX–TCP), який необхідно передавати з певною швидкістю через WAN, незалежно від наявності інших видів трафіка. Хоча це і не підходить для всіх середовищ, MX–TCP може підтримувати передачу даних з хорошою пропускною спроможністю, де погана якість мережі, а саме дуже великі втрати пакетів, які погіршують продуктивність і пропускну здатність звичайних з'єднань TCP. MX–TCP ефективно обробляє ці втрати пакетів, і вже без втрат працює в звичайному режимі передачі TCP.

Деякі прикладні протоколи, такі як HTTP, використовують швидко створені, короткоживучі з'єднання TCP. Для оптимізації цих протоколів, пристрої Steelhead створює невикористовувані пули для з'єднань TCP. Коли клієнт намагається створити нове підключення до раніше відвідав сервері, Steelhead використовує один з його пулу з'єднань. Таким чином, клієнту і приладу Steelhead не доведеться чекати довгого відповіді TCP вітання, щоб пропустити через WAN. Ця функція і називається пул з'єднань. Пул сполук

доступна тільки для з'єднань, яке використовує правильний режим адресації і видимості WAN мережі.

Транспортне впорядкування гарантує, що завжди є ставлення один–до–одному для активних сполук TCP між Steelhead і з'єднаннями TCP у клієнтів і на серверах. Незалежно від режиму видимості WAN в використанні, Steelhead виконує мультиплексування і демультиплексування даних по з'єднаннях.

Видимість WAN.

Кожне з'єднання LAN– на стороні TCP, яке оптимізовано за допомогою приладу Steelhead здійснюється на унікальному WAN– боковим підключенням. Налаштувавши режим видимості WAN для деяких або всіх оптимізованих з'єднань, ви можете контролювати, які IP–адреси і порти TCP використовуються на цих WAN – бічних з'єднань TCP.

RiOS v6.0 або більш пізньої версії пропонує наступні опції для настройки режимів видимості WAN:

- правильна адресація – WAN з'єднання обов'язково використовують Steelhead Appliance IP–адрес і портів Steelhead Appliance сервера;
- прозора адресація– прикладами є такі прозорі адресації.

Прозорість порта– WAN з'єднання обов'язково використовують Steelhead прилад IP–адреси, але не використовує серверні TCP порти, які відображають підключення до локальної мережі на стороні.

Повна прозорість– WAN з'єднання обов'язково відображають всі IP–адреси і порти TCP, використовувані на підключення до локальної мережі.

Повна прозорість з Forward Reset – так само, як повна прозорість, тільки з додатковим пакетом. Під час автоматичного виявлення, щоб допомогти з інтеграцією зі станом мережевих пристроїв в глобальній мережі.

Найбільш відповідний режим видимості WAN залежить перш за все від існуючої конфігурації мережі. Наприклад, якщо вам вдасться закріпити IP адреса на основі TCP політики або на основі портів QoS для оптимізації трафіка на ваших WAN маршрутизаторів, можна використовувати повну прозо-

рість адреси або прозорість порту. Однак, якщо потрібен оптимізований трафік проходить через міжмережевий екран, який в створює тривогу, в разі, коли порти додатків використовуються на вже в оптимізованої навантаження трафіка, тоді вірніше буде використання режиму (правильна адресація). Можна налаштувати режими видимості WAN на стороні клієнта Steelhead приладу (де починається з'єднання).

Там можуть бути різні типи режимів адресації на одному і тому ж Steelhead пристрої. Вибирати потрібно найбільш підходящий режим адресації для наявний конфігурації мережі, заснованої на IP–адреси, підмереж, портів TCP і VLAN.

Правильна адресація.

Правильне адресація використовує Steelhead Appliance IP адреси і номери портів в полях заголовка пакета TCP / IP для оптимізації трафіка в обох напрямках по всій глобальній мережі. За замовчуванням прилади Steelhead використовують режим правильної адресації.

Заголовки TCP / IP пакетів при правильній адресації використовуються IP–адреси і номери портів з наявних Steelhead приладів можуть бачити через WAN мережу.

Правильна адресація використовує наступні значення в TCP / заголовків IP–пакетів в обох напрямках:

- клієнт для клієнтської сторони Steelhead приладу – IP адреса клієнта і порт + IP адреса сервера і порт;

- на стороні клієнта Steelhead прилад для серверних Steelhead Appliance – на стороні клієнта IP адреса Steelhead приладу і порт + IP адреси на стороні сервера Steelhead приладу і його порт;

- серверне Steelhead пристрій на сервер IP– адреса клієнта і порт + IP адреса сервера і порт;

Правильні адресації уникають ризики мереж, які притаманні режиму прозорої адресації.

Режим правильної адресації дозволяє використовувати функцію оптимізації пулу з'єднань. Оскільки пул з'єднань працює тільки для з'єднань, оптимізованих за допомогою режиму правильної адресації. Пул сполук дозволяє Steelhead приладів для створювати кілька з'єднань TCP між собою, до того як вони стануть необхідні. Коли прозоре рішення включено, Steelhead техніка не може створити з'єднання TCP заздалегідь, тому що вони не можуть виявити, які типи клієнтів і IP–адрес серверів і портів необхідні їм.

Прозора адресація.

Прозорий клієнт використовується для повторної адресації і як сервер вирішенні для оптимізації трафіка через WAN. Трафік оптимізований, хоча в режимі видимої адресації, без особливих змін. Обидва режими оптимізують проходить через нього трафік. Надає адресну інформацію до маршрутизатора і моніторингу мережевих пристроїв.

У RiOS v5.0.x або більш пізньої версії, в режимі прозорою адресації можуть бути використані в поєднанні з великою кількістю конфігурацій і функцій розгортання:

- фізичні – шляхи розгортання (серійні кластери, майстер / резервного копіювання та розгортання з використанням з'єднання між клієнтом і сервером);
- віртуальні – шляхи розгортання (WCCP, PBR, 4–го рівня перемикання і розгортання перехоплювач техніки);
- автоматичне виявлення, в тому числі підвищеного автоматичного виявлення;
- асиметричне виявлення маршруту;
- QoS маркування та їх класифікації;
- експорт даних потоку.

Режим прозорою адресації не підтримує такі зміни:

- Серверний поза шляху конфігурації приладу Steelhead.
- Пул сполук.

Налаштувати прозору адресацію можна на стороні клієнта Steelhead приладу (де починається з'єднання). І на стороні сервера і Steelhead пристрою. На стороні клієнта має підтримуватися прозора адресація (RiOS v5.0.x або вище). Можна налаштувати Steelhead прилад для прозорої адресації, навіть якщо його партнер не підтримує його. З'єднання оптимізовано, але це не є прозорим.

При використанні повної або порту прозорості, Steelhead техніка додає поле TCP опції до пакетним заголовкам оптимізованого трафіка. Це TCP опція відправляється між технікою Steelhead. Для забезпечення прозорості в роботі, цей варіант не повинен бути для проміжних мережевих пристроїв.

Дана пара Steelhead техніки також може мати кілька типів прозорою адресації включеної для різних з'єднань. Наприклад, пара Steelhead пристроїв можуть використовувати режим правильної адресації для підключень до мережі одержувача, а також можуть використовувати режим повної прозорості адреси або прозорості порту для підключення до іншого призначення підмережі.

Якщо необхідно використовувати клієнтські IP–адреси або IP–адресоа серверів в масштабах глобальної мережі, повна прозорість адреса ваш єдиний вибір конфігурації. Повна прозорість адреси дозволяє додаткам мережевого моніторингу, наявних в рамках глобальної мережі (між приладами Steelhead) вимірювати трафік до глобальної мережі на кінцевому хості. Маршрутизатор також можуть виконувати балансування навантаження і маршрутизацію на основі цих політик. Повна прозорість адреси також дозволяє керувати і застосовувати політиками QoS на основі портів або IP– адрес.

Важливо: При включенні режиму повної прозорості адресації, політики QoS на маршрутизаторі не може відрізнити оптимізований від неоптимізованого трафіка, хоча оптимізований пакет може представляти набагато більше даних. Повна прозорість адресації також дозволяє використовувати Network Address Translation (NAT). У режимі правильної адресації, Steelhead пристрій використовує свої власні IP–адреси в заголовку пакета, який NAT не визнає.

Коли повна прозорість адресації включена, і використовуються оригінальні IP–адреси сервера і клієнта, то з'єднання стають пізнаваними для NAT.

Повна прозорість адресації також підтримує деякі параметри прозорості для зв'язку OOB.

Важливо – деякі брандмауери, пристрої QoS і інші зберігають стан пристрої можуть вимагати додаткової настройки, тому для успішної з'єднання використовують повну прозорість адресації для оптимізації.

Налаштування мережі VLAN і повної адресації прозорості.

Повна прозорість адресації підтримує прозорі мережі VLAN. Можна налаштувати повну прозорість адресації так, щоб оптимізований трафік залишається на оригінальних VLAN. Тому що можна зраджувати трафік через віртуальні локальні мережі, повна прозорість адресації дозволяє виконувати VLAN на основі політик QoS WAN– на стороні пристрою Steelhead.

Примітка: спочатку налаштувати видимість WAN, при режимі повної адресації прозорості VLAN для правильного функціонування. Щоб налаштувати повну прозорість адресації для VLAN

На пристрої Steelhead, підключення до CLI і введіть наступні команди:

- enable;
- configure terminal;
- in–path peering auto;
- in–path simplified routing all;
- in–path vlan–conn–based;
- in–path mac–match–vlan;
- no in–path probe–caching enable;
- in–path probe–ftp–data;
- in–path probe–mapi–data;
- write memory;
- restart.

Примітка: необхідно зберегти зміни, інакше вони будуть втрачені при перезавантаженні.

Оптимізований трафік при втрати маршруту.

Включення прозорою адресації призводить до ймовірності втрати маршруту оптимізованого трафіка в разі відмови одного з приладів Steelhead.

Пристрої Steelhead використовують власний протокол Riverbed для спілкування між собою пакетами вітання. Як правило, функціонують вони на стороні сервера Steelhead прилад отримує пакет з глобальної мережі, і перетворює цей пакет у власний формат до відправки його на інший сервер.

У середовищі, в якій використовується прозоре рішення адресації, то якщо на стороні сервера Steelhead прилад не працює, або якщо пакет направляється за іншим маршрутом мережі, пакет може перейти від стороні клієнта Steelhead пристрою безпосередньо до сервера. Оскільки на стороні сервера Steelhead прилад не має можливості для конвертації пакету в своєму рідному форматі, сервер не може розпізнати його, і з'єднання втрачено.

У більшості випадків, сервер здатний виявити те, що пакет містить або невірну інформацію, або, в даному прикладі, має невпізнанним формат, і відкидає цей пакет. Припускаючи, що сервер не виявить, що формат невпізнанним, сервер відхиляє пакет і скидає з'єднання TCP. Якщо з'єднання з клієнтом TCP скидається, клієнт може підключитися до сервера без будь–якої участі Steelhead пристрою.

Цей тип трафіка при втрати маршруту може статися в обох напрямках через WAN.

Якщо на боці клієнта Steelhead прилад вийде з ладу, або якщо альтернативний шлях мережі існує з сервера на клієнт, трафік може піти від серверного Steelhead приладу безпосередньо клієнтові.

Важливо: Перш ніж почати користуватися і використанням повної прозорості адресації, ретельно необхідно розрахувати ризики в тому випадку, якщо сервер прийме пакет, який має невпізнанним формат.

Сценарій при помилки з форматом пакета йдуть такі дії:

1) клієнт посилає дані HTTP на сервер;

2) Steelhead B отримує дані HTTP оптимізує його. Steelhead B в кінцевому рахунку передає пакети, що несуть оптимізовані дані в сторону Steelhead C, але через режим прозорою адресації, пакети адресовані на Сервер D;

3) Steelhead З зазнає невдачі і починає працювати в аварійному режимі, так що всі пакети обходять його, в тому числі пакети з Steelhead B;

4) сервер D приймає пакети від Steelhead B, але не розпізнає формат пакета, тому з'єднання не утворюється або погано працює додаток, залежить від помилки;

Можна уникнути такого роду проблеми як втрати маршруту, яка властива до прозорого вирішення, за допомогою правильної адресації (рисунок 1.13).

Якщо правильна адресація налаштований для цього сценарію, на стороні клієнта Steelhead прилад виявляє, що на стороні сервера Steelhead прилад вийшов з ладу. На стороні клієнта Steelhead прилад автоматично скидає з'єднання клієнта, що дозволяє клієнту підключатися безпосередньо до сервера без Steelhead участі приладу.

Рисунок 1.13– Міжмережеві екрани і режим прозорою адресації

1.4 Технологія стиснення трафіка LZO, LZMA, LZW

LZO це алгоритм стиснення даних, розроблений для досягнення максимальної швидкості розпакування. LZO [1] – це абревіатура від прізвищ ро-

зробників: Лемпеля, Зів, Оберхеймер. Це алгоритм стиснення без втрат і його базова реалізація може працювати в багатопотокової середовищі.

Вільної програмою, що реалізує LZO, є lzop. Вихідна бібліотека була написана на ANSI C і доступна під ліцензією GPL. Також існують реалізації LZO на мовах Асемблер (x86), Perl, Python і Java. Код написаний Маркусом Оберхеймером LZO–бібліотека реалізує кілька алгоритмів, з наступними особливостями:

1. Розпакування просте і дуже швидке.

2. Для декомпресії не потрібно додаткової пам'яті крім буферів для стислих і розпаковувати даних.

3. Стиснення також дуже швидке.

4. На стиснення потрібно 64 кб пам'яті.

Можна досягти додаткового стиснення, витративши трохи більше часу при стисненні. При цьому швидкість декомпресії не зменшується.

Існує кілька рівнів стиску. Зокрема реалізований варіант алгоритму, яким потрібно всього 8 кб пам'яті для стиснення. Алгоритм безпечно застосовувати в багатопотоковому середовищі.

1.4.1 LZMA

LZMA алгоритм стискає без втрат алгоритм стиснення даних, що розробляється з 2001 року. Використовується в архіваторі 7–Zip для створення стислих архівів у форматі 7z.

Алгоритм заснований на схемі стиснення даних по словнику, подібна до використаної в LZ, і забезпечує високий коефіцієнт стиснення (зазвичай перевищує коефіцієнт, що отримується при стисненні з використанням bzip2), а також дозволяє використовувати словники різного розміру (до 4 Гб).

Так само LZMA – вільна утиліта командного рядка для стиснення данних. Бібліотека стиснення з відкритим вихідним кодом LZMA, написана на мові C ++, використовує покращений алгоритм стиснення LZ77, доповнений

алгоритмом інтервального кодування, а також спеціальними процедурами для обробки двійкових файлів .

LZMA підтримує різні варіанти хеш–ланцюгів, довічних і префіксних дерев в якості основи алгоритмів пошуку по словнику.

LZMA SDK пропонує алгоритм BCJ / BCJ2, реалізований для процесорів архітектури x86, ARM, PowerPC, IA–64 і ARM Thumb. У ньому точки переходу перед стисненням нормалізуються – тобто, наприклад, для x86 це означає, що інструкції ближніх і умовних переходів і виклики функцій перетворюються з форми з відносним зсувом «перейти на 1665 байт назад» в форму з абсолютним адресою «перейти до адресою 5554 ».

Алгоритм BCJ2, реалізований в 7–Zip, використовує 32–бітну адресацію. У архіваторі для виконуваних файлів UPX адресація залежить від типу архітектури (наприклад, для виконуваних файлів DOS використовується 16–бітна адресація). Реалізація, доступна за ліцензією GNU LGPL, має такі властивості:

- швидкість стиснення: приблизно 1 Мб / с на процесорі x86 з частотою 2 ГГц;
- швидкість витягання: близько 10–20 Мб / с на процесорі x86 з частотою 2 ГГц;
- підтримка багатопоточності;
- розмір коду розпакування LZMA становить близько 5 Кб;
- витрата динамічної пам'яті залежить від розміру словників;

Ці можливості дозволяють реалізувати розпакування на вбудованих системах.

Використання особливостей Microsoft Windows в вихідному коді ускладнює створення версій програми для Unix. Проте, існує дві працездатні портований версії: в p7zip більш–менш портировані версії утиліт командного рядка 7z і 7za для POSIX–систем (GNU / Linux, Solaris, OpenBSD, FreeBSD, Cygwin і інших), Mac OS X і BeOS. Також є офіційна портіруемость реалізація – LZMA Utils, призначена для створення потокових компресорів подібних

gzip. З 2008 року вона починає все частіше використовуватися в системах управління пакетами – зокрема, DPKG та RPM.

7–Zip використовує досить гнучкий формат архіву, його підтримують і деякі сторонні утиліти (наприклад, читання 7z підтримує WinRAR).

Також існує порт 7–Zip для Mac OS X, який називається Compress, в даний час являє собою досить недопрацьований інструмент. Для Mac OS X існують ще збірки p7zip і 7zX.

Для роботи з LZMA автор надає свій кроссплаформенний SDK, що володіє перерахованими вище властивостями. Основна частина SDK написана на C ++ і спочатку поширювалася на умовах GNU LGPL. Варто відзначити кілька моментів:

З версії 4.62, LZMA SDK став доступний на умовах Public Domain, тобто допускається його використання для будь–яких цілей без будь–яких обмежень.

Деякі мережеві пристрої (на зразок US Robotics 9105 і 9106) в якості вбудованого використовують модифікований Linux, що завантажується зі стислою файлової системи. В якості алгоритму стиснення файлової системи замість Zlib використовується алгоритм LZMA. Як правило, такий файлової системою є squashfs з LZMA–патчем.

LZMA2 – нова версія алгоритму LZMA. Даний алгоритм має наступні переваги перед алгоритмом LZMA:

- кращий ступінь стиснення для даних, що погано піддаються компресії.
- Контейнер може одночасно включати в себе стиснені та стислі дані.
- Краща підтримка багатопоточності при компресії і декомпресії.

1.4.2 LZW

LZW– це універсальний алгоритм стиснення даних без втрат. Даний алгоритм при стисненні (кодуванні) динамічно створює таблицю перетво-

рення рядків: певним послідовностям символів (слів) ставляться у відповідність групи біт фіксованої довжини (зазвичай 12–бітові).

Таблиця ініціалізується усіма 1–символьними рядками (в разі 8–бітних символів – це 256 записів). У міру кодування, алгоритм переглядає текст символ за символом, і зберігає кожну нову, унікальну 2–символьний рядок в таблицю у вигляді пари код / символ, де код посилається на відповідний перший символ. Після того як нова 2–символьний рядок збережена в таблиці, на вихід передається код першого символу.

1.5 Мета й задачі проекту

Основною метою цього проекту є в розробка архітектури програмного забезпечення для створення Acceleration Framework WAN мережевого трафіка по Linux Box.

Цілями проекту є: установка нового мережевого обладнання в якості резерву, в разі виходу з ладу основного, щоб уникнути простою мережі, і стиснення трафіка, що передається між головним офісом і філіями, який проходить через глобальну мережу, в так званих WAN мережах. Тому що основні витрати припадають на надання провайдерами послуг надання ширини смуги пропускання і швидкості Інтернету.

В даному проекті необхідно розглянути питання модернізації мережі зв'язку в результаті оптимізації трафіка WAN мереж і отримати повну її відмовостійкість. Основними задачами проекту є:

- Аналіз і вибір програмних засобів для оптимізації трафіка WAN мережі.
- Модернізація мережі за допомогою додаткового мережевого обладнання Cisco.
- Розробка алгоритму динамічного управління трафіком мережі.
- Оптимізація трафіка наявних WAN мереж між головним офісом і іншими філіями.

В експериментальній частині необхідно провести повні установки і настройки кожного сценаріїв тестів із зазначенням повного списку дій і прикладів.

Після чого необхідно виконати тестування по кожному окремо і вибрати оптимальне мережеве програмне забезпечення для оптимізації трафіка.

Для сучасних корпоративних мереж передачі даних характерні такі тенденції, як централізація IT–ресурсів в ЦОД, активний доступ до них мобільних користувачів, використання Інтернету або виділених каналів WAN для організації комунікацій між офісами. Консолідація, віртуалізація, хмарні обчислення, Web–сервіси, зростання числа і різноманітності мобільних пристроїв, віддалена робота, збільшення обсягів збережених і переданих даних, централізація додатків – все це змушує звернути більш пильну увагу на оптимізацію WAN.

При збільшенні завантаження каналу WAN втрати пакетів відбуваються частіше, що, в свою чергу, веде до погіршення якості роботи і збільшення часу відгуку додатків. Нарощування пропускної здатності каналів (власних або орендованих) нерідко обходиться дорого і не завжди допомагає – затримка в мережі все одно залишається дуже великою. Іноді проблему вдається вирішити (частково або повністю) за рахунок застосування правил пріоритетного обслуговування (CoS / QoS), зміни налаштувань додатків або перегляду архітектури рішення.

З метою оптимізації результатів дослідження розглянемо сегмент мережі WAN, як транспортну мережу хмари.

2 ОБҐРУНТУВАННЯ ОПТИМІЗАЦІЇ ТА МОДЕРНІЗАЦІЇ МЕРЕЖІ

На сьогоднішній день в зв'язку з ростом компанії, в зв'язку з постійним підвищенням цін на послуги телекомунікацій. Необхідно шукати шляхи вирішення, для того щоб якомога сильніше поліпшити наявну мережу і знизити навантаження на те обладнання, яке в даний момент знаходиться в експлуатації. Але й особливої завданням є зниження витрат компанії, безпосередньо на телекомунікаційні послуги.

Вся увага цього проекту полягає в оптимізації трафіка WAN мереж, саме той, на який більш за все відбуваються витрати. Оскільки переданий трафік займає майже всю смугу пропускання, і швидкість, яку надають провайдери. У випадку з очікуємо результатом оптимізації, будуть прискорений документообіг, робота мережевих додатків, швидкість виконання грошових операцій не буде необхідності купувати великі швидкості у Інтернет провайдерів.

І з цим зростає і навантаження на всю мережу компанії в цілому і наявні обладнання. І випадку з перебоями стали відбуватися регулярно, тому необхідна модернізація, в основу буде взята мережу головного офісу, оскільки всі основні сервера: сервер терміналів, DC, сервер СУБД, проксі–сервер, ФТП і т.д., робочі станції і інші мережеві пристрої знаходяться в головному офісі.

За рахунок впровадження в наявну мережу додаткових маршрутизаторів і комутаторів буде підвищена відмовостійкість.

2.1 Аналіз існуючої мережі та обладнання

Єдина інформаційна мережа компанії «Alliance» складається з локальних мереж головного офісу (ГО), філій та відділень, об'єднаних в загальну мережеву структуру з використанням 2–х технологій VPN. Філії та головний

офіс об'єднані в єдину мережу за допомогою технології DMVPN на базі обладнання Cisco. Логічним центром мережі, де розташовані всі основні мережеві сервіси компанії, є головний офіс.

Однак в цілому мережа компанії має розподілену структуру, де кожна філія зі своїми відділеннями утворює власну область, зі своїм адресним планом і маршрутизацією, і може виконувати основну операційну діяльність незалежно від головного офісу.

Локальні мережі ГО, філій та відділень побудовані за технологією FastEthernet 100 або Gigabit Ethernet 1000 Мбіт / с. Також використовуються бездротові точки доступу з обов'язковим шифруванням для підключення мобільних клієнтів. Підключення до мережі Інтернет у філіях і ГО налаштоване через ВАТ Укртелеком. Філії додатково підключені до послуги IP VPN. Технологія підключення – ADSL або SHDSL. Існує можливість підключення мобільних співробітників до робочої мережі ГО і філій через Інтернет за допомогою технології ВПН. На маршрутизаторах ГО і філій налаштовані сервери EasyVPN. Завдяки цьому співробітники Компанії мають можливість користуватися ресурсами нашої мережі і виконувати багато функцій віддалено використовуючи ПО Cisco VPN Client. Відділення підключені до Мережі через послугу ВАТ Укртелеком обласної IPVPN. Вихід в Інтернет,

2.1.1 Мережа головного офісу

Мережа Головного офісу розділена на кілька сегментів: підмережа 192.168.x.0 / 24 – локальна мережа ГО. У ній розташовані локальні сервери ГО: сервер терміналів, DC, сервер СУБД, проксі–сервер, ФТП і т.д., робочі станції і інші мережеві пристрої співробітників ГО.

На рисунку 2.1 зображена загальна схема WAN мережі компанії.

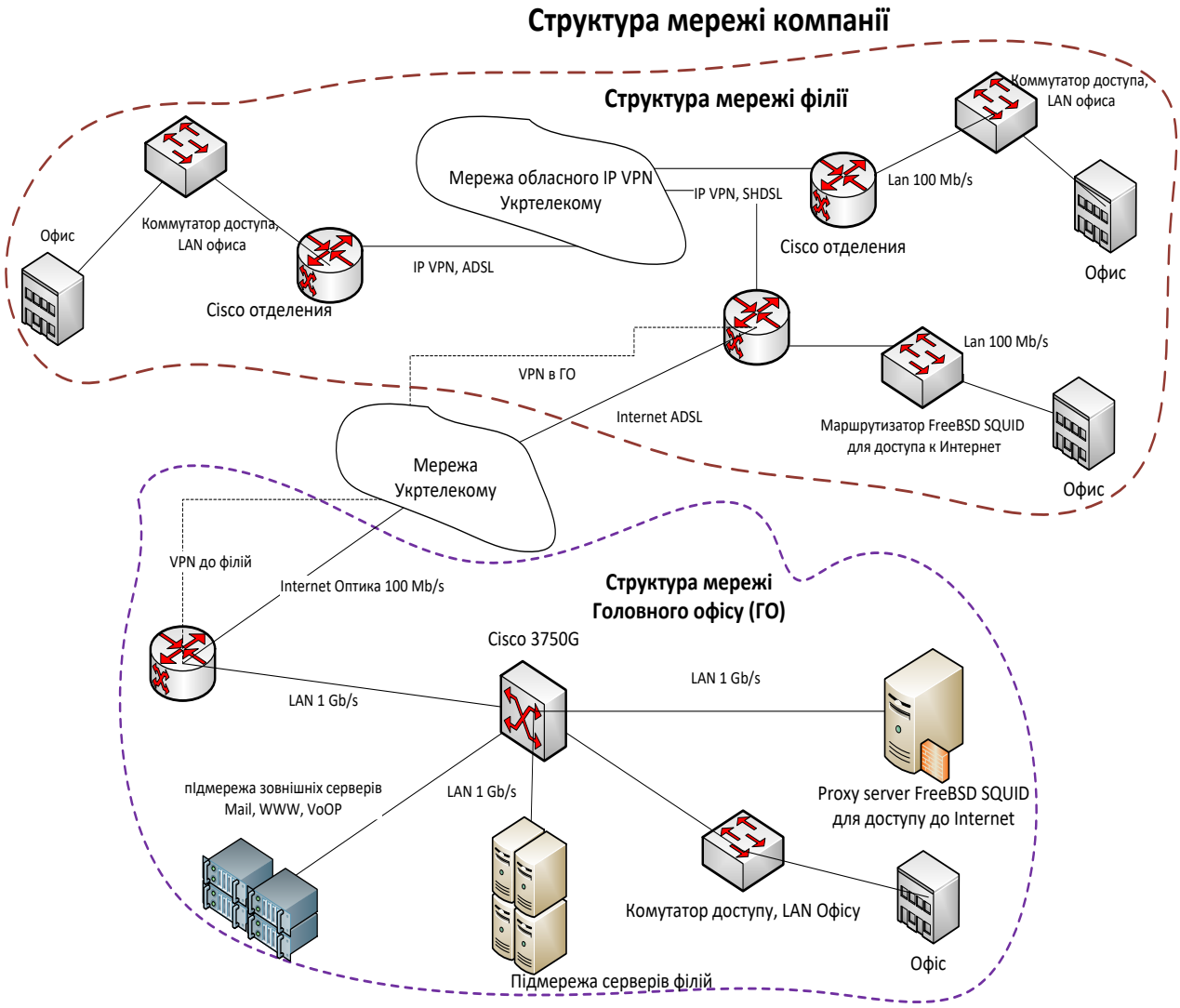

Рисунок 2.1 Схема існуючої мережі компанії, до модернізації

Мережеві параметри лунають автоматично за допомогою DHCP сервера запущеного на контролері домену ГО. Під мережа 212.154.xxx.xxx 255.255.255.248 – блок з 8 адрес, орендований у Укртелекому, в цій підмережі розташовані сервери Mail, WWW, VoIP, комутатор Cisco 3750, який є для них шлюзом. Інші адреси вільні.

Підмережа 172.16.x.0 / 24 –підмережа серверів, до яких мають доступ філії. Тут розташовані сервери 1С8, Міраполіс, також планується поставити сервер первинного контролера домену.

Гостьова підмережа 10.10.xx 255.255.255.128– обмежена мережа, для підключення гостей до мережі Інтернет.

Підмережа 192.168.x.0 255.255.255.0 –підмережа між L3 комутатором і маршрутизаторами.

У ГО підключення до мережі Інтернет зроблено через двох провайдерів: Укртелеком і Тріолан.

Укртелеком підключений через маршрутизатор Cisco 3945. Автоматичне балансування трафіка налаштована на комутаторі Cisco 3750, який є ядром мережі ГО. Цей комутатор підключений до всіх вищевказаних мереж і є для всіх цих мереж шлюзом за замовчуванням.

2.2 Вибір обладнання та налаштування оптимізатора, також для відмовостійкості мережі

Потужний персональний комп'ютер з процесором 64–біт, який буде здатний обробляти велику кількість різного трафіка між філією та головним офісом.

Для тестів їх необхідно в кількості 2 штуки вдалих результатів, будуть закуплені комп'ютери для інших філій. Вибір зупиняємо на персональному комп'ютері iCeleron G1610.

Для того, щоб оптимізатор працював в режимі моста йому необхідно 2 мережевих інтерфейсу. Купуємо 2 мережеві карти TP–LINK TG–3468 PCI–Express.

Для отримання повної відмовостійкості і для зняття навантаження з основних комутаторів в існуючу мережу будуть встановлені 2 додаткових комутатора Cisco Catalyst 3750 24 і Cisco Catalyst 3560 24, маршрутизатора Cisco 2811.

Основні технічні характеристики комутаторів наведені в таблиці 2.1–2.3.

Таблиця 2.1 Основні технічні характеристики комутатора Cisco Catalyst 3750

Тип пристрою	Перемикач – 24 порти – L3–керуючий – нарощу-ваних
Тип корпусу	стійку 1U
Інтерфейси	Fast Ethernet
Розмір таблиці MAC ад-рес	12K записів
протокол маршрутизації	RIP–1, RIP–2, EIGRP, статична маршрутизація IP, RIPng
Протокол віддаленого управління	SNMP 1, RMON 1, RMON 2, RMON 3, RMON 9, Telnet, SNMP 3, SNMP 2c, HTTP, SSH
відповідність стандартам	IEEE 802.3, IEEE 802.3u, IEEE 802.3z, IEEE 802.1D, IEEE 802.1Q, IEEE 802.3ab, IEEE 802.3x, IEEE 802.3ad (LACP), IEEE 802.1w, IEEE 802.1x, 802.3ae,
потужність	AC 120/230 В (50/60 Гц)
Розміри (ШхГхВ)	44,5 см x 30 см x 4,4 см
вага	3,6 кг

Таблиця 2.2 Технічні характеристики комутатора Catalyst 3560

Тип пристрою	Catalyst 3560–24PS	Catalyst 3560–48PS
Кількість портів Fast Ethernet 10/100 TX	24	48
Кількість портів Gigabit Ethernet SFP	2	4
Пропускна здатність, Гбіт / с	8,8	17,6
Продуктивність маршрутизації, млн. Пакетів / с	6,6	13,1
Тип транків VLAN	802.1q, ISL	
Тип ПО	SMI або EMI	
Обсяг flash–пам'яті, Мб	16	
Обсяг ОЗУ, Мб	128	
Розміри (В x Ш x Г), дюймів	I, 73 x 17,50 x II, 81	1,73 x 17,50 x 14,85

Таблиця 2.4– Технічні характеристики маршрутизатора Cisco 2811

Продуктивність	120 000 пакетів / с
WAN–протоколи:	Static IP / Routing, Dynamic IP / DHCP, PPPoE, PPPtP, NAT, PAT
LAN–сервіси	802.3af PoE (160 Вт), 802.1q
Мережева безпека	– Cisco IOS Software Firewall – Secure Sockets Layer (SSL) – Onboard VPN Encryption Acceleration – Network Admission Control (NAC) – Підтримка Multiprotocol Label Switching (MPLS) VPN – Intrusion Prevention System (IPS)
Протоколи маршрутизації	BGP, EIGRP, OSPF, RIPv1, RIPv2
QoS	L2 ToS, FIFO, RSVP, Priority Queuing, Custom Queuing, Weighted Fair Queuing, Class Based Weighted Fair Queuing
Управління	SSH, Telnet, RSH, SNMP, RADIUS, WEB–інтерфейс
Характеристики пам'яті:	
Флеш пам`ять	– За замовчуванням: 64 МБ – Максимум: 256 МБ
Оперативна пам'ять	– За замовчуванням: 256 МБ – Максимум: 768 МБ
Фізичні характеристики:	
Розміри, мм	438.2 x 416.6 x 44.5 1U
Вага, кг	6.4
Параметри живлення	– AC: 100 – 240 В, 47 – 63 Гц, 2 – 1 А – DC: 24 – 60 В, 8 – 3 А

2.3 Розробка алгоритму динамічного управління трафіком мережі

Для сучасних додатків вузьким місцем є мережі передачі даних, і однією з важливих задач забезпечення функціонування систем – програмне конфігурування завантажених мереж, спрямованих на використання додатків в гібридної хмарної інфраструктурі. І наша мережа тому не виключення.

2.3.1 Постановка задачі

При перевантаженні каналу зв'язку пакети поміщаються в черзі, в разі переповнення черги пакети відкидаються, що призводить до уповільнення швидкості передачі пакетів протоколом TCP / IP і втрати пакетів. Щоб домогтися гарантії якості обслуговування, застосовують QoS–архітектуру (Quality of service), яка включає в себе підтримку якості на всіх рівнях стека протоколів TCP / IP і у всіх мережевих елементах. Але і при цьому забезпечення гарантованої якості обслуговування все одно залишається найслабшим місцем процесу передачі інформації від джерела до приймача, так як QoS–архітектура являє собою систему поділу трафіка на статичні, заздалегідь визначаються класи з фіксованими пріоритетами, процентним співвідношенням ширини каналу для кожного типу трафіка .

Завдання полягає в розробці модуля на базі мережевого ПО, динамічного, що встановлює пріоритети типам трафіка в QoS–архітектурі.

2.3.2 Експериментальна частина

Для імітації методів управління був створений імітаційний експериментальний стенд (рисунок 2.1) у вигляді схеми нашої стей. Основою для експериментів є дані з інших корпоративних мереж. На імітаційному експериментальному стенді корпоративна мережа замінена хостом–джерелом.

Як центральний маршрутизатора був використаний Cisco 2811, підключений до внутрішньої мережі через порт FastEthernet, а до зовнішньої мережі – через порт з останньої розширеною версією операційної системи IOS advipservicesk версії 12.4, що підтримує всі методи QoS.

В якості маршрутизатора використаний Cisco 3750, підключений до внутрішньої мережі та зовнішньої мережі через порти GigabitEthernet. На даному маршрутизаторі створені сабінтерфейси з IP адресами з різних підмереж для кожного класу трафіка.

Для моделювання трафіка використовувалися динамічні моделі, побудовані за експериментальними даними [5].

Завдання динамічного управління завантаженням каналу зв'язку сформульована таким чином. Необхідно для кожного класу трафіка ($s = \overline{1,n}$) Побудувати закон управління us(T) на інтервалі [t1; t2] (горизонті прогнозу), де модель динаміки трафіка має вигляд динамічної системи:

$$\mathbf{x}^s(k+1) = A^s\mathbf{x}^s(k) + B^s u^s(k) + \Phi(\mathbf{x}^s(k)),$$
$$y^s = C^s{}_0(\mathbf{x}^s(k)),$$

при заданих критеріях:

$$\sum_{k=t_1}^{t_2}\left(\hat{y}^s(k) - y^s(k)\right)^2 \to \min,$$
$$\xi \geq y^s(k) > \eta,$$

де xs $\in$ Rn_s– ns –мірний вектор станів системи s; us$\in$ Us$\subset$R1 – управління; k – дискретний час; As, Bs – матриці; Ф (xs (k)) – нелінійна функція, ідентифікована за методом [5]; ys (k) – необхідна пропускна здатність; ys (k) – пропускна здатність каналу s в умовах обмежень;ξ,η– задані обмеження на ширину каналу.

2.3.3 Алгоритм динамічного управління

Перед нами стоїть завдання розробити алгоритм управління, що забезпечує підвищення надійності передачі та обробки інформації.

Укрупнено алгоритм складається з наступних етапів.

1. Моніторинг трафіка за типами.

2. Якщо завантаження каналу з будь–якого типу трафіка наближається до 70%, включається модуль динамічного управління.

3. Якщо прогнозне значення (на горизонт прогнозу) трафіка старшого пріоритету збільшується, то спостерігається збільшення ширини каналу для даного типу на прогнозне значення.

4. Якщо прогнозне значення (на горизонт прогнозу) трафіка старшого пріоритету зменшується, то спостерігається зменшення ширини каналу для даного типу на прогнозне значення.

5. Якщо відбувається збільшення трафіка і і і + 1 пріоритету, то спостерігається зменшення трафіка меншого пріоритету на величину прогнозу, але не більше ніж на критичне значення. Для трьох типів трафіка блок–схема приведена на рисунку 2.2.

У наведеній блок–схемі використані такі позначення: k – дискретний час; u1 (k) – пропускна здатність для першого типу трафіка; u2 (k) – пропускна здатність для другого типу трафіка; u3 (k) – пропускна здатність для третього типу трафіка; y2 (k) – прогнозована пропускна здатність для другого типу трафіка; y3 (k) – прогнозована пропускна здатність для третього типу трафіка; V – загальна пропускна здатність каналу; m – поточний коефіцієнт стиснення пропускної здатності для другого типу трафіка; l – поточний коефіцієнт стиснення пропускної здатності для другого типу трафіка.

В оперативному режимі відбувається перевірка прогнозованого значення зміни пропускної здатності трафіка другого типу за умови передачі трафіка третього типу в наступний момент часу без втрат. Пропускна здат-

ність другого типу трафіка збільшується на встановлений коефіцієнт максі стиснення для зменшення втрат трафіка третього типу.

Далі відбувається перевірка прогнозованого кроку зміни пропускної здатності за умови передачі трафіка третього типу на кожному кроці без втрат і не перевищення коефіцієнта максимального стиснення пропускної здатності для другого типу трафіка. В результаті встановлюються параметри пропускних спроможностей для всіх класів.

Алгоритм реалізований у вигляді програмного модуля програмної конфігурації мережі з наступному розділі атестаційної роботи.

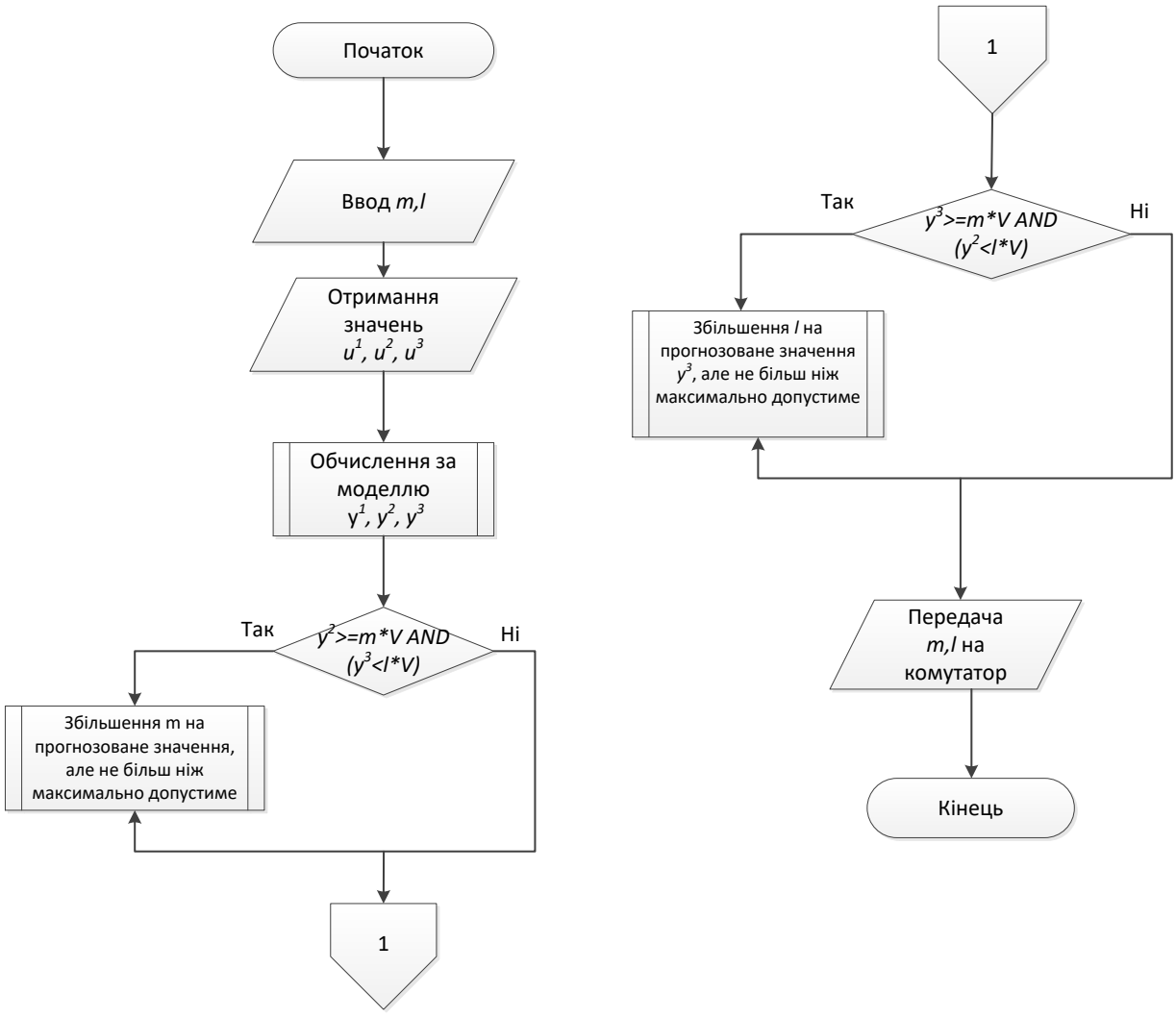

Рисунок 2.2 – Блок–схема алгоритму динамічного поділу пропускною спроможністю

2.3.4 Висновок

Отримані результати впровадження свідчать про ефективність використання динамічного управління в завантажених мережах, що використовують гібридні хмарні технології. Зокрема, воно дозволило функціонувати без затримок системам реального часу: відеоконференції, передача освітнього мультимедіа контенту в умовах великої кількості підключень.

2.4 Постановка задачі для загального експерименту й реалізації проекту

В рамках реалізації завдань проекту необхідно:

- Вибрати технологію оптимізації найбільш підходящу для існуючої мережі.
- З урахуванням вимог обраної технології, визначитися з обладнанням необхідним для розгортки оптимізатора.
- Після вибору апаратної частини, потрібно визначитися з необхідним ПЗ, з урахуванням вимог обраної технології.
- Ознайомитися з матеріалом (якщо такий є) по установці даної технології по оптимізації.
- З урахуванням існуючої мережі, нове апаратне забезпечення мережі вже до встановленої технології оптимізації трафіка WAN мереж, в розрив між маршрутизатором виходять в інтернет і локальним комутатором, оскільки дана технологія буде працювати в режимі моста.
- Додати необхідні настройки на маршрутизаторі і комутаторі, щоб вони пропускали і передавали трафік через цей міст, оскільки режим моста є прозорим, то при команді tracert (визначення маршруту) нашого заліза як хопу видно не буде. У разі вдалого проходження трафіка і працездатності мережі в цілому з новим доповненням, виконати все те ж на іншій машині і встановити в філії.

- Наступним завданням стає, проведення тестів, які перевіряють функціонування обраної технології, з метою переконатися, що вона працює і дає хороший показник в стиснення і швидкості передачі трафіка.

3 ЕКСПЕРИМЕНТАЛЬНА ЧАСТИНА МОДЕРНІЗАЦІЇ WAN МЕРЕЖІ

3.1 Мережеве програмне забезпечення TrafficSqueezer

Мережеве програмне забезпечення – це програмне забезпечення, що дозволяє організувати роботу користувача в мережі. Воно представлено загальним, системним і спеціальним програмним забезпеченням.

Загальна мережеве програмне забезпечення включає:

- браузер – програма перегляду веб–сторінок (наприклад, Internet Explorer). Браузер містить такі засоби: програму для роботи з електронною поштою (читання, створення, редагування і відправка поштових повідомлень); програму для роботи з сервером новин (підписка на групу новин, читання новин, створення і пересилання повідомлень), редактор тексту;

- HTML–редактори – редактори, призначені для створення веб–сторінок;

- графічні веб–засоби – засоби, призначені для оптимізації графічних елементів веб–сторінок;

- антивірусні мережеві програми – програми, використовувані для запобігання потрапляння програмних вірусів на комп'ютер користувача.

Перша технологія для оптимізації трафіка WAN мереж була обрана Traffic Squeezer. Це проект open source, і оскільки даний проект Traffic Squeezer підтримується виключно СА сімействі по Linux, для підняття був обраний дистрибутив Fedora 20.

Було встановлено ПО Fedora 20, для переважної розвантаження на процесор дистрибутив був встановлений без графічного інтерфейс. Всі настройки і установки проходили безпосередньо з командного рядка Fedora. Першим кроком було скачування проекту Traffic Squeezer на нашу систему Linux, і установки необхідних програм таких як база даних MySql, висновок веб інтерфейсу Apache і т.д. Після повної установки всіх необхідних утиліт самого

проекту, необхідна була компіляція ядра нашого ПО Linux і Traffic Squeezer. Все це було зроблено за допомогою наступних команд з командного рядка (приклад 3.1).

```
      (Установка mc)
      yum install mc;
      (Завантаження Traffic Squeezer Cotton Candy)
      wget
http://sourceforge.net/projects/trafficsqueezer/files/trafficsqueezer-cotton-
candy-3.12.5-12_aquarium-geometry-cotton-candy_7.00.4.tar.xz / download;
      (Перейменувати скачаний архів)
      mv  download  trafficsqueezer-cotton-candy-3.12.5-12_aquarium-geometry-
cotton-candy_7.00.4.tar.xz;
      (Розпакувати скачаний архів)
      tar    -xvf    trafficsqueezer-cotton-candy-3.12.5-12_aquarium-geometry-
cotton-candy_7.00.4.tar.xz;
      (Починається установка)
      cd trafficsqueezer-cotton-candy-3.12.5-12;
      cd linux-3.12.5;
      make menuconfig;
      yum install ncurses-devel; (Знову повторити)
      make menuconfig;
      (В сплив вікні вибрати exit)
      cat c;
      catc_slow;
      ./c_slow;
      (Почалася компіляція з ядром Linux) (Перейти в директорію aquarium і ви-
конуємо)
      ./make_clean;
      ./make_install;
      yum –y remove iptables;
      yum install phpmyadmin;
      yum install mysql-server; (Запуск apahce)
      systemctl start httpd.service; (Запуск mysql)
      systemctl start mysqld.service; (Зайти в базу даних)
      mysql –u root –p; (Створити нову базу)
      create database aquarium; (Змінити пароль)
      set password for 'root' @ 'localhost' = password ( 'qwerty123');
      (В каталозі aquarium_cotton-candy_7.00.4 / saas_gui / db_saas в скрипті
dbscripts.sql додати рядок)
      USE aquarium; (Зайти в директорію)
      myaquarium_cotton-candy_7.00.4 / saas_gui / db_saass; (Прописати)
      mysql –u root –p <./dbscripts.sql; (Перезапуск всієї системи)
      shutdown –r now;
```

Приклад 3.1 – Компіляція ядра ПО Linux і Traffic Squeezer

Після перезапуску системи, всі встановлені утиліти стартують автоматично, але для повної переконливості перевіряємо це за допомогою команди pstree побачити повинні в дереві запущених процесів такі як mysql, apache, aquariumd. Система готова, тепер для подальшої настройки потрібно перейти в веб інтерфейс (рисунок 3.1).

Рисунок 3.1 – Меню авторизації веб інтерфейсу aquarium GUI

Ввести стандартний пароль для користувача root–admin. Після чого він пускає в головне меню веб інтерфейсу проекту TrafficSqueezer. Потрібно знайти майстра настройки.

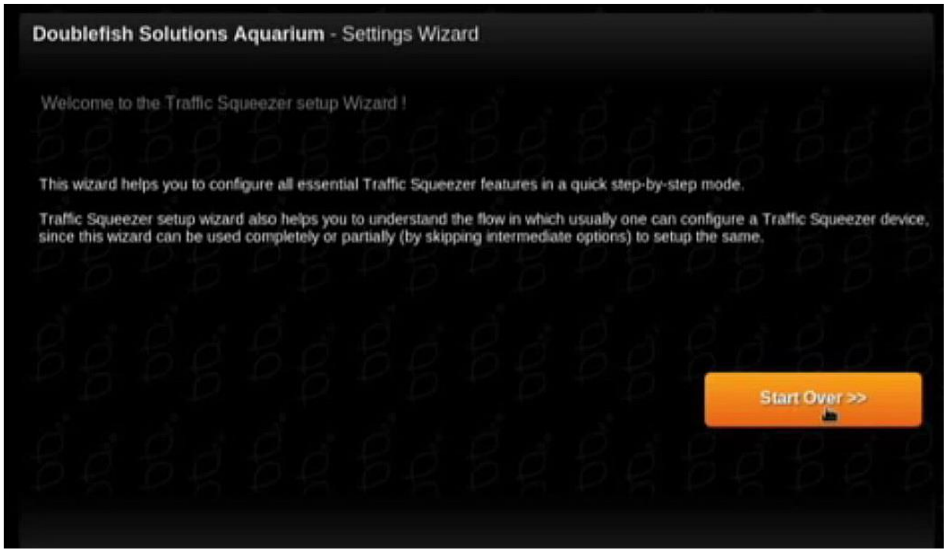

Рисунок 3.2 – Меню запуску помічника налаштування

Після кліка на start over (рисунок 3.2) установник пропонує нам мінімальні необхідні настройки для того, щоб вся система почала функціонувати як оптимізатор трафіка.

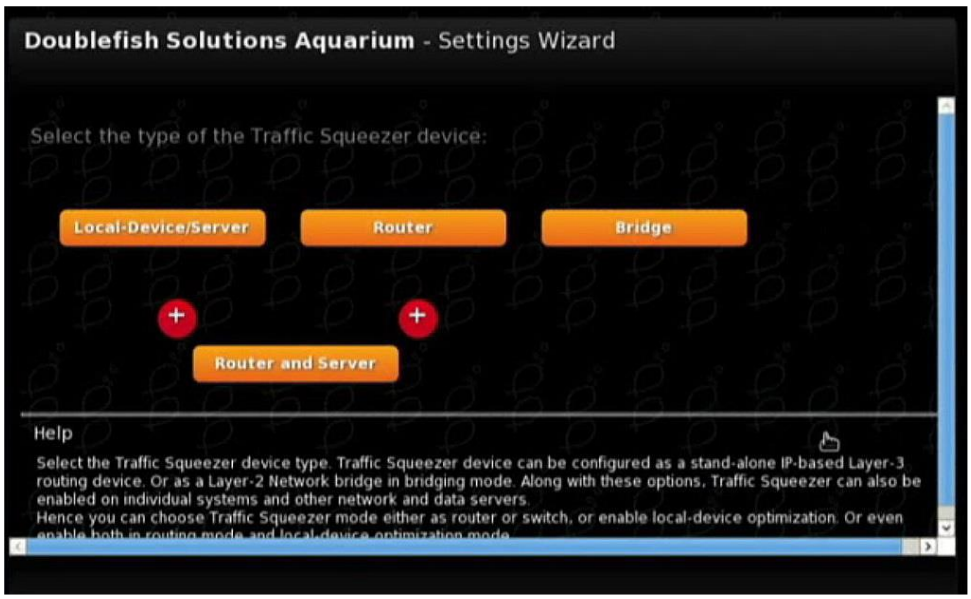

Рисунок 3.3–Вибір необхідного режиму роботи

Потрібно вибрати Bridge (рисунок 3.3), оскільки оптимізатор і буде власне мостом між головним офісом і філією.

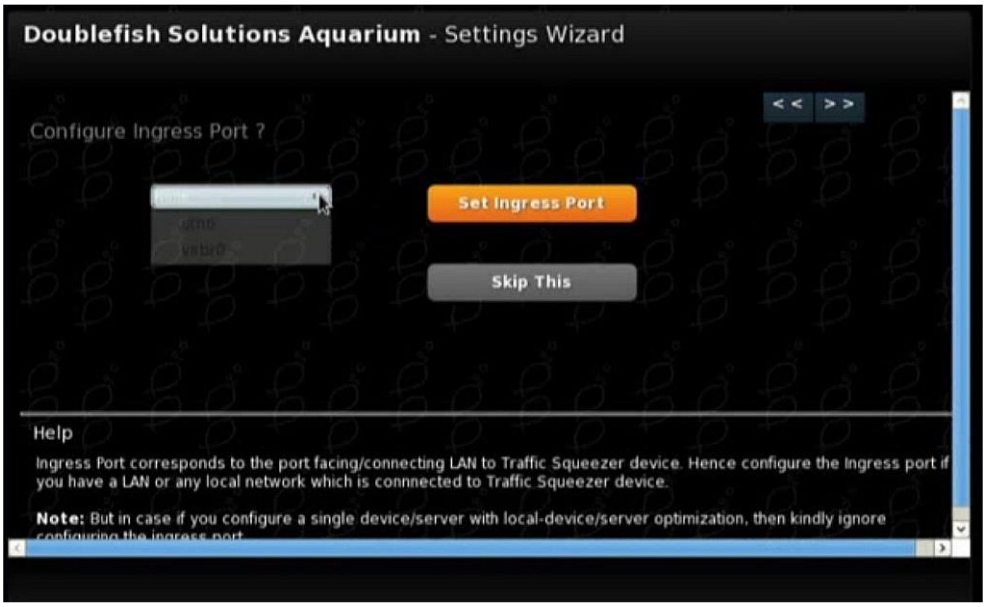

Рисунок 3.4 – Меню завдання порту на вхідний трафік

На порт Ingress (рисунок 3.4) необхідно вказати той інтерфейс, який підключений до локального комутатора. На який буде входити локальний трафік. У нашому випадку це інтерфейс eth0.

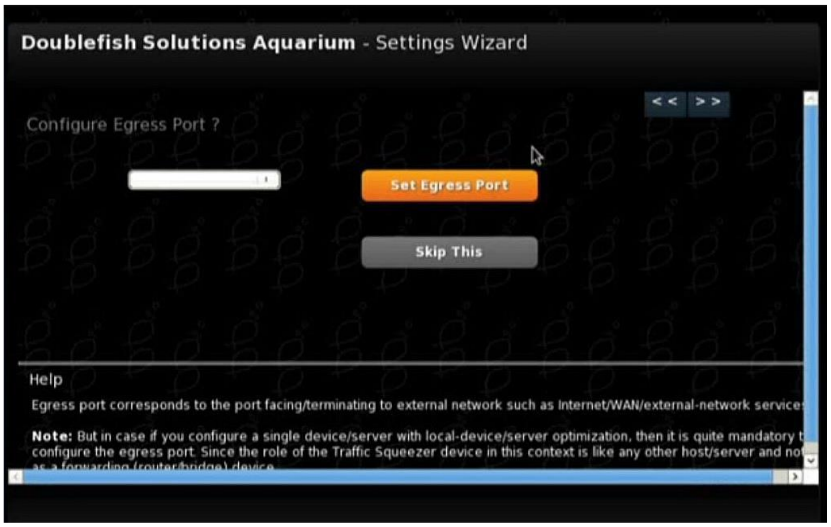

Рисунок 3.5 – Меню завдання порту на вихідний трафік

На порт Egress (Рисунок 3.5) необхідно вказати той інтерфейс, який підключений до маршрутизатора, який дивиться в інтернет, і з якого вже буде виходити оптимізований трафік на маршрутизатор. У нашому випадку це інтерфейс eth1. Наступним пунктом настройки є вибір безпосередньо самої оптимізації трафіка.

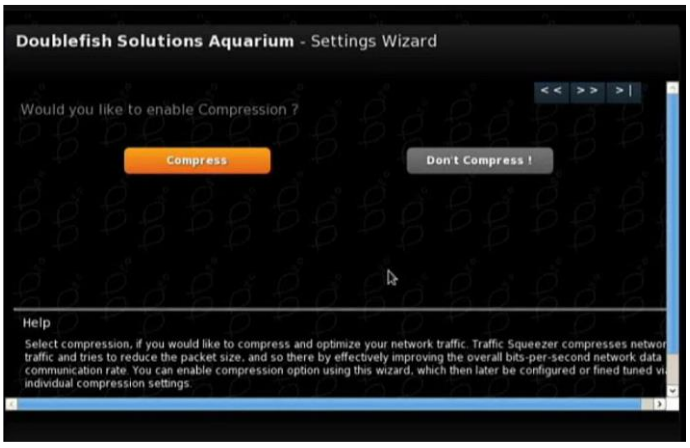

Рисунок 3.6– Меню вибору опції оптимізації

Клік підтвердження опції оптимізації(Рисунок 3.6), і це є останнім кроком для налаштування нашого оптимізатора. Тепер все готово, слід виконати цю ж роботу з другим оптимізатором Traffic Squeezer, який буде встановлений в філії.

Переходимо до тестування. Тестування передбачає собою, передачу різних документів різних форматів спочатку без участі нашого оптимізатора, і заміри швидкості і часу передачі, а після вже з включенням в нього режиму компресії.

Рисунок 3.7 – Заключний етап майстра настройки оптимізатора

Пробні тести передачі файлів в робочий час, була спроба передачі файлів розширенням .doc, .xls, .txt. При першому тестуванні без оптимізатора результат був.

Таблиця 3.1 – Результати тестування без оптимізатора

Величина що вимірюється	З філії в головний	З головного офісу до філії
Формат	.doc	.doc
Розмір	30 Мб	30 Мб
Швидкість	1,7 Мбіт / сек	560 Кбіт / сек
Час	22 сек	1 хв 7 сек

Після включення наших двох оптимізаторів в мережу, ми виконали аналогічне тестування.

Таблиця 3.2 – Результати тестування з 2ма оптимізаторами

Величина що вимірюється	З філії в головний офіс	З головного офісу до філії
Формат	.doc	.doc
Розмір	30 Мб	30 Мб
Швидкість	1,8 Мбіт / сек	700 Кбіт / сек
Час	20 сек	1 хв

Дані тести проводилися тричі, не один не показав потрібних результатів в прирості не в прирості продуктивності мережі ні в прискоренні.

3.2 Мережеве програмне забезпечення Wanos

Оскільки результати тестування обраної раніше технології Traffic Squeezer не дали, потрібних результатів, ми вибираємо наступний open source проект Wanos.

Проект Wanos поширюється у вигляді образу в форматі OVA і VMDK. Дані формати образів відкриваються за допомогою віртуальних машин VMware. Можна розгорнути проект Wanos на VMware Workstation або на VMware Player.

Але оскільки VMware Workstation є платною. До першого тестування купувати даний софт не доцільно, а VMware Player не дає можливості конфігурації порядку інтерфейсів, тому ми вибираємо варіант розгортання з USB на "голе" залізо. Тобто без пред установки будь–якого програмного забезпечення. Для того, щоб зробити USB флеш накопичувач завантажувальним, скористаємося програмою Win32 Disk Imager.

Потрібно образ проекту Wanos перетворити USB флеш накопичувач в завантажувальний USB (рисунок 3.8).

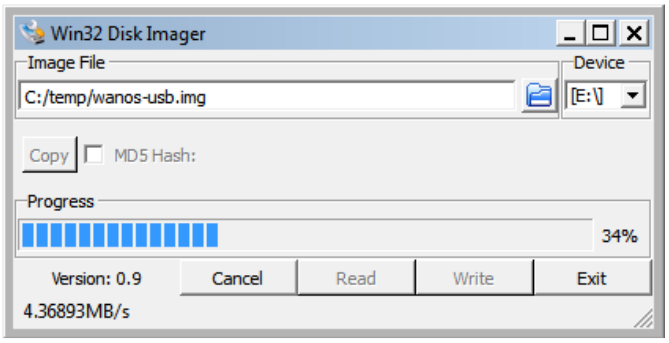

Рисунок 3.8 – Перетворення USB флеш накопичувача

Після завершення перетворення, можна вставляти флеш накопичувач в USB порт системного блоку.

Після запуску системи запускається невелика операційна система під назвою Tiny Core Linux.

Tiny Core Linux– мінімалістичний дистрибутив GNU / Linux, мета якого забезпечення базової системи з використанням BusyBox, FLTK і іншого легкого програмного забезпечення. Розмір дистрибутива близько 10 мегабайт. Це система призначена в основному, для того щоб максимально знизити навантаження з процесора машини, для більш коректної і швидкої оптимізації трафіка. Вона розуміє мінімальний набір команд Linux. Тому всі основні конфігурації можливо виконати за допомогою веб інтрефейса.

Мережеві налаштування за замовчуванням:

- IP Address: 192.168.1.200;
- Mask: / 24;
- Gateway: 192.168.1.1;
- Mode: Edge;
- Web User / Pass: wanos / wanos;
- SSH User / Pass: tc / ChangeM3;

В консолі потрібно поміняти default ip address на потрібний для мережі компанії, і вже по ньому можна перейти в веб інтерфейс (Рисунок 3.9).

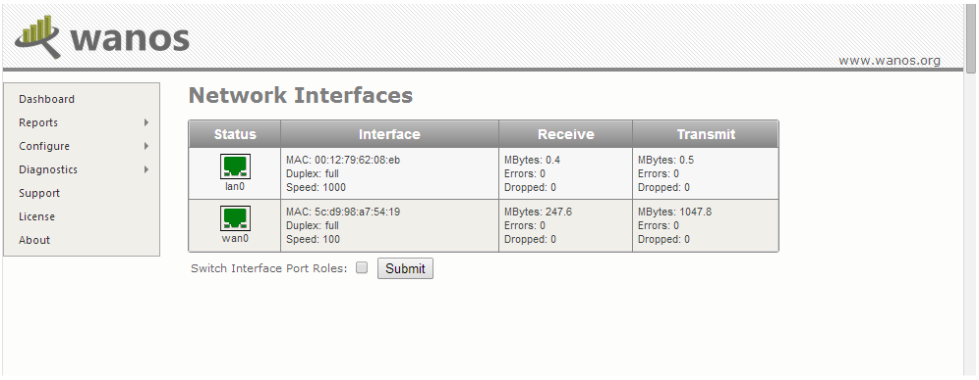

Рисунок 3.9 – Перевірка правильності підключення мережевих інтерфейсів меню налаштувань в в веб інтерфейсі

Після того як переконалися, що інтерфейси активні, показані зеленим значком. Можна продовжувати основні настройки, для того щоб наш оптимізатор заробив. Як і в попередньому сценарії, оптимізатор встановлюється в розрив між локальних комутатором і маршрутизатором, що «дивиться» у інтернет. Також він працює в режимі моста.

Першими настройками оптимізатора будуть вибір режиму в якому він буде працювати. Їх 2 це режим Core і Edge. У головному офісі ставимо режим Core (рисунок 3.11).

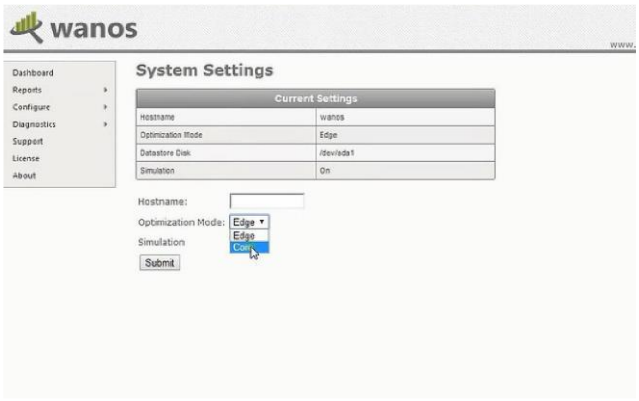

Рисунок 3.10 – Меню налаштувань вибору настройки режиму роботи Core

На 2–ій машині, яка знаходиться у філії, ми вибираємо режим Edge (рисунок 3.11). Оскільки основні настройки слід виконувати з режимом Core, а вона знаходиться в головному офісі.

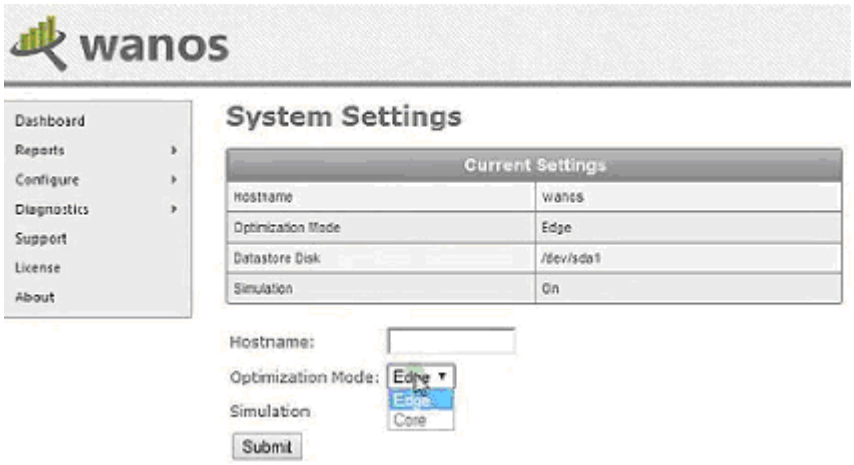

Рисунок 3.11 – Меню налаштувань вибору режиму роботи Edge

Далі проходимо в меню налаштувань політик трафіка, для того щоб вказати з якою підмережі піде трафік і в яку всі настройки політик трафіка налаштовуються, тільки на машині з режимом Core.

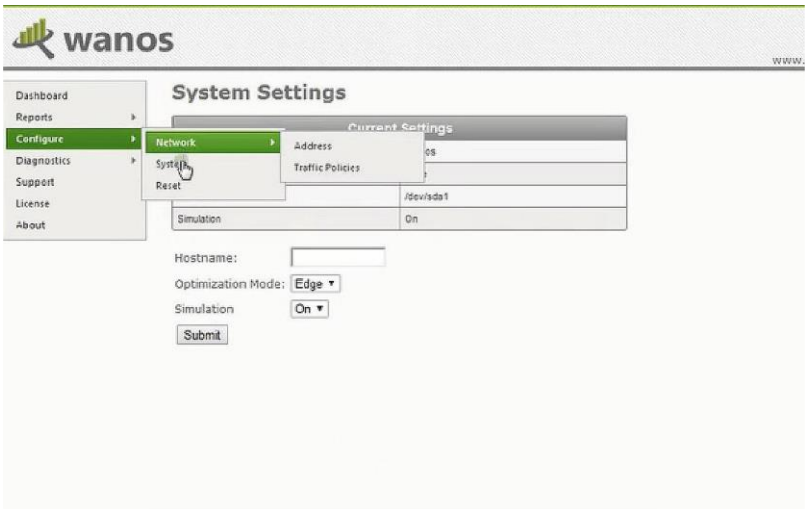

Рисунок 3.12– Вкладка налаштувань політик трафіка

Після установки інтерфейсів, режимів роботи, вказівки політик трафіка і установки наших машин в філії та головному офісі. Необхідно переконатися, що трафік до застосування політик, проходить без оптимізації, тобто на нинішній момент наші оптимізатори створили прозорі міст. Застосовуємо політики і переходимо до тестів.

Оскільки Core машина стоїть в головному офісі, поему destination address ми вказуємо підмережа філії.

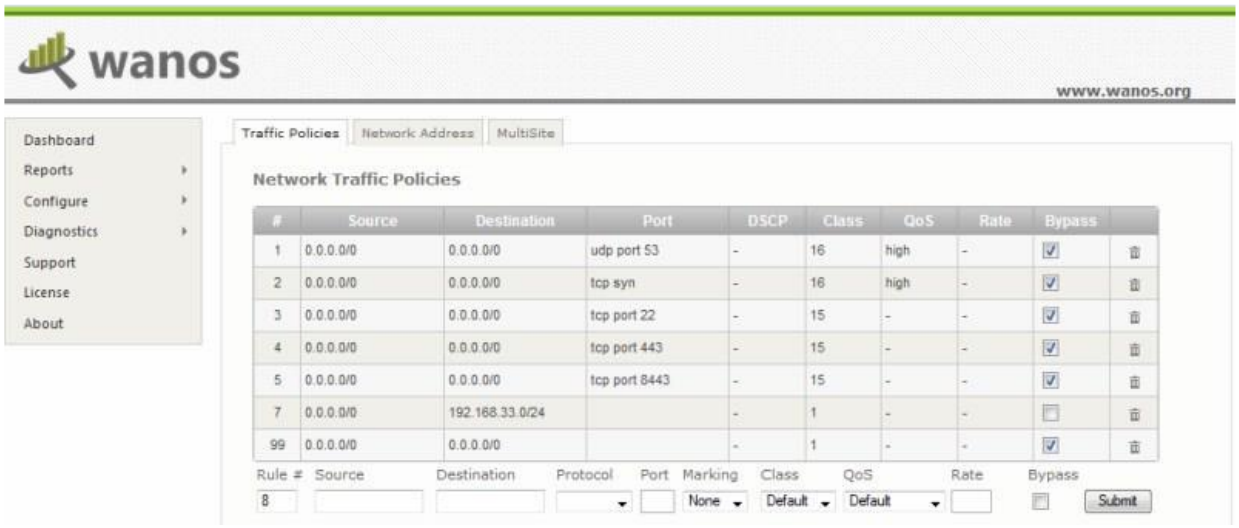

Рисунок 3.13 – Приклад налаштувань політик трафіка на машині Core, з вказаною підмережею філії 33.0

Після установки інтерфейсів, режимів роботи, вказівки політик трафіка і установки наших машин в філії та головному офісі. Необхідно переконатися, що трафік до застосування політик, проходить без оптимізації, тобто на нинішній момент наші оптимізатори створили прозорі міст. Застосовуємо політики і переходимо до тестів.

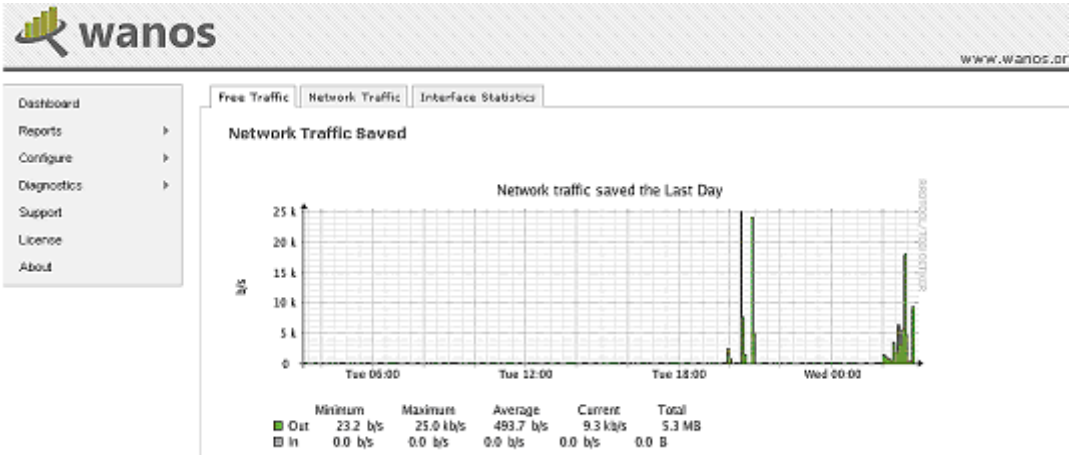

Рисунок 3.14 – Початковий графік проходить трафіка, після застосування всіх політик

Найкраще підходить для стиснення формати .txt / .doc / .xls з основною повторюваних рядків, наприклад, все 0 або 1. В цьому випадку може показати дуже хороший результат, як стиск на 90%

3.2.1 Перший тест без оптимізатора Wanos

Передаємо файл в розпалі робочого дня, для того щоб побачити реальну картину навантаження на канал і швидкості документообігу.

Таблиця 3.3 – Результати тестування навантаження на канал і швидкості документообігу

Величина що вимірюється	З філії в головний офіс	З головного офісу в філію
Формат	.doc	.doc
Розмір	37 Мб	37 Мб
Швидкість	1,6 Мбіт / сек	550 Кбіт / сек
Час	24 сек	1хв 9 сек

Для точності вибрали інший формат файлу.

Таблиця 3.4 – Результати тестування

Величина що вимірюється	з філії в головний офіс	з головного офісу до філії
Формат	.xlsx	.xlsx
Розмір	35 Мб	35 Мб
Швидкість	1,7 Мбіт / сек	600 Кбіт / сек
Час	22 сек	1 хв

3.2.2 Тест з включеним оптимізатором Wanos

Для точності вибрали інший формат файлу, результати тестування показані в таблицях 3.5–3.6.

Таблиця 3.5 – Результати тестування

Величина що вимірюється	З філії в головний офіс	з головного офісу до філії
Формат	.doc	.doc
Розмір	37 Мб	37 Мб
Швидкість	2 Мбіт / сек	740 Кбіт / сек
Час	16 сек	50 сек

Таблиця 3.6 – Результати тестування

Величина що вимірюється	З філії в головний офіс	з головного офісу до філії
Формат	.xlsx	.xlsx
Розмір	35 Мб	35 Мб
Швидкість	2,2 Мбіт / сек	750 Кбіт / сек
Час	15 сек	48 сек

Як видно з результатів великої різниці в режимі роботи з оптимізатором немає. Найкраще для дедуплікації, коли той же самий файл відправляється вдруге. Наприклад, взяти 10 Мб. Поштовий файл і відправити глобальної мережі. Перший екземпляр повинен бути поруч швидкості WAN, оскільки він вже стиснутий. Другий примірник повинен дати 4–10х швидкості пропускної здатності. Пробуємо проробити те ж, з тими ж документами, результати показані в таблицях 3.7–3.8.

Таблиця 3.7– Результати тестування

Величина що вимірюється	З філії в головний офіс	З головного офісу до філії
Формат	.doc	.doc
Розмір	37 Мб	37 Мб
Швидкість	1,6 Мбіт / сек	630 Кбіт / сек
Час	21 сек	1 хв

Таблиця 3.8 – Результати тестування

Вимірювана величина	З філії в головний офіс	З головного офісу до філії
Формат	.xlsx	.xlsx
Розмір	35 Мб	35 Мб
Швидкість	1,8 Мбіт / сек	670 Кбіт / сек
Час	219сек	56 сек

З усіх проведених тестів в результаті як такого очікуваного приросту швидкості передачі і звільнення смуги пропускання немає. Wanos працює, за рахунок зниження трафіка, що передається між ядром і прикордонних пристроїв. Зменшення трафіка зазвичай переводить до збільшення швидкості. У нашому випадку ми зробили тести ще кілька разів з FTP сервера. Зашифро-

ваний трафік (SSL, HTTPS, підпис SMB) Wanos НЕ оптимізує, тобто не забезпечить зниження або прискорення.

Під такими можливими видами оптимізації може забезпечить скорочення трафіка, але не обов'язково пропускне прискорення. Швидкість ядра локальної мережі близька або менше швидкості WAN або іншими словами швидкість WAN близька або більше швидкості ядра локальної мережі. Трафік не може передаватися швидше, ніж інформація в основній мережі. Швидкості випробування проходять швидше, ніж обладнання може їх обробляти.

3.3 Мережеве програмне забезпечення Riverbed

Після випробувань вже 2х сценаріїв open source по оптимізації трафіка WAN мереж, які не дали необхідних результатів, ми пробуємо випробувати триальное рішення від Riverbed.

Для того, щоб завантажити образ програмного забезпечення Riverbed, необхідно зареєструватися на офіційному сайті і вказати компанію, в якій буде проходити тестування. Після скачування образу, ми встановлюємо операційну систему Windows 10 professional, і VMware player і VMware Workstation. Він необхідний для того, щоб запустити Riverbed оскільки образ його в форматі віртуальних машин.

Після налаштувань наших інтерфейсів, інші настройки вже будемо робити за допомогою веб інтерфейсу. Inpath це наш віртуальний інтерфейс, і щоб він працював в правильному напрямку потрібно налаштувати правило для нього (рисунки 3.15–3.16).

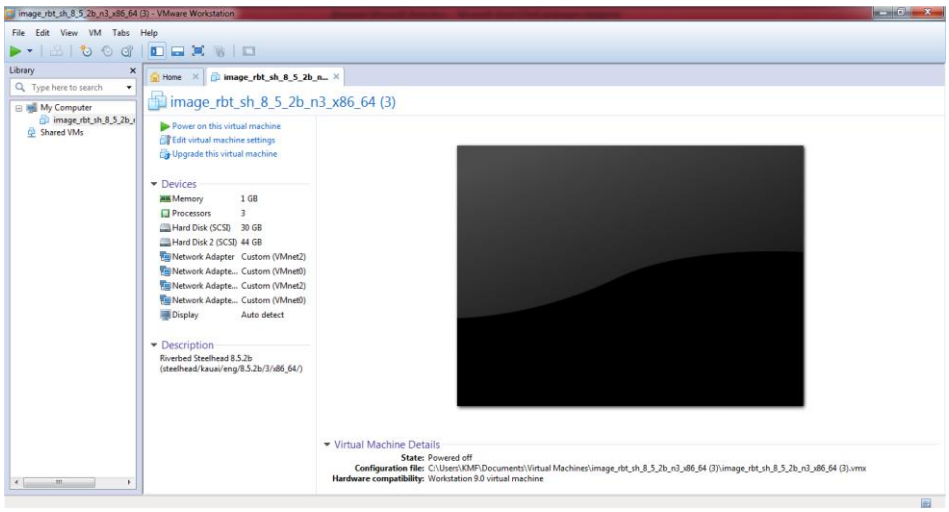

Рисунок 3.15 – Меню встановленої віртуальної машини Riverbed в VMware Workstation 10

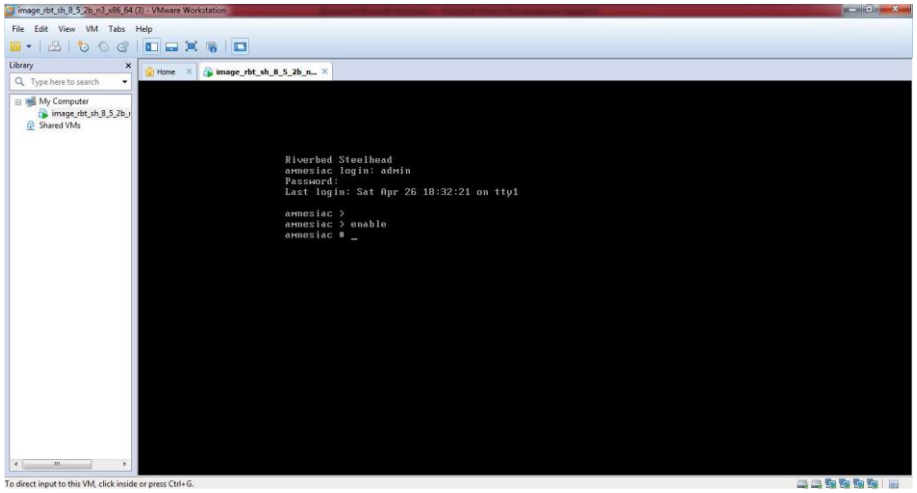

Рисунок 3.16 – Запуск віртуальної машини, для початкового налаштування

Після налаштувань наших інтерфейсів, інші настройки вже будемо робити за допомогою веб інтерфейсу. Inpath це наш віртуальний бріджовий інтерфейс, і щоб він працював в правильному напрямку потрібно налаштувати правило для нього (рисунок 3.17).

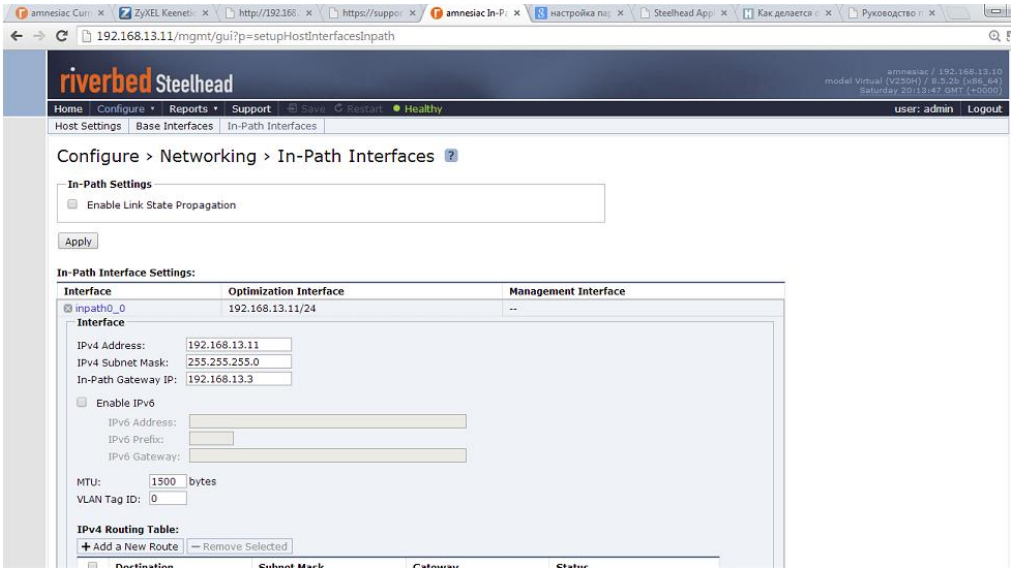

Рисунок 3.17 – Налаштування Inpath інтерфейсу

Після настройки бридж інтерфейсу необхідно включити його (Рисунок 3.18).

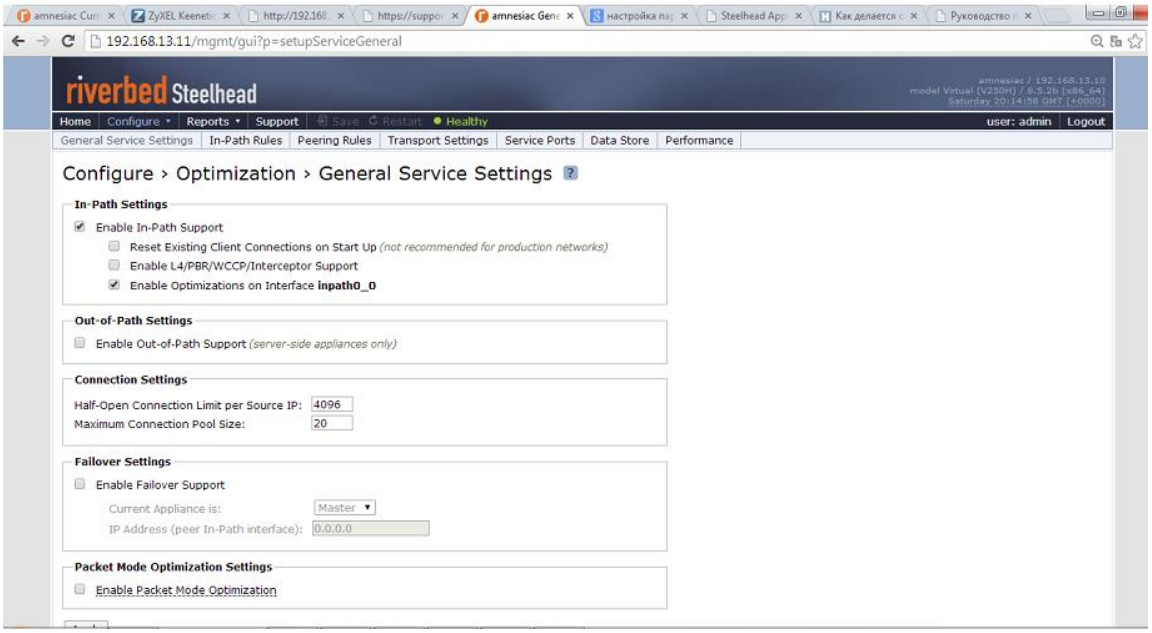

Рисунок 3.18 – Включення правило для інтерфейсу Inpath

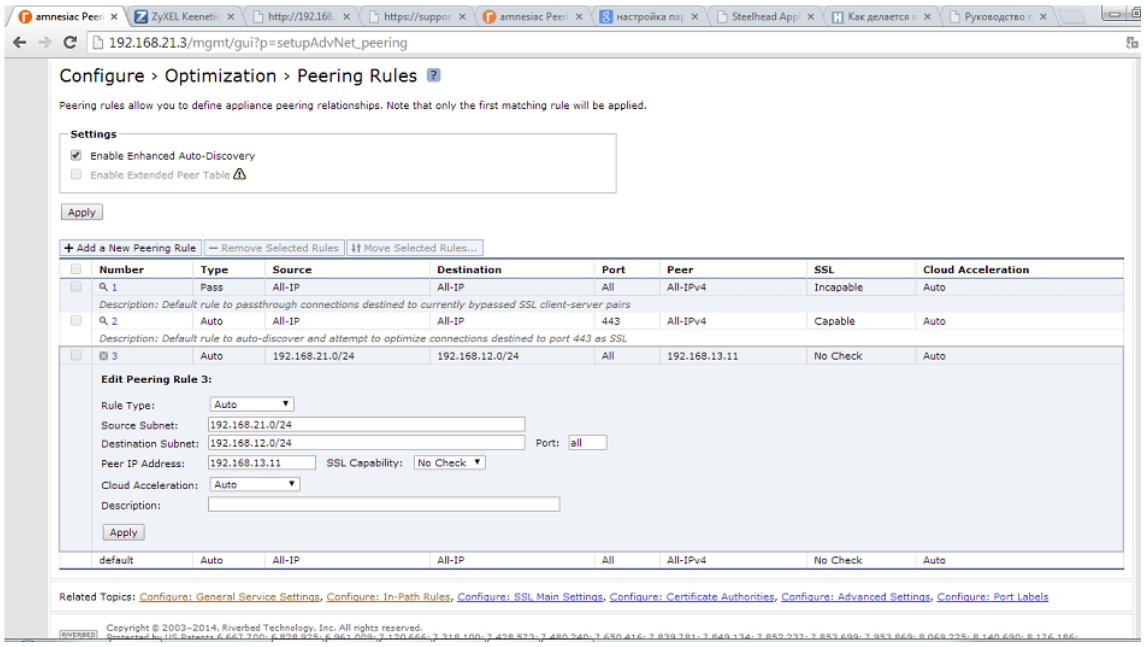

Рисунок 3.19 – Створення правила для бенкетів, трафік в напрямку з ГО до філії

Основні параметри налаштовані, тепер переходимо до тестування.

3.3.1 Перші тести без оптимізатора

Передаємо файл в середині робочого дня, для того щоб побачити реальну картину навантаження на канал і швидкості. Результати показані в таблицях 3.9–3.10.

Таблиця 3.9 – Результати тестування

Величина що вимірюється	З філії в головний офіс	З головного офісу до філії
Формат	.xls	.xls
Розмір	3,6 Мб	3,6 Мб
Швидкість	480 Кбіт / сек	520 Кбіт / сек
Час	2 хв 50	2 хв 40

Таблиця 3.10 – Результати тестування

Вимірювана величина	З філії в головний офіс	З головного офісу до філії
формат	.exe	.exe
Розмір	4,9 Мб	4,9 Мб
швидкість	520 Кбіт / сек	480 Кбіт / сек
час	3 хв 37 сек	3 хв 57 сек

При проведенні цих тестів, при передачі з ГО у філію ми також проводили моніторинг як мережі в цілому так і окремого інтерфейсу. За допомогою ПО Zabbix. Він показав нам навантаження каналу (рисунки 3.20–3.21), в момент передачі наших файлів.

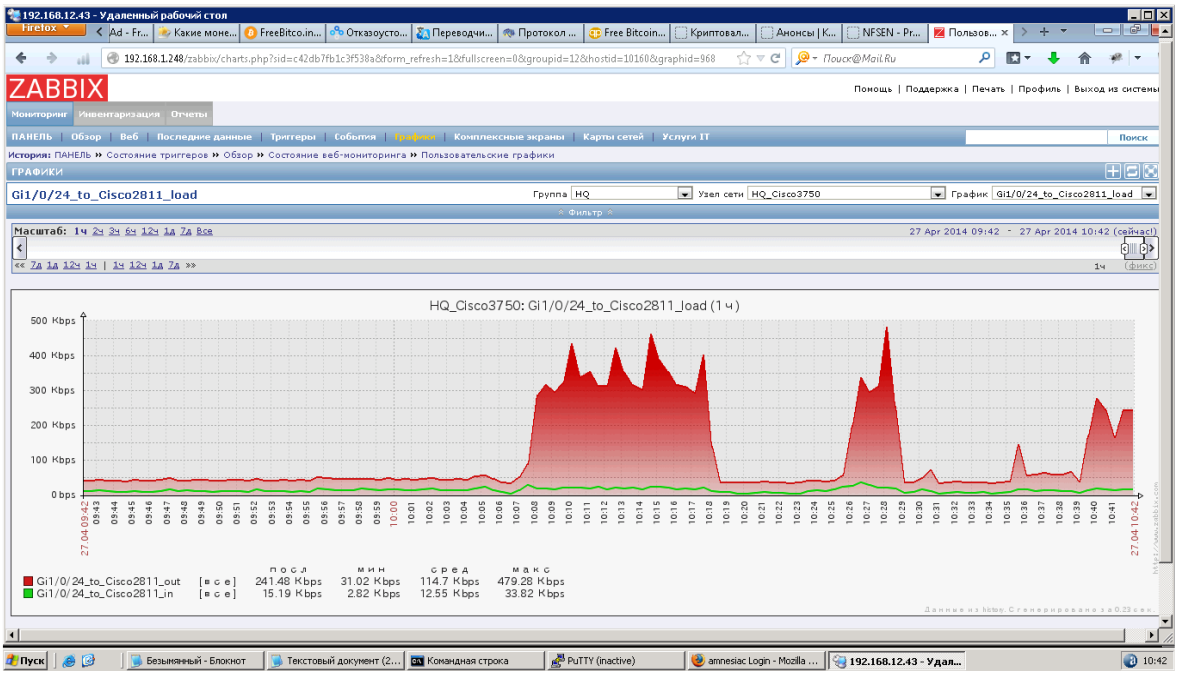

Рисунок 3.20 – Графік повного навантаження каналу при передачі файлу .xls розміром 3,6 Мб

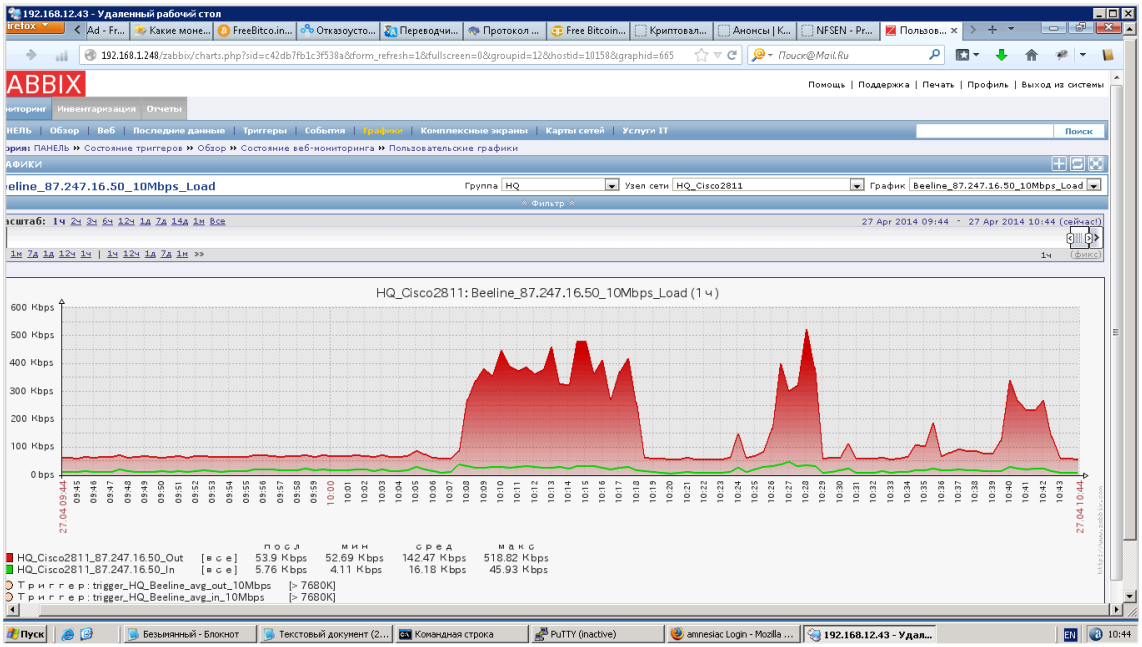

Рисунок 3.21 – Графік показує вихідний трафік з локальної мережі в WAN

Як видно з графіків (малюнки 3.22–3.23), вони абсолютно однакові, оскільки оптимізатор не був включений. Тепер подивимося, що відбувалося з каналом при нашій 2ій передачі файлу.

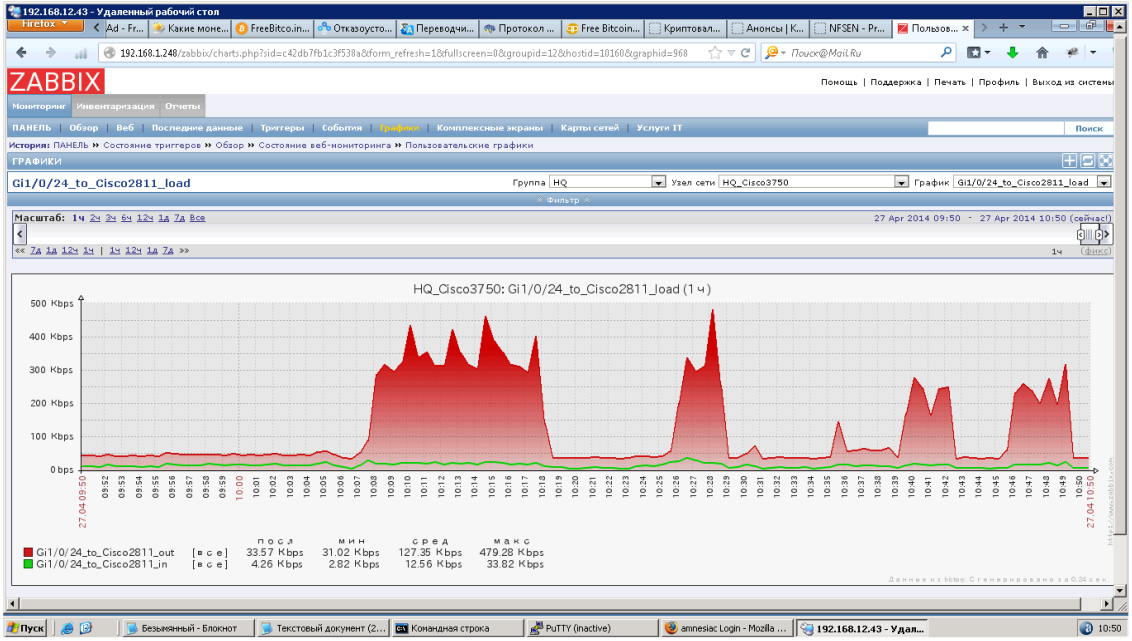

Рисунок 3.22 – Графік завантаження каналу при передачі файлу .exe розміром 4,9 Мб

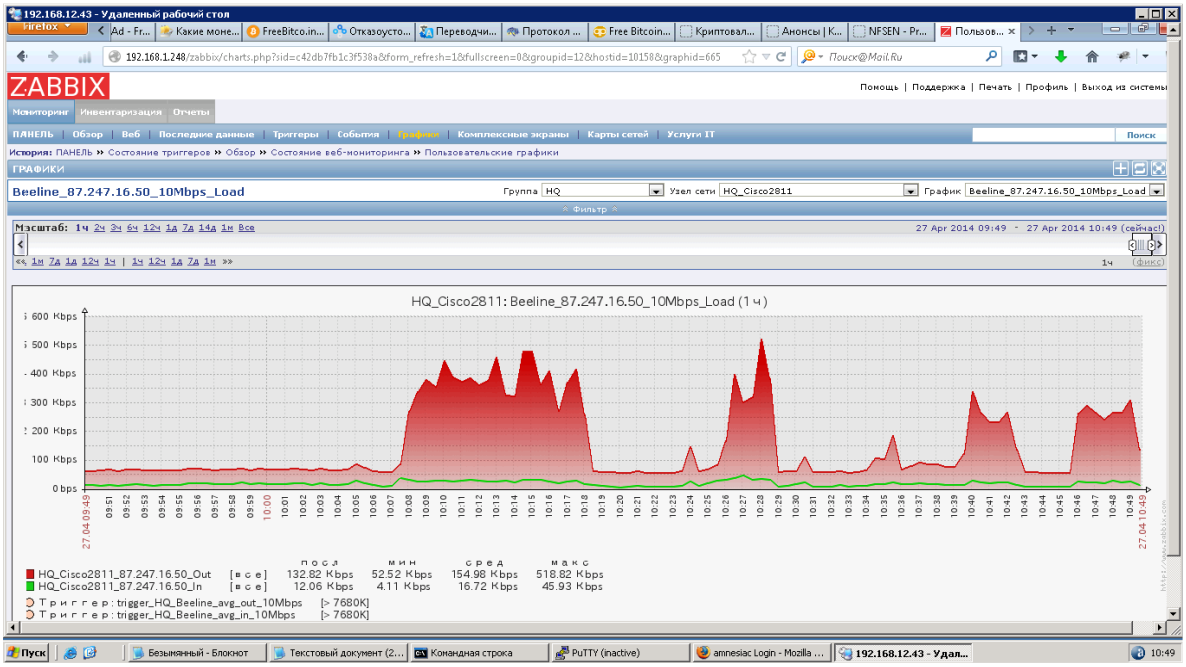

Рисунок 3.24 – Графік показує вихідний трафік з локальної мережі в WAN, при передачі файлу .exe розміром 4,9 Мб

3.3.2 Повтор тестів з включеним оптимізатором

Передаємо файл той же самий в файл формату .xls розміром 3,6 Мб, починаємо передачу його з головного офісу в філія, вже з включеним і налаштованим в мережу оптимізатором Riverbed. Перший тест з оптимізатором буде в холодному режимі, оскільки файлу ще немає в data store.

Таблиця 3.11 – Результати тестування

Величина що вимірюється	З філії в головний офіс	З головного офісу до філії
Формат	.xls	.xls
Розмір	3,6 Мб	3,6 Мб
Швидкість	480 Кбіт / сек	520 Кбіт / сек
Час	2 хв 50	2 хв 40

Таблиця 3.12 – Результати тестування

Величина що вимірюється	З філії в головний офіс	З головного офісу до філії
Формат	.exe	.exe
Розмір	4,9 Мб	4,9 Мб
Швидкість	520 Кбіт / сек	480 Кбіт / сек
Час	3 хв 37 сек	3 хв 57 сек

Подивимося, що відбувалося з каналом нашого головного офісу в момент передачі файлів, з включеною оптимізацією WAN мереж (рисунки 3.25–3.26).

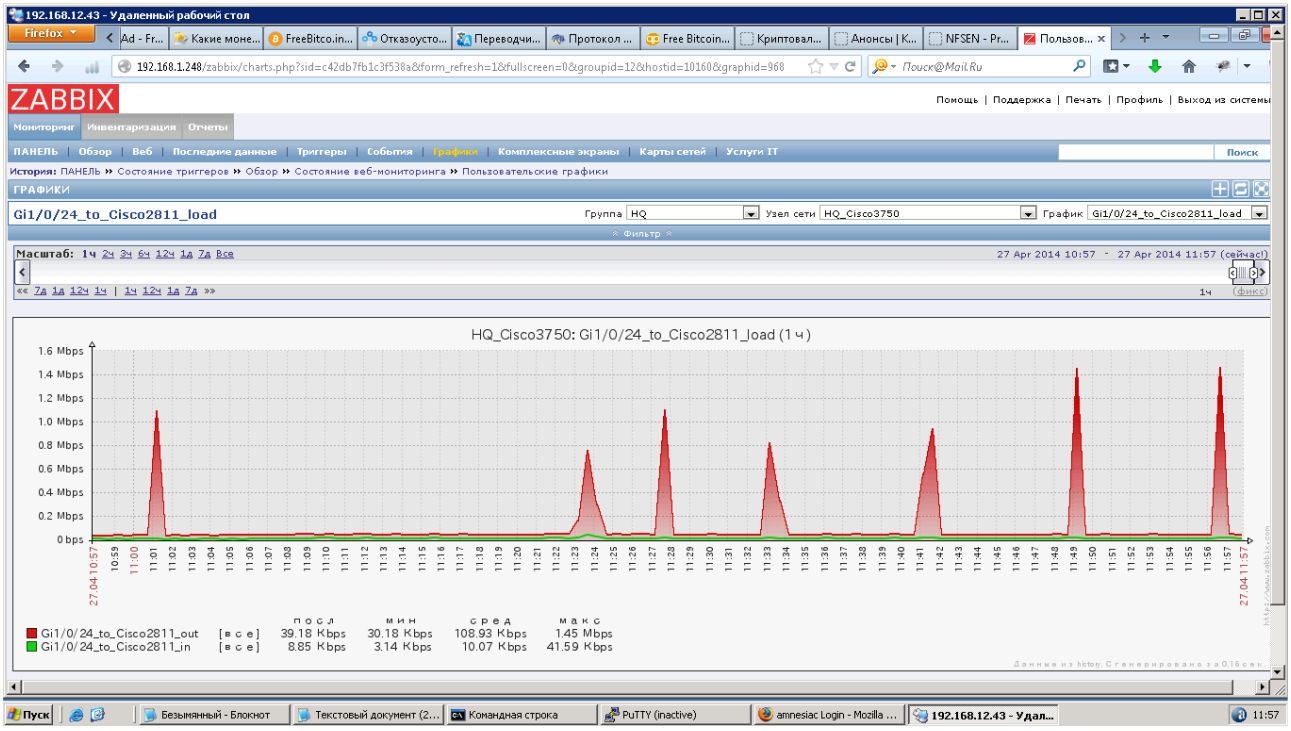

Рисунок 3.25 – Графік навантаження каналу при передачі файлу .exe розміром 4,9 Мб з включеною оптимізацією

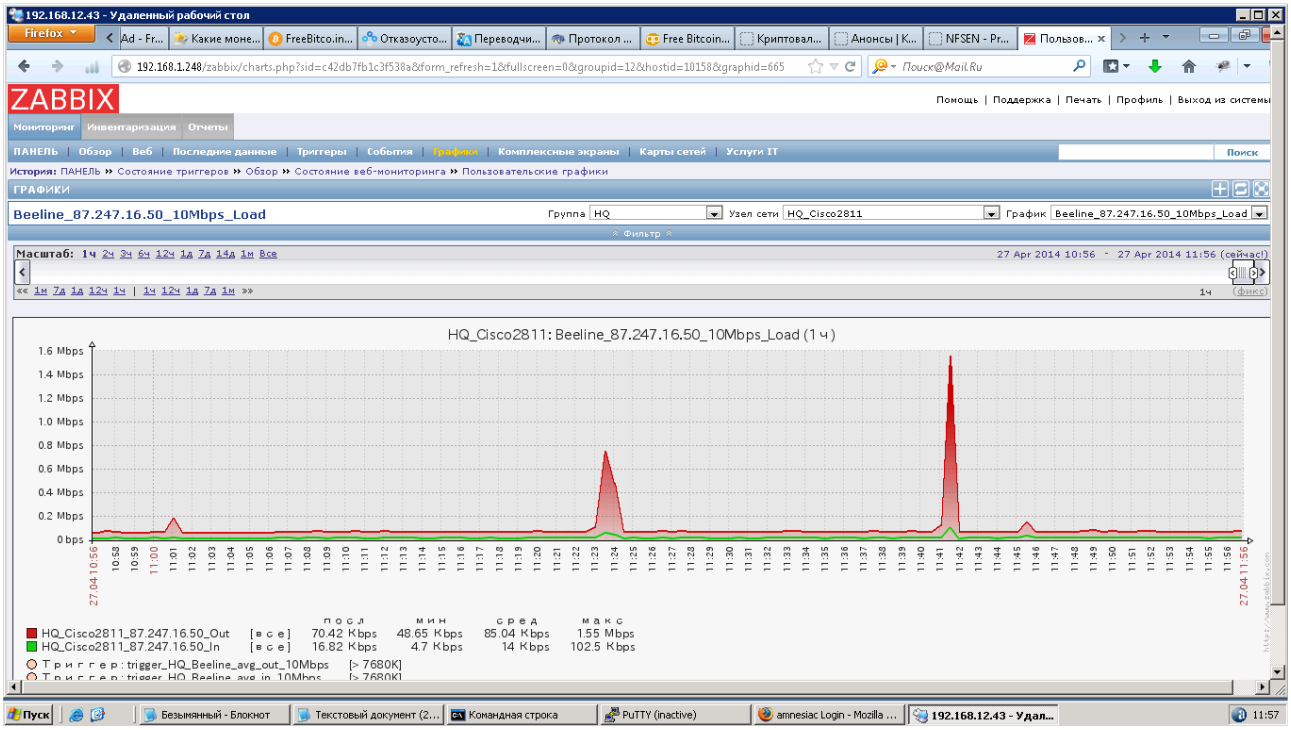

Рисунок 3.26 – Графік показує стислий вихідний трафік з локальної мережі в інтернет

3.3.3 Тест з відправкою файлів з філії в головний офіс і назад

Перший тест з оптимізатором буде в холодному режимі, оскільки файлу ще немає в data store. Результати представлені в таблицях 3.13–3.14.

Таблиця 3.13 – Результати тестування

Величина що вимірюється	З філії в головний офіс	З головного офісу до філії
Формат	.doc	.doc
Розмір	7,6 Мб	7,6 Мб
Швидкість	1,2 Мбіт / сек	1,6 Мбіт / сек
Час	43 сек	23

Таблиця 3.14 – Результати тестування

Величина що вимірюється	З філії в головний офіс	З головного офісу до філії
Формат	.exe	.exe
Розмір	4,9 Мб	4,9 Мб
Швидкість	520 Кбіт / сек	480 Кбіт / сек
Час	3 хв 37 сек	3 хв 57 сек

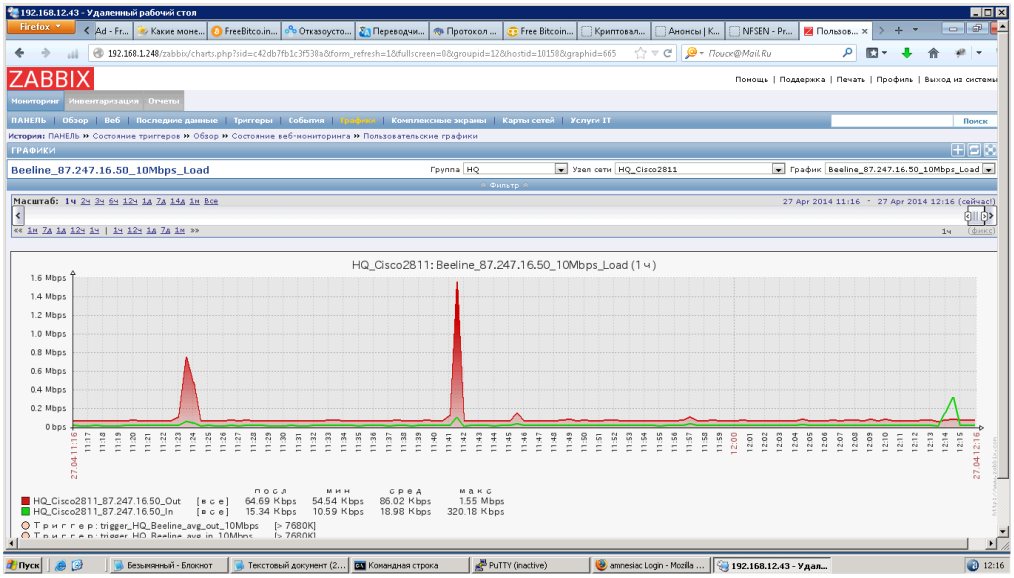

Рисунок 3.27 – Графік показує стислий вхідний трафік з інтернету в локальну мережу

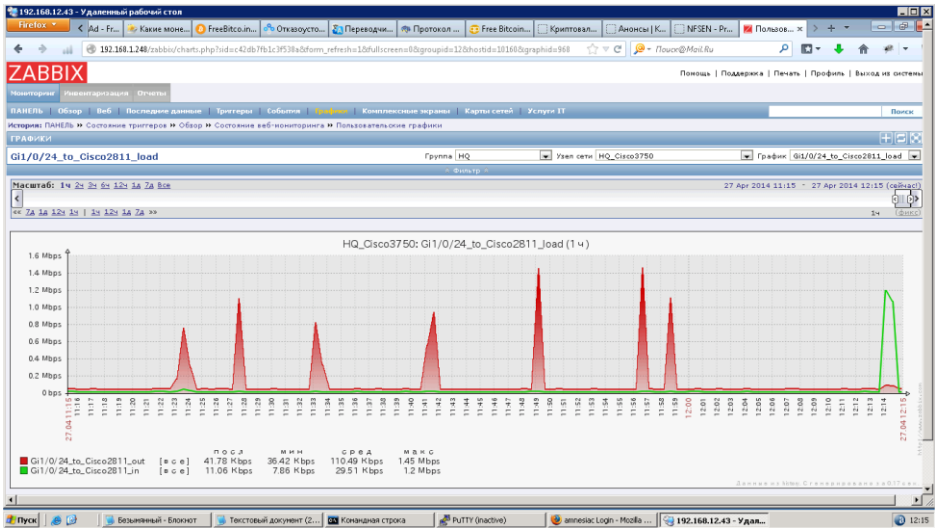

Рисунок 3.28– Графік показує розпакування стисненого вхідного трафіка з інтернету в локальну мережу

Після першої передачі файлу .doc розміром 7,6 Мб, пробуємо передати файл з включеним оптимізатором.

Таблиця 3.15 – Результати тестування

Величина що вимірюється	З філії в головний офіс	З головного офісу до філії
Формат	.doc	.doc
Розмір	7,6 Мб	7,6 Мб
Швидкість	1,8 Мбіт / сек	2,2 Мбіт / сек
Час	13 сек	09 сек

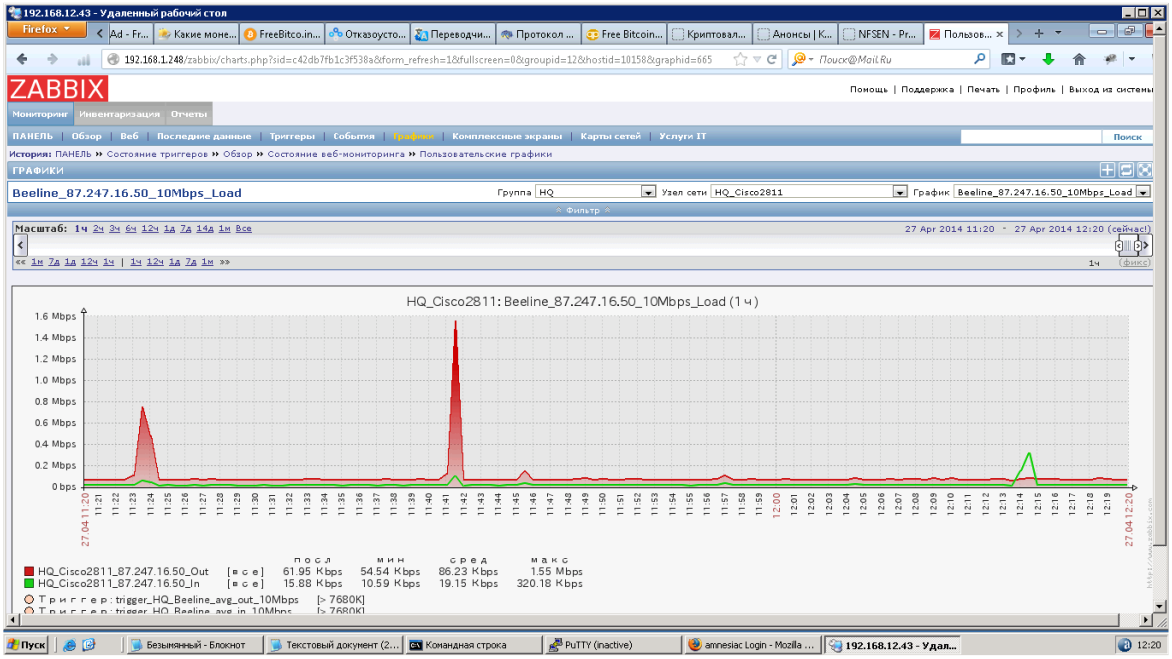

Рисунок 3.29– Графік навантаження каналу при повторній передачі з головного офісу в філія файлу .doc розміром 7,6 Мб з включеною оптимізацією

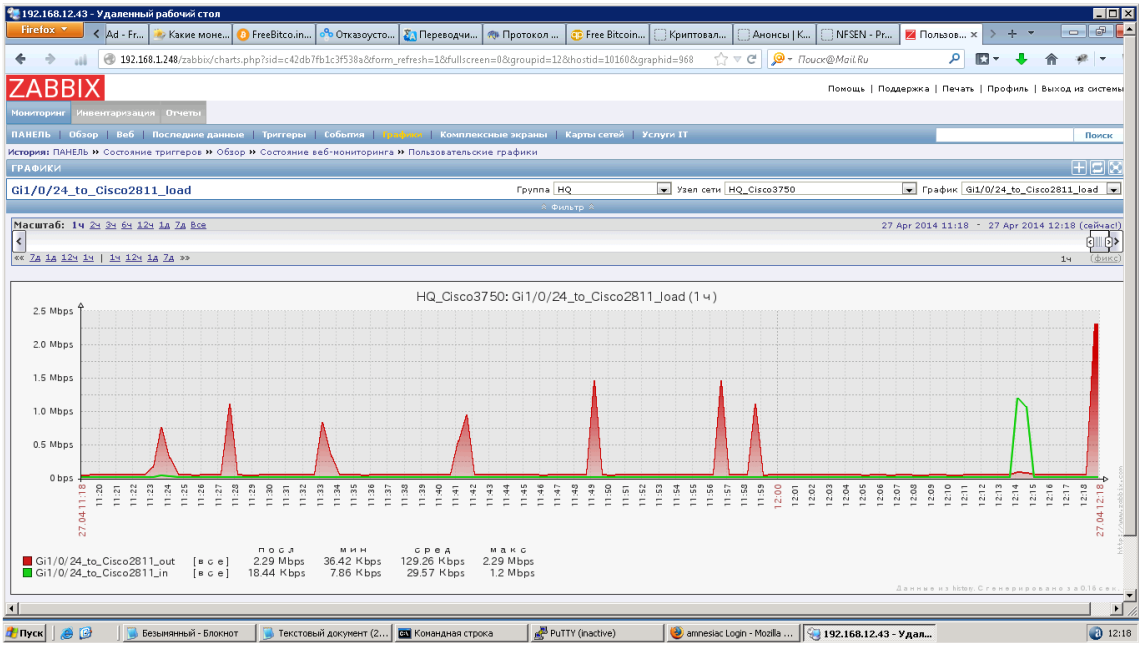

Рисунок 3.10– Графік вихідного стисненого трафіка при повторній передачі з головного офісу в філія файлу .doc розміром 7,6 Мб з включеною оптимізацією

4 РОЗРАХУНКОВА ЧАСТИНА

4.1 Визначення швидкості передачі даних

Згідно з приміткою 7 технічного регламенту ТР 2007/003 / BY [2] і міжнародного стандарту IEC 60027–2 одиницею кількості інформації є біт (російське позначення «біт», міжнародне – «bit») і байт. Відповідність між цими одиницями відбивається в рівність:

Швидкість передачі даних визначається, як відношення числа біт даних (L) до часу (t), за яке передані ці біти, т. Е.

$$C = L / t, \text{ біт / с.} \qquad (4.1)$$

Згідно таблиці 2 Технічного регламенту ТР 2007/003 / BY [2] множителями і приставками, використовуваними для утворення найменувань і позначень кратних одиниць СІ, які формувались на основі одиниці «біт / с», є наступні приставки:

1 кбіт / с = 1000 біт / с; 1 Мбіт / с = 1000 кбіт / с;

1 Гбіт / с = 1000 Мбіт / с; 1 Тбіт / с = 1000 Гбіт /

с;

На основі одиниці байт, позначеної «Б», є позначення в приставках в десятковій системі:

1 кБ / с = 1000 Б / с = 8000 біт / с; 1 МБ / с = 1000 кБ / с = 8000000 біт / с; Програмісти використовують множники і приставки в двійковій системі числення:

1 Кбіт / с = 210 біт / с = 1024 біт / с; 1 Мібіт / с = 210 Кібіт / с = 210. 210 біт / с = 1048576 біт / с; Однак існують одиниці, які використовуються на різних сайтах, але не належать до міжнародної системи одиниць (СІ).

Неправильне позначення одиниць швидкості передачі даних веде до недостовірних результатів вимірювання. Наприклад, на деяких сторінках web–сайтів призводять скорочені позначення: замість одиниці СІ «Кібіт / с» записують «Кбіт / с», або «КБ / с», що вносить плутанину у користувачів і розробників сайтів. Можна, наприклад, одиницю «МВ / s» прийняти за одиницю СІ «МіВ / s», опускаючи букву «і». Але в СІ це різні розмірності. Якщо порівнювати їх, то вони мають таке значення:

1 МВ / s = 1000000 В / s = 800000 bit / s; 1 МіВ / s = 1048576 bit / s. Порівнюючи отримані значення, можна побачити, що, приймаючи одиницю «МВ / s» за «МіВ / s», значення швидкості буде збільшено в 1048576/800000 = 1,31 рази в порівнянні з істинним.

Тому рекомендується приводити одиниці виміру швидкості передачі даних відповідно до ТР 2007/003 / BY [2]. Тоді можна гарантувати об'єктивність та достовірність вимірювань. Розрахунок швидкості передачі за формулою (2) представлений в загальному вигляді. Завдання оцінки швидкості передачі даних при наданні послуги доступу в мережу інтернет зводиться до виділення із загального потоку даних фізичного рівня числа біт L, що відноситься до потоку даних користувача (рисунок 4.2).

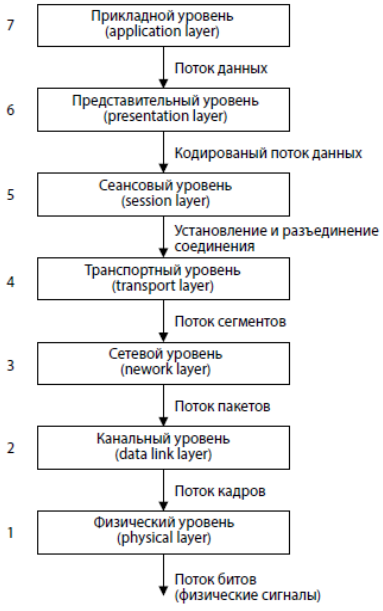

Рисунок 4.1–Мережева модель OSI (Open systems interconnection) [6]

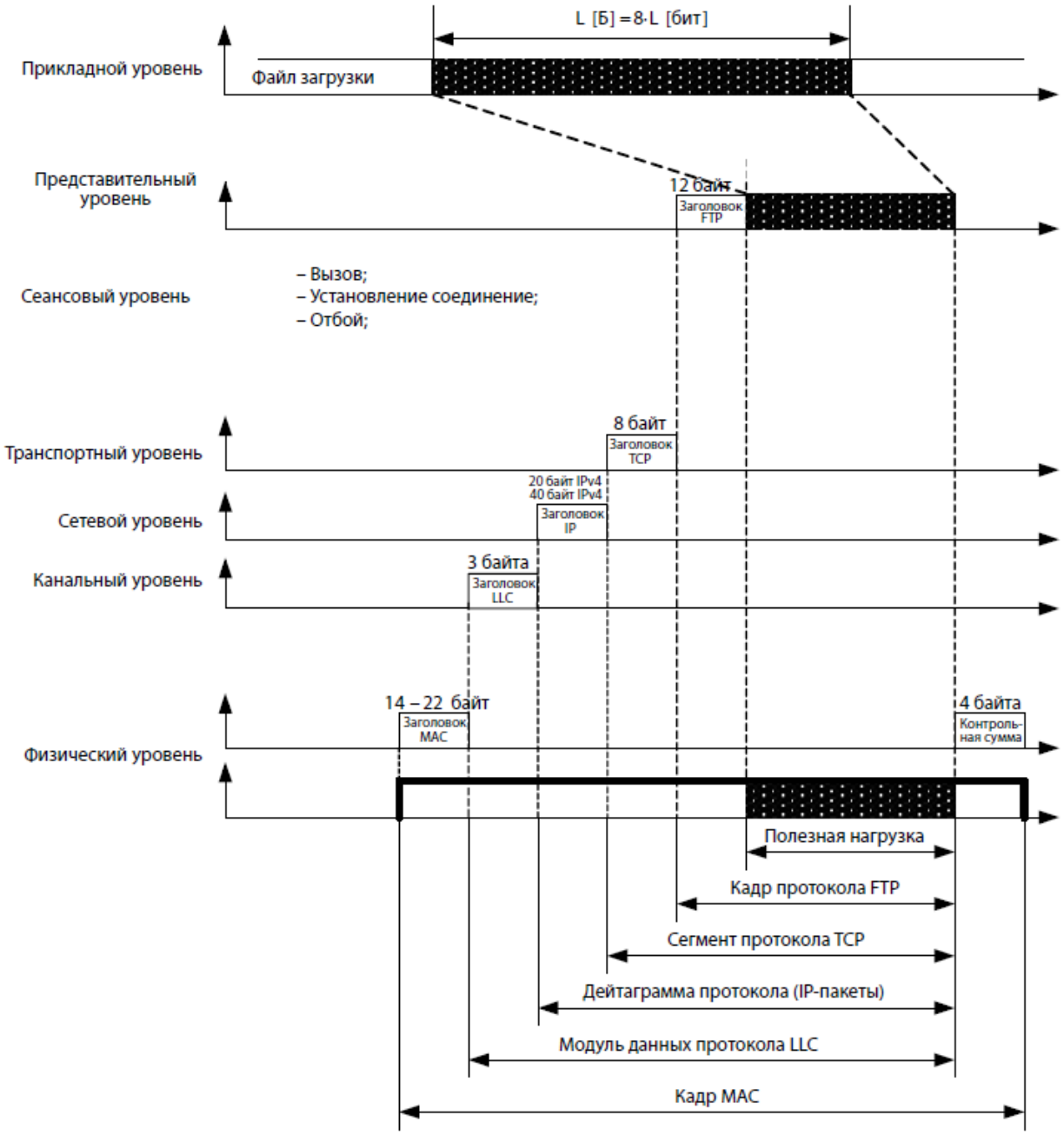

Рисунок 4.2–Тимчасова діаграма, яка пояснює процес формування потоків даних від прикладного рівня до фізичного [6]

Зробимо розрахунок, виходячи і наявних договорів на надання послуг від провайдерів Укртелеком і Тріолан. Середня швидкість завантаження каналу в головному офісі каналу Укртелеком складає на вихідний трафік 3Мбіт / сек, що входить 4Мбіт / сек.

1 Мб = 1024Кб = 1 Кб = 1024байт = 1 байт = 8біт 8 * 1024 * 1024 = 8388608 біт Розрахунок середньої швидкості передачі даних при вихідному трафіка через канал від провайдера Укртелеком:

1 Мбіт = 1048576 біт 1048576 * 3 = 3145728 біт З: =
3145728/3600 = 873,3 біт / сек.

Розрахунок середньої швидкості передачі даних при вхідному трафіка через канал від провайдера Укртелеком.

1 Мбіт = 1048576 біт 1048576 * 4 = 4194304 біт С2 =
4194304/3600 = 1165,08 біт / сек.

Розрахуємо середню швидкість передачі даних вхідного і вихідного трафіка за формулою

$$C_{ср}=\frac{C_1+C_2}{2}\text{ ,бит/сек} \qquad (4.2)$$

$$C_{ср}=\frac{873,3+1165,08}{2}=1019,19\text{ бит/сек}.$$

Середня швидкість завантаження каналу в головному офісі каналу Тріолан становить на вихідний трафік 1,5 Мбіт / сек, що входить 2,3Мбіт / сек.

Розрахунок середньої швидкості передачі даних при вихідному трафіка через канал від провайдера Тріолан:

1 Мбіт = 1048576 біт 1048576 * 1,5 = 1572864 біт С1
= 1572864/3600 = 436,9 біт / сек.

Розрахунок середньої швидкості передачі даних при вхідному трафіка через канал від провайдера Тріолан.

1 Мбіт = 1048576 біт 1048576 * 2,3 = 2411724,8 біт С2 = 2411724,8 / 3600 = 669,9 біт / сек. Розрахуємо середню швидкість передачі даних вхідного і вихідного трафіка за формулою:

$$C_{ср}=\frac{C_1+C_2}{2}\text{ ,бит/сек}$$

$$C_{ср}=\frac{436,9+669,9}{2}=553,41\text{ бит/сек}. \qquad (4.3)$$

4.2 Розрахунок колізій і їх вплив на продуктивність мережі

Надійне розпізнавання колізій усіма станціями мережі є необхідною умовою коректної роботи мережі Ethernet. З цією метою всі станції одночасно спостерігають за переданими по кабелю сигналами. Якщо передаються і спостерігаються сигнали відрізняються, то фіксується факт виявлення колізії.

Для збільшення ймовірності якнайшвидшого виявлення колізії всіма станціями мережі станція, яка виявила колізію, підсилює ситуацію колізії посилкою в мережу спеціальної послідовності з 32 біт, званої jam–послідовністю.

Виявила колізію передавальну станцію повинна припинити передачу і зробити паузу протягом короткого випадкового проміжку часу, що отримав назву усіченого експоненціального довічного алгоритму відстрочки.

4.3 Визначення співвідношення кількості робочих пакетів і колізій

Під робочими пакетами маються на увазі пакети, які досягають робочих станцій мережі без спотворень. При цьому змінним параметром був об-

раний розмір пакетів, який брав значення від 100 до 1500 з кроком 200. Установка параметрів моделювання проводилася в блоці параметрів прикладного рівня моделі (рисунок 4.1).

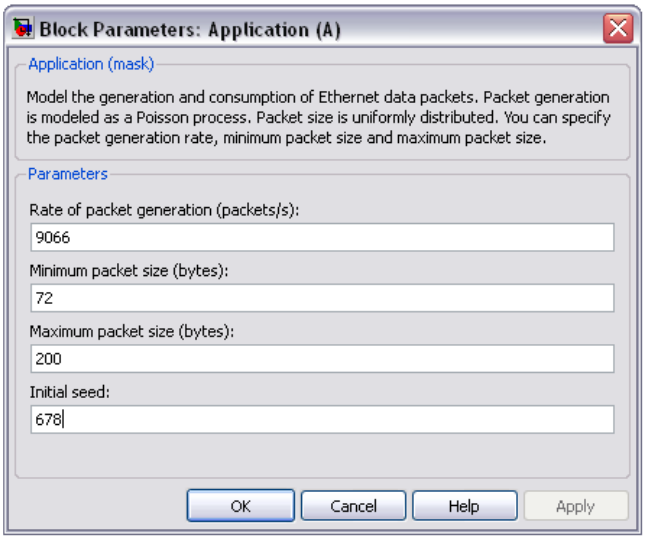

Рисунок 4.3 – Блок параметрів прикладного рівня робочої станції

Вікно Rate of packet generation (packets / s) дозволяє встановити норму генерування пакетів робочою станцією. Цей параметр може бути розрахований за формулою:

$$R = 106 / (Lf * 8 + (Tint–1)) – 0.1, (кадр / с.), \qquad (4.4)$$

де: Lf – максимальна довжина пакету в робочому діапазоні; (Наприклад, для блоку параметрів, зображеного на рис. 2, Lf = 200); Tint – міжкадровий інтервал.

У вікнах Minimum packet size (bytes) і Maximum packet size (bytes) встановлюються відповідно мінімальний і максимальний розмір пактів, які задають діапазон генерування розмірів пакетів датчиком псевдовипадкових чисел. Для параметра Lf = 100 (рис. 2.4) мінімальний і максимальний розмір кадру приймають значення 72 і 200 байт відповідно.

Моделювання виконувалось протягом 20 одиниць модельного часу. Результати моделювання представлені в таблиці 2.4 і на рисунку 2.5.

R = 106 / 0.0018 = 58780. R = 106 / 0.0042 = 25163.

R = 106 / 0.0018 = 15143.R = 106 / 0.0069 = 10640.

R = 106 / 0.0099 = 8083.R = 106 / 0.016 =

6456.

R = 106 / 0.019 = 5334. R = 106 / 0.023 = 4606.

Таблиця 4.1 – Вплив колізій на корисну пропускну здатність Ethernet

Розмір пакету, Lf	Число пакетів:		Відносний коефіцієнт робочих пакетів, D	Корисна пропускна здатність	
	робочих, R	Колізійних, C		розрахункова Мбіт/с, $P_{расч}$	робоча, Мбіт/с, $P_{раб}$
1	2	3	4	6	7
100	58780	2481	0.9595	6.61	6.35
300	25163	2656	0.9045	8.78	7.95
500	15143	2395	0.8634	9.26	7.99
700	10640	2292	0.8228	9.47	7.79
900	8083	2219	0.7846	9.58	7.52
1100	6456	2170	0.7484	9.66	7.23
1300	5334	2110	0.7166	9.71	6.96
1500	4606	2206	0.6762	**9.75**	**6.59**

Перший стовпець таблиці 4.1 призначений для подання розміру пакета. У наступних двох стовпчиках знаходяться кількості робочих і колізійних пакетів, отримані в результаті моделювання. У четвертому стовпці таблиці занесений відносний коефіцієнт робочих пакетів, який вираховується за формулою:

$$D = R / (R + C), \qquad (4,5)$$

де: R–число робочих пакетів; C – число колізійних пакетів.

D = 58780 / (58780 + 2481) = 0.9595; D = 25163 / (25163 + 2656) = 0.9045. D = 15143 / (15143 + 0.8634) = 0.8634; D = 10640 / (10640 + 2292) = 0.8228. D = 8083 / (8083 + 2219) = 0.7846; D = 6456 / (6456 + 2170) = 0.7484.

D = 5334 / (5334 + 2110) = 0.7166; D = 4606 / (4606 + 2206) = 0.6762.

На рисунку 4.4 представлена стовбикова діаграма, на якій показані кількості робочих пакетів і колізій в залежності від обсягу переданих по мережі пакетів.

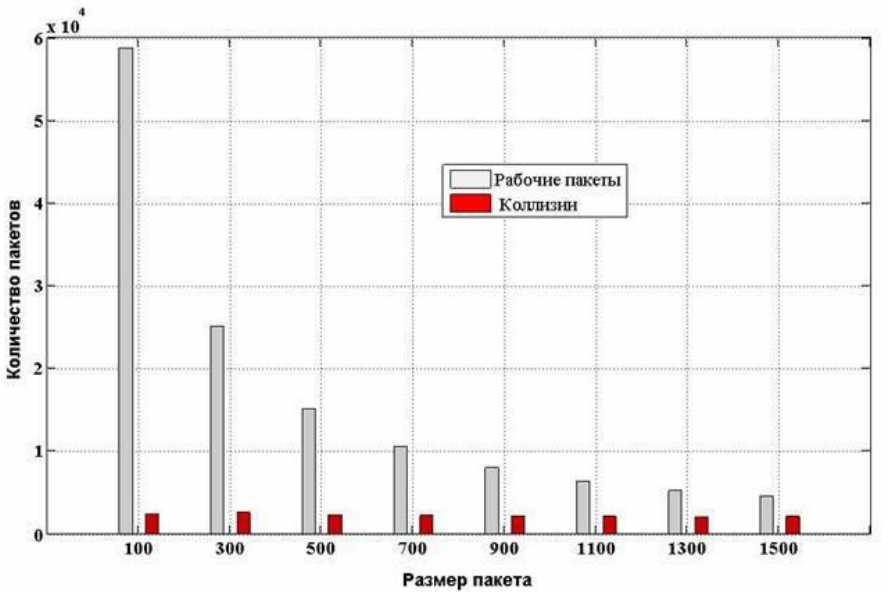

Рисунок 4.4 – Залежність кількості робочих пакетів і колізій від розміру пакета

Подальші дослідження моделі Ethernet були спрямовані на визначення впливу колізій на одну з основних характеристик мережі – корисна пропускна здатність. Під корисною пропускною здатністю протоколу розуміється максимальна швидкість передачі призначених для користувача даних, які переносяться полем даних пакета.

Корисна пропускна здатність представлена в стовпцях 6 і 7 таблиці 4.1, озаглавлених розрахункова і робоча. Розрахункова корисна пропускна здатність відповідає ідеальному випадку, коли в мережі відсутні колізії.

Розрахункова пропускна здатність обчислюється таким чином:

$$Ррасч = 106 / Lf * 8 + (Tint–1)) * 0.1, (пакет / с.), \quad (4,6)$$

Ррасч = 106 / 16,036 = 6.61; Ррасч = 106 / 12,07 = 8.78.
Ррасч = 106 / 11,44 = 9.26; Ррасч = 106 / 11,19 = 9.47.
Ррасч = 106 / 11,064 = 9.58; Ррасч = 106 / 10,97 = 9.66.
Ррасч = 106 / 10,19 = 9.71; Ррасч = 106 / 10,871 = 9.75.

Робоча пропускна здатність обчислюється таким чином:

$$Рраб = 106 * D / (Lf * 8 ++ (Tint–1)) * 0.1, (пакет / с.)], \quad (4,7)$$

де Е – відносний коефіцієнт робочих пакетів.

Рраб = 106 * 0.9595 / 16,036 = 6.35; Рраб = 106 * 0.9045 / 12,07 = 7.95.
Рраб = 106 * 0.8634 / 11,44 = 7.99; Рраб = 106 * 0.8228 / 11,19 = 7.79.
Рраб = 106 * 0.7846 / 11,064 = 7.52; Рраб = 106 * 0.7484 / 10,97 = 7.23.
Рраб = 106 * 0.7166 / 10,19 = 6.96; Рраб = 106 * 0.6762 / 10,871 = 6.59.

4.4 Оптимізація розподілу трафіку на основі імітаційної моделі

В даний час частка використання технології хмарних обчислень в сучасних бізнес процесах компаній неухильно зростає. Незважаючи на те, що це дозволяє знижувати вартість володіння і експлуатації IT інфраструктури, існує ряд проблем пов'язаних з управлінням центрами обробки даних. Однією з таких проблем є ефективність використання наявних у розпорядженні компаній обчислювальних і мережних ресурсів. Одним з напрямків оптимізації є процес управління трафіком хмарних додатків і сервісів в ЦОД. З

огляду на многозвенную архітектуру сучасних ЦОД, таке завдання досить не тривіальна. Перевагою сучасної інфраструктури віртуалізації є мож–ливість використання програмно–конфігуруються мереж і програмно–керованих сховищ даних. Однак існуючі алгоритмічні рішення при оптимізації не враховують ряд особливостей формування трафіку в мережі з декількома клас–самі додатків. В рамках проведеного дослідження вирішено завдання оптимізації розподілу трафіку хмарних додатків і сервісів для програмно–керованої інфраструктури віртуального ЦОД.

4.4.1 Матеріал та методи дослідження

Як правило, в програмно–керованої інфраструктурі віртуального ЦОД розміщуються декілька неоднорідних додатків і сервісів. На підставі цього припустимо, що, в мережі віртуального ЦОД присутній як мінімум три класи трафіку додатків, таких як: веб–додатки; case–додатки (прикладне ПО, доступне за DaaS або SaaS моделі); відео сервіси. При цьому в якості трафіку додатків будемо розглядати запити користувачів до кожного класу додатків. Для генерації запитів користувачів в імітаційної моделі застосуємо до кожного класу трафіку вагові коефіцієнти k1, k2, k3. Кожен з перерахованих коефіцієнтів дозволяє розділити заявки на класи і впливає на наступний набір параметрів: час виконання, маршрут в імітаційної моделі, пріоритет в черзі на обробку, інтенсивність надходження, а так же закон розподілу, згідно з яким здійснюється генерація трафіку певного класу.

Уявімо імітаційну модель програмно–керованої інфраструктури віртуального ЦОД у вигляді багатоканальної системи масового обслуговування. До її складу входить джерело заявок користувачів (I), черга (Qs) і планувальник (S), керуючий процесами розміщення і запуску додатків і сервісів (App), а також пул обчислювальних вузлів (Srv) і систем зберігання (Stg) ЦОД, що містять як самі додатки, так і необхідні їм дані. Схема СМО програмно–керованої інфраструктури віртуального ЦОД представлена на рисунку 4.5.

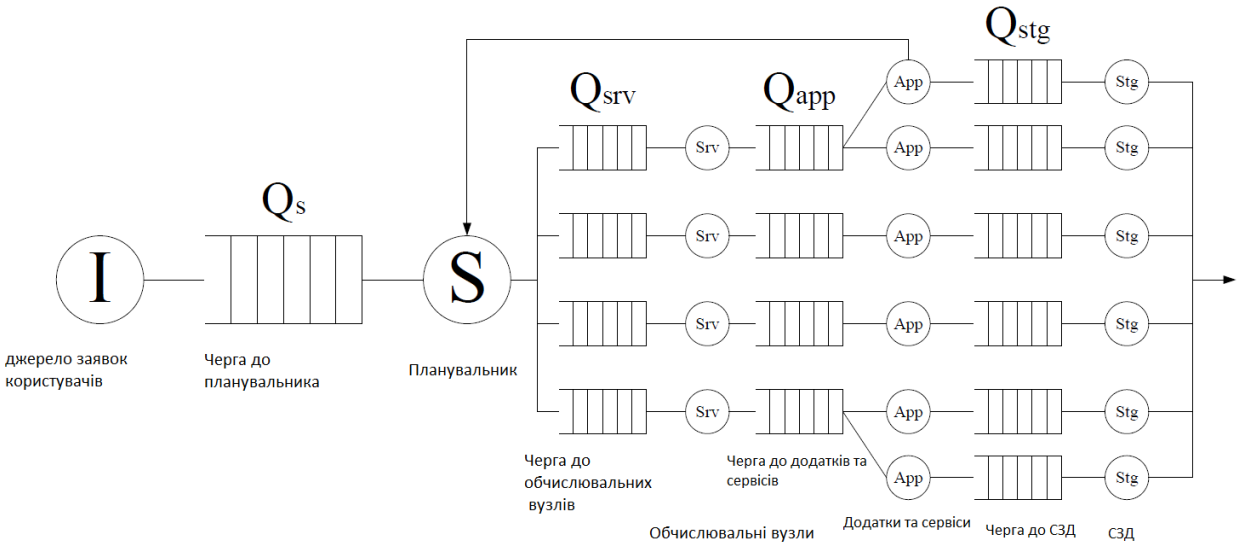

Рисунок 4.5 – Схема СМО програмно–керованої інфраструктури віртуального ЦОД

Модель СМО носить стохастичний характер. Для її роботи необхідно створити потік запитів користувачів до хмарним додаткам і сервісам, що враховує закони розподілу і інтенсивність надходження заявок для кожного класу хмарного додатка і сервісу.

Для вирішення завдання щодо оптимізації управління розміщенням додатків в хмарному середовищі віртуального ЦОД необхідно випередити закони розподілу трафіку для кожного класу додатків, а так же розподілити сам трафік по об'єктах доступу (віртуальним серверам, контейнерів і систем зберігання). Для цього необхідно встановити певний маршрут і побудувати для нього закон управління на тимчасовому інтервалі T = [t1, t2].

В динаміці трафік хмарних додатку–ний і сервісів в програмно–керованої інфраструктурі віртуального ЦОД можна описати наступною дискретною системою:

$$x_{i,j}(t+\Delta t) = x_{i,j}(t) - \sum_{k=1}^{K}\sum_{l=1}^{N} s_{i,j}(t)u_{i,l}^{j,k}(t) + \\ + \sum_{m=1}^{N} s_{m,i}(t)u_{m,l}^{j}(t) + y_{i,j}(t) \quad (4.8)$$

,

де N – кількість віртуальних вузлів в мережі; K – кількість класів додатків і сервісів в мережі; , $S_{i,j}(t)$ – пропускна здатність каналів між і–м обчислювальним вузлом і j–ой системою зберігання даних ($i \neq j$); $y_{i,j}(t)=\lambda_{i,j}(t)= \Delta t$ – обсяг трафіку (кількість запитів користувачів), що надходить в момент часу t на віртуальний вузол і і призначеної для передачі системі зберігання j; , $\lambda_{i,j}(t)=$ – інтенсивність надходить навантаження, яка визначається як сумарна інтенсивність потоку запитів користувачів, що підключаються до віртуального вузла і та провідних обмін з j–му вузлом системи зберігання даних; $u^{i,j}_{i,l}(t)$ – частка пропускної здатності каналу виділеного в сегменті програмно–конфігуруємої мережі (i, l), в момент часу t потоку запитів користувачів до додатка k–го класу, що здійснює роботу з даними в системі зберігання j.

4.4.2 Запропоновані алгоритми

Для вирішення оптимізаційної задачі нами розроблений алгоритм управління трафіком хмарних додатків і сервісів в програмно–керованої інфраструктурі віртуального центру обробки даних. У порівнянні з наявними аналогами алгоритм використовує евристичний аналіз потоків запитів, а також класифікацію трафіку по типу переданих даних в процесі роботи додатка, розміщеного в хмарної платформі. Гнучкість пропонованого рішення. Це дозволяє динамічно змінювати розміщення додатків в хмарної системі щодо фізичних пристроїв, що дає можливість надавати безперервний доступ до послуг і сервісів. Пропоноване рішення прозоро для клієнта і масштабує хмарні додатки на кілька віртуальних пристроїв зберігання. Це забезпечує скорочення часу відгуку програми, а так само підвищує відмовостійкість всієї системи

в цілому.

Формування програмно–керованих систем, що самоорганізуються сховищ даних на базі віртуальних машин і контейнерів дозволяє не тільки знизити ризики, пов'язані з втратою або недоступністю даних, але і забезпечує інтелектуальний аналіз затребуваності хмарних додатків. На базі отриманих даних формуються карти розміщення віртуальних машин і контейнерів, а так само правила для формування потоків в програмно–конфігурується мережі Віртуального ЦОД.

В основу алгоритму управління трафіком хмарних додатків і сервісів в програмно–керованої інфраструктурі покладена розроблена імітаційна модель. Отримана інформація про характер розподілення і інтенсивності надходження запитів піддається аналізу з застосуванням алгоритмів машинного навчання (Data Mining), заснованого на нейромережевому підході. В результаті якого формується карта оптимального розташування додатків і сервісів всередині самої хмарної платформи з прив'язкою до фізичних пристроїв, а так само створюється карта маршрутів для формування потоків трафіку з урахуванням затребуваності даних в системі зберігання. Шляхом аналізу двох карт і евристичного алгоритму прогнозування система управління віртуальним ЦОД приймається рішення про реконфігурацію структури хмарної платформи і перебудові маршрутів для виділених класів трафіку. При цьому обидві карти є динамічними об'єктами, які формувались не тільки в міру виникнення подій в програмно–керованої інфраструктурі, а й із заданим інтервалом часу $t\Delta$, підбираються індивідуально для кожної хмарної платформи. В рамках дослідження визначено найбільш оптимальний інтервал часу для аналізу і перестроювання карт, при якому робота системи буде найбільш ефективною.

При роботі з хмарними додатками і сервісами не виключена ситуація, при якій для обслуговування запиту користувача можуть бути задіяні відразу кілька типів ресурсів інфраструктури віртуального ЦОД з різними характеристиками доступу. При роботі з такими даними хмарної системі необхідно

здійснювати підготовку інфраструктури доступу для оптимізації часу відклику на запит. Для цього розроблений алгоритм управління трафіком хмарних додатків і сервісів в програмно–управ–неушкодженої інфраструктурі віртуального ЦОД в ході роботи будує план виконання запитів, тим самим підлаштовуючи кожен задіяний об'єкт інфраструктури під потік запитів користувачів.

В результаті, план виконання потоку запитів користувачів з однаковою інтенсивністю в різні моменти часу може бути складений по–різному. Перестроювання плану відбувається відповідно до затребуваних ресурсів, що дозволяє ефективно управляти розподілом і динамічним балансуванням навантаження у програмно–керуємій інфраструктурі віртуального ЦОД. Узагальнена блок–схема роботи алгоритму управління трафіком хмарних додатків і сервісів в програмно–керованої інфраструктурою віртуального центру обробки даних представлена на рисунку 4.6.

Всього в роботі алгоритму управління трафіком хмарних додатків і сервісів в програмно–керованої інфраструктурі віртуального центру обробки даних можна виділити три етапи планування та оптимізації, що виконуються для обслуговування запитів користувачів.

На першому етапі в плані виконання аналізуються характеристики запиту, що поступають від користувача до додатка, а саме, визначається тип програми та вид переданих даних. Запит даних є багатокомпонентним, тобто для організації каналу задіються відразу кілька елементів системи. Тому на другому етапі система управління визначає найбільш підходящі ресурси, здатні гарантувати виконання запиту. При цьому якщо серед ресурсів існує варіативність, то на основі інформації про прогноз затребуваності кожного з доступних варіантів ресурсу будується ранжирний список з прив'язкою до віртуального сховища даних. При цьому при побудові маршруту трафіку враховуються тільки найменш завантажені сховища даних і канали зв'язку.

Рисунок 4.6 – Узагальнена блок–схема роботи алгоритму управління трафіком хмарних додатків і сервісів в програмно–керованої інфраструктурі віртуального центру обробки даних

На третьому етапі роботи алгоритму аналізуються поточні параметри ресурсів і прогнозується час, необхідний на виконання поточного запиту. Отримані результати зберігаються для подальшого використання при самонавчанні алгоритму. У випадках високої ресурсоємності запиту для попередньої оцінки в алгоритмі застосовується поелементний аналіз об'єктів

доступу, що входять в запит.

4.4.3 Експериментальна частина

Для оцінки ефективності розробленого алгоритму управління трафіком хмарних додатків і сервісів в програмно–конфігуруємих сховищах даних віртуального ЦОД, нами проведено дослідження роботи хмарної системи, побудованої на базі Openstack з різними–ними параметрами. Для порівняння в експеримент використовувалися алгоритми, що застосовуються в хмарних системах для управління запуском додатків і розміщенням даних.

Для аналізу ефективності та продуктивності роботи алгоритмів на різних системах зберігання, нами визначено типові умови експерименту, що включають традиційні пристрої на магнітних дисках (HDD), твердотільні накопичувачі (SSD) і віртуальні сховища (SDS). Для Експериментальні дослідження створено прототип хмарний середовища, що включає в себе основні вузли, а так само програмні модулі для розроблених алгоритмів, що виконують перерозподіл запитів користувачів до даних у програмно–конфигуємому сховищи.

У хмарної системі OpenStack реалізований модуль, який застосовує розроблені алгоритм управління трафіком хмарних додатків і сервісів в програмно–конфігуруються сховищах даних віртуального ЦОД для раціонального використання обчислювальних ресурсів хмарної системи і ефективного розміщення віртуальних машин по фізичним вузлів, а так само пов'язаних з ними даних. В ході експерименту для аналізу даних створений потік запитів, аналогічний реальним запитам користувачів до хмарним додаткам, заснований на даних лог–записів доступу до певних видів ресурсів з класифікацією за типами програм і структурі запиту.

Ретроспектива відтворюваних запитів склала 3 роки, при цьому для навантажувального експерименту застосовувалися усереднені дані. Отримані дані розподілені на пул віртуальних машин за наступними критеріями: тип

клієнта, який здійснив звернення до даних, тип сервісу, затребуваного при підключенні. При цьому кількість одночасних запитів, що надійшли в систему, склало 100000, що відповідає максимальному числу потенціальних користувачів системи.

Усі сформовані запити відтворювались послідовно на трьох експериментальних майданчиках. Дане обмеження введено в зв'язку необхідністю зіставлення результатів з фізичними системами зберігання даних, які не здатні до реконфігурації. Основною відмінністю експериментальних майданчиків є використання твердотільних накопичувачів.

Крім майданчиків для аналізу ефективності сформовано 3 групи експериментів, спрямованих на інтенсивне виконання запитів з читання (експеримент 1), записи (експеримент 2) і одночасних операціях читання і запису даних (експеримент 3) для кожного класу додатка.

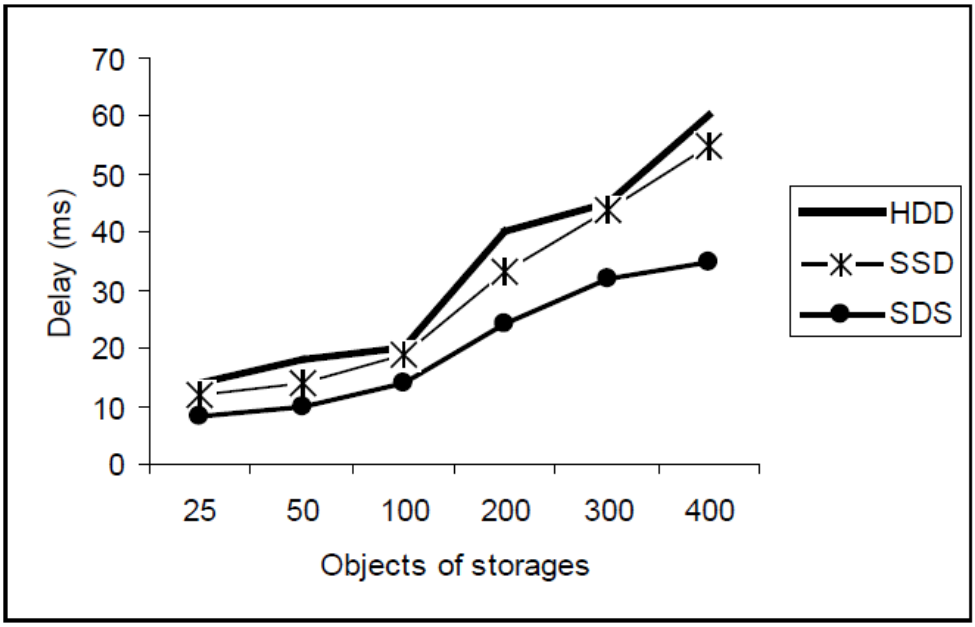

Рисунок 4.7 Аналіз часу відгуку додатків при виконанні запитів на читання даних (експеримент 1)

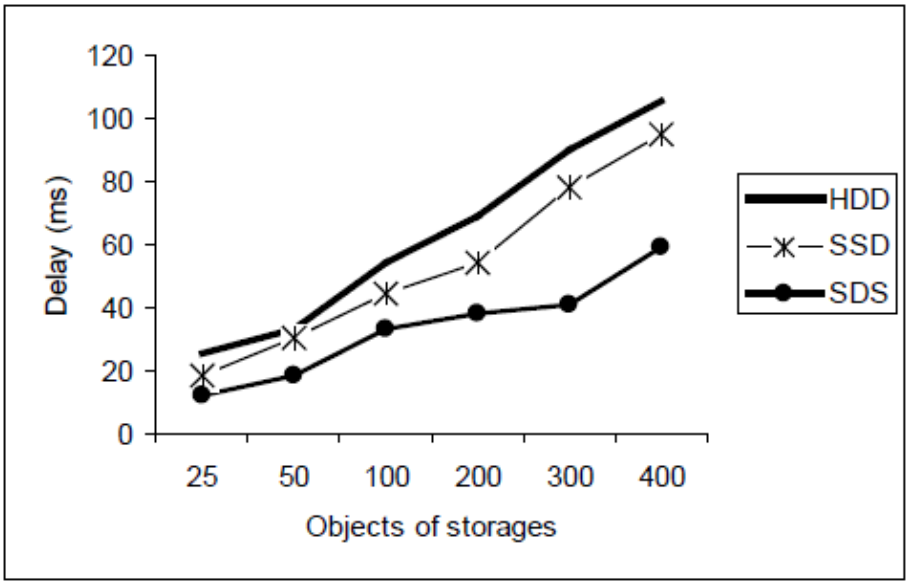

Рисунок 4.8 Аналіз часу відгуку додатків при виконанні запитів на запис даних (експеримент 2)

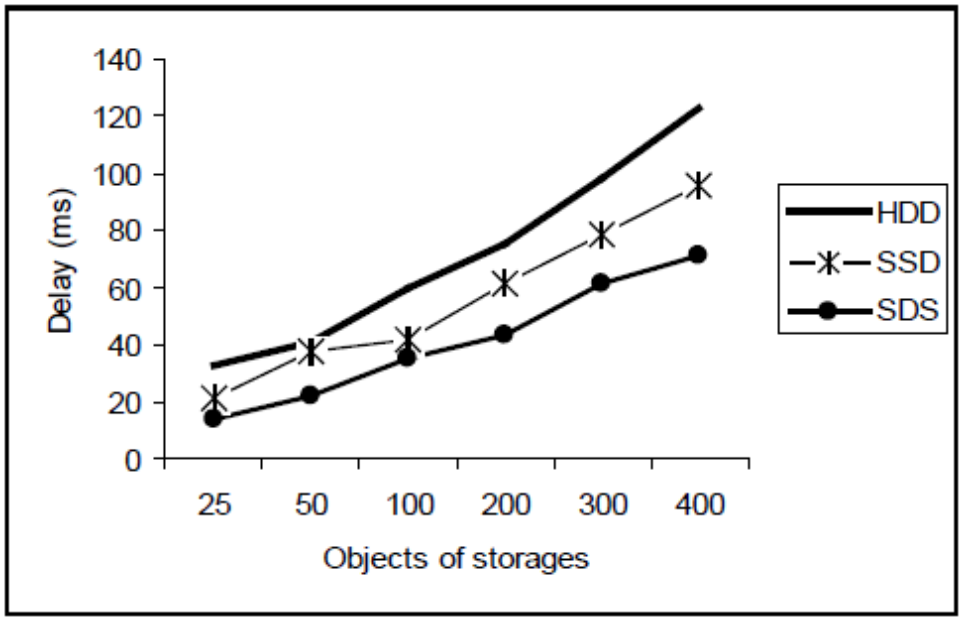

Рисунок 4.9 Аналіз часу відгуку додатків при одночасному виконанні запитів на читання і запису даних (експеримент 3)

Час експерименту склало одну годину, що відповідає найбільш тривалого періоду часу пікового навантаження системи, зафіксованому у

реальному трафіку. Аналіз даних експериментальних досліджень довів, що управління трафіком хмарних додатків і сервісів в програмно–конфігуруються сховищах даних віртуального ЦОД із використанням користуванням запропонованого алгоритму більш ефективно незалежно від типу фізичних пристроїв, які використовуються в сховище даних.

Отримані дані підтверджують цільових перевірок доцільність застосування розробленого алгоритму для надання ефективного доступу до додатків і сервісам хмарних систем. При цьому за результатами проведених експериментів можна так само можна зробити висновок про зниження на 20–25% кількості відмов при обслуговуванні запитів додатками і сервісами, розміщений–ними в програмно–керованих сховищах даних, що дозволяють гнучко адаптувати інфраструктуру в залежності від навантаження.

ВИСНОВКИ

У атестаційній роботі була розглянута модернізація мережі зв'язку результаті оптимізації трафіка WAN мережі (як транспортної магістралі хмарного сервісу) та отримання її повної відмово стійкості. У теоретичній частині були розглянуті 3 сценарії щодо оптимізації трафіка WAN мереж. Із зазначенням переваг і недоліків кожного. В експериментальній частині були проведені повні установки і настройки кожного з цих проектів із зазначенням повного списку дій і прикладів. Після чого було проведено тестування з кожного окремо, і вибрано оптимальне мережеве програмне забезпечення по оптимізації трафіка.

В результаті проведеного дослідження вирішено завдання оптимізації розподілу трафіку хмарних додатків і сервісів для програмно–керованої інфраструктури віртуального ЦОД. Запропоновано імітаційну модель, що дозволяє описати трафік в програмно–конфігуруються сегментах мережі ЦОД, що беруть участь в обробки запитів користувачів до додатків і сервісам розташованих в мережевому середовищі, що включає в себе гетерогенну хмарну платформу і програмно–конфігуровані сховища даних. Розроблена модель дозволила реалізувати алгоритм керування трафіком хмарних додатків і оптимізувати доступ системі зберігання, за рахунок ефективного вико–користування каналу для передачі даних.

Запропоновано імітаційну модель, що дозволяє описати трафік в програмно–конфігуруються сегментах мережі ЦОД, що беруть участь в обробки запитів користувачів до додатків і сервісам розташованих мережевому середовищі, що включає в себе гетерогенну хмарну платформу і програмно–конфігуровані сховища даних. Розроблена модель дозволила реалізувати алгоритм керування трафіком хмарних додатків і оптимізувати доступ системі зберігання, за рахунок ефективного використання каналу для передачі даних. В ході експериментальних досліджень встановлено, що застосування розроб-

леного алгоритму дозволяє скоротити час відгуку хмарних додатків і сервісів, і, як наслідок, підвищена сить продуктивність обробки запитів користувачів і знизити кількість відмов.

Підводячи підсумок зробленого у атестаційній роботі, хотілося б відзначити, що результати вийшли краще, ніж очікували. Перш за все, була з нуля розроблена схема і план модернізації мережі організації, встановлено мережеве програмне забезпечення Riverbed Steelhead. За результатами тестувань даний проект показав хороші результати.

Також в ході виконання роботи було придбано багато корисних навичок по роботі з мережевим обладнанням та віртуалізацією машин.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Іванісенко І.М, Євтушенко А.В. Оптимізація управління розподілом трафіка у хмарній інфраструктурі на базі імітаційної моделі // Тези доповідей восьмої міжнародної науково–технічної конференції.– Черкаси–Харків–Баку–Бельсько–Бяла:2020,– Т.1.–с. 69.

2. Інокентій Руденко «Маршрутизатор CISCO для IP – мереж». – М .: 2013р. – 450 с.

3. Джером Ф. Дімарціо «Маршрутизатор CISCO» посібник для самостійного вивчення. – Санкт–Петербург – Москва. Лорі 2013. 428 с.

4. Амато, Віто «Основи організації мереж CISCO» – М .: Видавничий дім "Вільямс" 2012. – 152 с.

5. Тодд Леммле, Кевін Хейлз «Налаштування комутаторів CISCO». – М.: Видавництво «Лорі». – 2012. – 142.

5. Оліфер В.Г., Оліфер Н.А. "Комп'ютерні мережі. Принципи, тіхнології, протоколи ». Підручник. – Санкт–Петербург – Пітер, 2013. 923 с.

6. Одом У. Официальное руководство Cisco по подготовке к сертификационным экзаменам CCNA ICND2 200–10: маршрутизация и коммутация / Уэнделл Одом. – Москва: ООО "И.Д. Вильямс", 2015. – 736 с. – (академическое издание).

7. Кеннеді Кларк, Кевин Гамільтон «Принципи комутації в локальних мережах CISCO» – Москва – Санкт–Петербург – Київ 2003р.

8. Девід Хьюкабі, Стів Мак–Квері «Керівництво CISCO по конфігурації комутаторів Catalyst» – Москва – Санкт–Петербург – Київ 2003р .;

9. Браун С. Віртуальні приватні мережі. М Видавництво «Лорі», 2004. – 508 с .

15. Лагутін В.С., Костров В.О. Оцінка характеристик пропускної здатності мультисервісних пакетних мереж при реалізації технології поділу типів навантаження // Електрозв'язок. – № 3. – 2003. – с. 28–32 ..

16. Михалевич І.Ф., Сичов К.І., Лузін В.Ю. Оптимізація пропускної здатності корпоративних мереж зв'язку // Електрозв'язок. – № 10. – 2003. – с. 36–39 .

17. Thomas Barnett, Jr. 2018 Complete VNI Forecast Update – What's Trending? / Thomas Barnett, Jr.. // Cisco public. – 2018.

18. Шварцман В.О. Вибір технології передачі і комутації в мультисервісних мережах на основі оптичних кабелів // Електрозв'язок. – № 8. –2003. – с. 33–39.

19. Czarkowski M. Traffic Type Influence on Performance of OSPF QoS Routing / M. Czarkowski, S. Kaczmarek, M. Wolff // Journal of telecommunication and information technology. – 2013. – Vol. 3. – pp. 19–28.

20. Харитонов В.Х. Мультисервісна мережа і методи комутації // Електрозв'язок. – № 1. – 2004. – с. 17–25 ..

21. Бейлі Д., Райт Е. волоконна оптика: теорія і практика волоконна оптика: теорія і практика / Пер. з англ. – М: КУДИЦ–ПРЕСС,2008 р. , 320 с.

22. Ivanisenko Igor. Using cloud storage services in decision support system / Igor Ivanisenko, Yu Kobytska // Проблеми автоматизації: Третя міжнар. наук.–техн. конф., 12–13 лист. 2015: тези доп. – Черкаси–Баку–Бельсько–Бяла–Полтава, Украина. – С. 29.